```
USE [UniversityDB]
GO
/****** Object:  Role [aspnet_Membership_BasicAccess]    Script Date: 08/16/2011 01:24:34 ↙
    ******/
CREATE ROLE [aspnet_Membership_BasicAccess] AUTHORIZATION [dbo]
GO
/****** Object:  Role [aspnet_Membership_FullAccess]    Script Date: 08/16/2011 01:24:34 *↙
    *****/
CREATE ROLE [aspnet_Membership_FullAccess] AUTHORIZATION [dbo]
GO
/****** Object:  Role [aspnet_Membership_ReportingAccess]    Script Date: 08/16/2011 01:24↙
    :34 ******/
CREATE ROLE [aspnet_Membership_ReportingAccess] AUTHORIZATION [dbo]
GO
/****** Object:  Role [aspnet_Personalization_BasicAccess]    Script Date: 08/16/2011 01: ↙
    24:34 ******/
CREATE ROLE [aspnet_Personalization_BasicAccess] AUTHORIZATION [dbo]
GO
/****** Object:  Role [aspnet_Personalization_FullAccess]    Script Date: 08/16/2011 01:24↙
    :34 ******/
CREATE ROLE [aspnet_Personalization_FullAccess] AUTHORIZATION [dbo]
GO
/****** Object:  Role [aspnet_Personalization_ReportingAccess]    Script Date: 08/16/2011 ↙
    01:24:34 ******/
CREATE ROLE [aspnet_Personalization_ReportingAccess] AUTHORIZATION [dbo]
GO
/****** Object:  Role [aspnet_Profile_BasicAccess]    Script Date: 08/16/2011 01:24:34 ***↙
    ***/
CREATE ROLE [aspnet_Profile_BasicAccess] AUTHORIZATION [dbo]
GO
/****** Object:  Role [aspnet_Profile_FullAccess]    Script Date: 08/16/2011 01:24:34 ****↙
    **/
CREATE ROLE [aspnet_Profile_FullAccess] AUTHORIZATION [dbo]
GO
/****** Object:  Role [aspnet_Profile_ReportingAccess]    Script Date: 08/16/2011 01:24:34↙
     ******/
CREATE ROLE [aspnet_Profile_ReportingAccess] AUTHORIZATION [dbo]
GO
/****** Object:  Role [aspnet_Roles_BasicAccess]    Script Date: 08/16/2011 01:24:34 *****↙
    */
CREATE ROLE [aspnet_Roles_BasicAccess] AUTHORIZATION [dbo]
GO
/****** Object:  Role [aspnet_Roles_FullAccess]    Script Date: 08/16/2011 01:24:34 ******↙
    /
CREATE ROLE [aspnet_Roles_FullAccess] AUTHORIZATION [dbo]
GO
/****** Object:  Role [aspnet_Roles_ReportingAccess]    Script Date: 08/16/2011 01:24:34 *↙
    *****/
CREATE ROLE [aspnet_Roles_ReportingAccess] AUTHORIZATION [dbo]
GO
/****** Object:  Role [aspnet_WebEvent_FullAccess]    Script Date: 08/16/2011 01:24:34 ***↙
    ***/
CREATE ROLE [aspnet_WebEvent_FullAccess] AUTHORIZATION [dbo]
GO
/****** Object:  Schema [Academics]    Script Date: 08/16/2011 01:24:25 ******/
CREATE SCHEMA [Academics] AUTHORIZATION [dbo]
GO
/****** Object:  Schema [aspnet_Membership_BasicAccess]    Script Date: 08/16/2011 01:24: ↙
    25 ******/
CREATE SCHEMA [aspnet_Membership_BasicAccess] AUTHORIZATION                             ↙
    [aspnet_Membership_BasicAccess]
GO
/****** Object:  Schema [aspnet_Membership_FullAccess]    Script Date: 08/16/2011 01:24:25↙
     ******/
CREATE SCHEMA [aspnet_Membership_FullAccess] AUTHORIZATION [aspnet_Membership_FullAccess]
GO
/****** Object:  Schema [aspnet_Membership_ReportingAccess]    Script Date: 08/16/2011 01:↙
```

```
    24:25 ******/
CREATE SCHEMA [aspnet_Membership_ReportingAccess] AUTHORIZATION                    ↙
    [aspnet_Membership_ReportingAccess]
GO
/****** Object:  Schema [aspnet_Personalization_BasicAccess]    Script Date: 08/16/2011 01↙
    :24:25 ******/
CREATE SCHEMA [aspnet_Personalization_BasicAccess] AUTHORIZATION                   ↙
    [aspnet_Personalization_BasicAccess]
GO
/****** Object:  Schema [aspnet_Personalization_FullAccess]    Script Date: 08/16/2011 01:↙
    24:25 ******/
CREATE SCHEMA [aspnet_Personalization_FullAccess] AUTHORIZATION                    ↙
    [aspnet_Personalization_FullAccess]
GO
/****** Object:  Schema [aspnet_Personalization_ReportingAccess]    Script Date: 08/16/   ↙
    2011 01:24:25 ******/
CREATE SCHEMA [aspnet_Personalization_ReportingAccess] AUTHORIZATION               ↙
    [aspnet_Personalization_ReportingAccess]
GO
/****** Object:  Schema [aspnet_Profile_BasicAccess]    Script Date: 08/16/2011 01:24:25 *↙
    *****/
CREATE SCHEMA [aspnet_Profile_BasicAccess] AUTHORIZATION [aspnet_Profile_BasicAccess]
GO
/****** Object:  Schema [aspnet_Profile_FullAccess]    Script Date: 08/16/2011 01:24:25 **↙
    ****/
CREATE SCHEMA [aspnet_Profile_FullAccess] AUTHORIZATION [aspnet_Profile_FullAccess]
GO
/****** Object:  Schema [aspnet_Profile_ReportingAccess]    Script Date: 08/16/2011 01:24:↙
    25 ******/
CREATE SCHEMA [aspnet_Profile_ReportingAccess] AUTHORIZATION                       ↙
    [aspnet_Profile_ReportingAccess]
GO
/****** Object:  Schema [aspnet_Roles_BasicAccess]    Script Date: 08/16/2011 01:24:25 ***↙
    ***/
CREATE SCHEMA [aspnet_Roles_BasicAccess] AUTHORIZATION [aspnet_Roles_BasicAccess]
GO
/****** Object:  Schema [aspnet_Roles_FullAccess]    Script Date: 08/16/2011 01:24:25 ****↙
    **/
CREATE SCHEMA [aspnet_Roles_FullAccess] AUTHORIZATION [aspnet_Roles_FullAccess]
GO
/****** Object:  Schema [aspnet_Roles_ReportingAccess]    Script Date: 08/16/2011 01:24:25↙
     ******/
CREATE SCHEMA [aspnet_Roles_ReportingAccess] AUTHORIZATION [aspnet_Roles_ReportingAccess]
GO
/****** Object:  Schema [aspnet_WebEvent_FullAccess]    Script Date: 08/16/2011 01:24:25 *↙
    *****/
CREATE SCHEMA [aspnet_WebEvent_FullAccess] AUTHORIZATION [aspnet_WebEvent_FullAccess]
GO
/****** Object:  Schema [Assessment]    Script Date: 08/16/2011 01:24:25 ******/
CREATE SCHEMA [Assessment] AUTHORIZATION [dbo]
GO
/****** Object:  Schema [Finance]    Script Date: 08/16/2011 01:24:25 ******/
CREATE SCHEMA [Finance] AUTHORIZATION [dbo]
GO
/****** Object:  Schema [Personals]    Script Date: 08/16/2011 01:24:25 ******/
CREATE SCHEMA [Personals] AUTHORIZATION [dbo]
GO
/****** Object:  Schema [Registry]    Script Date: 08/16/2011 01:24:25 ******/
CREATE SCHEMA [Registry] AUTHORIZATION [dbo]
GO
/****** Object:  Schema [SetUp]    Script Date: 08/16/2011 01:24:25 ******/
CREATE SCHEMA [SetUp] AUTHORIZATION [dbo]
GO
/****** Object:  FullTextCatalog [UniversityCatalog]    Script Date: 08/16/2011 01:24:34 *↙
    *****/
CREATE FULLTEXT CATALOG [UniversityCatalog]WITH ACCENT_SENSITIVITY = ON
AUTHORIZATION [dbo]
```

```
GO
/****** Object:  UserDefinedFunction [Academics].[GetTotalDays]    Script Date: 08/16/2011↙
     01:24:33 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE FUNCTION [Academics].[GetTotalDays](@StartDate DateTime2(7),@EndDate DateTime2(7)) ↙
    RETURNS INT
AS
BEGIN
        DECLARE @Diff INT=NULL
        SET @Diff=(SELECT DATEDIFF(dd,@StartDate,@EndDate) FROM [Academics].[Semesters])
        RETURN ISNULL(@Diff,0)
END
GO
/****** Object:  UserDefinedFunction [SetUp].[FnGetProgramDescription]    Script Date: 08/↙
    16/2011 01:24:33 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Godwin Mathias
--Reviewer          :   Godwin Mathias
--Date Created      :   04/28/2011
--Last Updated      :   04/28/2011
--Last Updated By   :   Godwin Mathias
--Description       :   Function for retrieving discounted amount from [SetUp].[Modules] ↙
    Table

CREATE FUNCTION [SetUp].[FnGetProgramDescription](@Code BIGINT) RETURNS NVARCHAR(256)
AS
BEGIN
        DECLARE @Decription NVARCHAR(256)
        SET @Decription= (SELECT [ProgName]+' ('+[AwardInViewName]+')' FROM [SetUp].      ↙
    [Programs] WHERE ([Code]=@Code))
        RETURN (@Decription)
END
GO
/****** Object:  Table [SetUp].[Universities]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [SetUp].[Universities](
    [Code] [nvarchar](50) NOT NULL,
    [Description] [nvarchar](256) NULL,
    [UniversityTypeCode] [nvarchar](50) NULL,
    [BannerCode] [nvarchar](50) NULL,
    [LogoCode] [nvarchar](50) NULL,
    [Url] [nvarchar](256) NULL,
    [CountryCode] [nvarchar](50) NULL,
    [StateCode] [nvarchar](50) NULL,
    [LgaCode] [nvarchar](50) NULL,
    [Motto] [nvarchar](50) NULL,
    [Notes] [nvarchar](max) NULL,
    [EstablishedYear] [int] NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Universities] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,     ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
INSERT [SetUp].[Universities] ([Code], [Description], [UniversityTypeCode], [BannerCode], ↙
    [LogoCode], [Url], [CountryCode], [StateCode], [LgaCode], [Motto], [Notes],           ↙
    [EstablishedYear], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],   ↙
    [DeletedOn], [DeletedBy]) VALUES (N'ACU', N'Ajayi Crowther University', N'UTC-24',    ↙
    NULL, NULL, N'http://localhost/UniversityPortalWeb/', N'NGN', N'OY', N'ATB', N        ↙
    'Scientia Probitas', N'', 2005, CAST(0x0700000000007B340B AS DateTime2), N'ademola',  ↙
    CAST(0x074E0987107095340B AS DateTime2), N'', 0, NULL, NULL)
/****** Object:  UserDefinedFunction [SetUp].[FnGetCourseLevel]    Script Date: 08/16/2011↙
     01:24:33 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE FUNCTION [SetUp].[FnGetCourseLevel](@Code BIGINT) RETURNS NVARCHAR(256)
AS
BEGIN
    DECLARE @Description NVARCHAR(256)
    SET @Description= (SELECT Convert(NVARCHAR(4),[Code])+' '+'(Course Numbers :'+Convert ↙
    (NVARCHAR(4),NumberingLowerBound)+'-'+ Convert(NVARCHAR(4),NumberingUpperBound)+')'   ↙
    FROM [SetUp].[CourseNumbering] WHERE ([Code]=@Code))
    RETURN @Description
END
GO
/****** Object:  Table [SetUp].[CourseNumbering]    Script Date: 08/16/2011 01:24:32 *****↙
    */
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [SetUp].[CourseNumbering](
    [Code] [bigint] NOT NULL,
    [NumberingLowerBound] [bigint] NULL,
    [NumberingUpperBound] [bigint] NULL,
    [Notes] [nvarchar](500) NULL,
    [UniversityCode] [nvarchar](50) NOT NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
    [Description]  AS ([SetUp].[FnGetCourseLevel]([Code])),
 CONSTRAINT [PK_CourseNumbering] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,     ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
INSERT [SetUp].[CourseNumbering] ([Code], [NumberingLowerBound], [NumberingUpperBound],   ↙
    [Notes], [UniversityCode], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy],      ↙
    [Deleted], [DeletedOn], [DeletedBy]) VALUES (100, 1001, 1999, N'Course numbering for  ↙
    100 level students', N'ACU', CAST(0x07117A46744696340B AS DateTime2), N'', CAST       ↙
    (0x07FAF131364796340B AS DateTime2), N'', 0, CAST(0x0750DE23394796340B AS DateTime2), ↙
    N'')
INSERT [SetUp].[CourseNumbering] ([Code], [NumberingLowerBound], [NumberingUpperBound],   ↙
    [Notes], [UniversityCode], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy],      ↙
    [Deleted], [DeletedOn], [DeletedBy]) VALUES (200, 2001, 2999, N'Course numbering for  ↙
    200 level students', N'ACU', CAST(0x07001E48574796340B AS DateTime2), N'', CAST       ↙
    (0x07B7FD10614796340B AS DateTime2), N'', 0, NULL, NULL)
INSERT [SetUp].[CourseNumbering] ([Code], [NumberingLowerBound], [NumberingUpperBound],   ↙
    [Notes], [UniversityCode], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy],      ↙
    [Deleted], [DeletedOn], [DeletedBy]) VALUES (300, 3001, 3999, N'Course numbering for  ↙
    300 level students', N'ACU', CAST(0x07C68CF7724796340B AS DateTime2), N'', NULL, NULL,↙
     0, NULL, NULL)
INSERT [SetUp].[CourseNumbering] ([Code], [NumberingLowerBound], [NumberingUpperBound],   ↙
    [Notes], [UniversityCode], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy],      ↙
    [Deleted], [DeletedOn], [DeletedBy]) VALUES (400, 4001, 4999, N'Course numbering for  ↙
    400 level students', N'ACU', CAST(0x076299A2884796340B AS DateTime2), N'', NULL, NULL,↙
     0, NULL, NULL)
INSERT [SetUp].[CourseNumbering] ([Code], [NumberingLowerBound], [NumberingUpperBound],   ↙
    [Notes], [UniversityCode], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy],      ↙
    [Deleted], [DeletedOn], [DeletedBy]) VALUES (500, 5001, 5999, N'Course numbering for  ↙
    500 level students', N'ACU', CAST(0x076C51349E4796340B AS DateTime2), N'', NULL, NULL,↙
     0, NULL, NULL)
INSERT [SetUp].[CourseNumbering] ([Code], [NumberingLowerBound], [NumberingUpperBound],   ↙
    [Notes], [UniversityCode], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy],      ↙
    [Deleted], [DeletedOn], [DeletedBy]) VALUES (600, 6001, 6999, N'Course numbering for  ↙
    600 level students', N'ACU', CAST(0x0784A3BD3F4996340B AS DateTime2), N'', CAST       ↙
    (0x07175822754996340B AS DateTime2), N'', 0, NULL, NULL)
INSERT [SetUp].[CourseNumbering] ([Code], [NumberingLowerBound], [NumberingUpperBound],   ↙
    [Notes], [UniversityCode], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy],      ↙
    [Deleted], [DeletedOn], [DeletedBy]) VALUES (700, 7001, 7999, N'Course numbering for  ↙
    700 level students', N'ACU', CAST(0x070693C44E4996340B AS DateTime2), N'', CAST       ↙
    (0x0724194C7D4996340B AS DateTime2), N'', 0, NULL, NULL)
INSERT [SetUp].[CourseNumbering] ([Code], [NumberingLowerBound], [NumberingUpperBound],   ↙
    [Notes], [UniversityCode], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy],      ↙
    [Deleted], [DeletedOn], [DeletedBy]) VALUES (800, 8001, 8999, N'Course numbering for  ↙
    800 level students', N'ACU', CAST(0x0752B08C594996340B AS DateTime2), N'', CAST       ↙
    (0x07D1697B8C4996340B AS DateTime2), N'', 0, NULL, NULL)
INSERT [SetUp].[CourseNumbering] ([Code], [NumberingLowerBound], [NumberingUpperBound],   ↙
    [Notes], [UniversityCode], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy],      ↙
    [Deleted], [DeletedOn], [DeletedBy]) VALUES (900, 9001, 9999, N'Course numbering for  ↙
    900 level students', N'ACU', CAST(0x07CD9A46674996340B AS DateTime2), N'', CAST       ↙
    (0x07CF59F1914996340B AS DateTime2), N'', 0, NULL, NULL)
/****** Object:  StoredProcedure [SetUp].[SPCourseNumberingSelect]    Script Date: 08/16/ ↙
    2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for retrieving values from SetUp.CourseNumbering ↙
    Table

CREATE PROC [SetUp].[SPCourseNumberingSelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[CourseNumbering] WHERE ([Code] = @Code) AND (   ↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
            SELECT * FROM [SetUp].[CourseNumbering]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [SetUp].[CourseNumbering]
            WHERE
                (
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPCourseNumberingSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPCourseNumberingInsertUpdate]    Script Date:  ↙
    08/16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into SetUp.CourseNumbering     ↙
    Table

CREATE PROC [SetUp].[SPCourseNumberingInsertUpdate]
(
    @Code BIGINT=NULL,
    @NumberingLowerBound BIGINT=NULL,
    @NumberingUpperBound BIGINT=NULL,
    @Notes NVARCHAR(500)=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL

)
AS
```

```
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[CourseNumbering]  WHERE ([Code] = @Code) AND↙
     ([UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
                INSERT INTO [SetUp].[CourseNumbering]
                (
                    [Code],
                    [NumberingLowerBound],
                    [NumberingUpperBound],
                    [Notes],
                    [UniversityCode],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]

                )
                VALUES
                (
                    @Code,
                    @NumberingLowerBound,
                    @NumberingUpperBound,
                    @Notes,
                    @UniversityCode,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [SetUp].[CourseNumbering]
                SET
                    [NumberingLowerBound]=@NumberingLowerBound,
                    [NumberingUpperBound]=@NumberingUpperBound,
                    [Notes]=@Notes,
                    [UniversityCode]=@UniversityCode,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH
```

```
GRANT EXECUTE ON [SetUp].[SPCourseNumberingInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPCourseNumberingDeletePermanently]    Script   ↙
    Date: 08/16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from SetUp.CourseNumbering   ↙
    Table

CREATE PROC [SetUp].[SPCourseNumberingDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[CourseNumbering] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [SetUp].[CourseNumbering]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPCourseNumberingDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPCourseNumberingDelete]    Script Date: 08/16/ ↙
    2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
```

```
--Reviewer          :   Godwin Mathias
--Date Created      :   07/27/2011
--Last Updated      :   07/27/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from SetUp.  ↙
    CourseNumbering Table

CREATE PROC [SetUp].[SPCourseNumberingDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[CourseNumbering] WHERE (([Code] = @Code) AND↙
     (Deleted=@Deleted)))
            BEGIN
            UPDATE [SetUp].[CourseNumbering]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPCourseNumberingDelete] TO PUBLIC
GO
/****** Object:  Table [SetUp].[Parameters]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [SetUp].[Parameters](
    [Code] [nvarchar](50) NOT NULL,
    [Description] [nvarchar](256) NULL,
 CONSTRAINT [PK_Parameters] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,  ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY],
 CONSTRAINT [UNIQUEParameters] UNIQUE NONCLUSTERED
(
    [Description] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,  ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
INSERT [SetUp].[Parameters] ([Code], [Description]) VALUES (N'ACT', N'Actions')
INSERT [SetUp].[Parameters] ([Code], [Description]) VALUES (N'AIV', N'Award in View')
INSERT [SetUp].[Parameters] ([Code], [Description]) VALUES (N'CS', N'Civil Status')
INSERT [SetUp].[Parameters] ([Code], [Description]) VALUES (N'CSS', N'Course Status')
INSERT [SetUp].[Parameters] ([Code], [Description]) VALUES (N'DU', N'Duration Units')
INSERT [SetUp].[Parameters] ([Code], [Description]) VALUES (N'ENT', N'Entity Types')
INSERT [SetUp].[Parameters] ([Code], [Description]) VALUES (N'GD', N'Gender')
INSERT [SetUp].[Parameters] ([Code], [Description]) VALUES (N'HSC', N'Health Status Code')
INSERT [SetUp].[Parameters] ([Code], [Description]) VALUES (N'MOS', N'Mode of Study')
INSERT [SetUp].[Parameters] ([Code], [Description]) VALUES (N'PRG', N'Program Types')
INSERT [SetUp].[Parameters] ([Code], [Description]) VALUES (N'SEM', N'Semesters')
INSERT [SetUp].[Parameters] ([Code], [Description]) VALUES (N'SFT', N'Staff Type')
INSERT [SetUp].[Parameters] ([Code], [Description]) VALUES (N'SSS', N'Status')
INSERT [SetUp].[Parameters] ([Code], [Description]) VALUES (N'SDT', N'Student Types')
INSERT [SetUp].[Parameters] ([Code], [Description]) VALUES (N'SUB', N'Subjects')
INSERT [SetUp].[Parameters] ([Code], [Description]) VALUES (N'TKT', N'Ticket Types')
INSERT [SetUp].[Parameters] ([Code], [Description]) VALUES (N'UTC', N'University Types')
/****** Object:  UserDefinedFunction [SetUp].[FnGetDescriptionCode]    Script Date: 08/16/↙
    2011 01:24:33 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE FUNCTION [SetUp].[FnGetDescriptionCode](@Code NVARCHAR(50),@Id INT) RETURNS  ↙
    NVARCHAR(50)
AS
BEGIN
        RETURN @Code+ '-'+ Convert(NVARCHAR(50),@Id)
END
GO
/****** Object:  Table [SetUp].[Descriptions]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [SetUp].[Descriptions](
    [Code]  AS ([SetUp].[FnGetDescriptionCode]([ParametersCode],[ID])),
    [ParametersCode] [nvarchar](50) NULL,
    [Name] [nvarchar](256) NULL,
    [Notes] [nvarchar](256) NULL,
    [ID] [int] IDENTITY(1,1) NOT NULL,
 CONSTRAINT [PK_Descriptions] PRIMARY KEY CLUSTERED
(
    [ID] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,  ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
SET IDENTITY_INSERT [SetUp].[Descriptions] ON
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'UTC', N ↙
    'Private University', N'', 24)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'UTC', N ↙
    'Federal University', N'', 25)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'UTC', N ↙
    'State University', N'', 26)
```

```
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'UTC', N ↙
    'Federal Polytechnic', N'', 27)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SUB', N ↙
    'English', N'Arts', 29)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SUB', N ↙
    'Mathematics', N'Science', 30)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SUB', N ↙
    'Physics', N'Science', 31)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SUB', N ↙
    'Economics', N'Social Science', 32)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SUB', N ↙
    'Government', N'Arts', 33)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SUB', N ↙
    'History', N'Arts', 34)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SUB', N ↙
    'Chemistry', N'Science', 35)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SUB', N ↙
    'Biology', N'Science', 36)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SUB', N ↙
    'Geography', N'Social Science', 37)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SUB', N ↙
    'Account', N'Social Science', 38)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SUB', N ↙
    'Commerce', N'Social Science', 39)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SUB', N ↙
    'Yoruba Language', N'Arts', 40)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SUB', N ↙
    'Islamic Studies', N'Arts', 41)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SUB', N ↙
    'Literature-In-English', N'Arts', 42)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SUB', N ↙
    'Agricultural Science', N'Science', 43)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SUB', N ↙
    'Bible Knowledge/C.R.K/C.R.S', N'Arts', 44)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SUB', N ↙
    'Education (P & P)', N'Arts', 45)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SUB', N ↙
    'Hausa Language', N'Arts', 46)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SUB', N ↙
    'French', N'Arts', 47)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SUB', N ↙
    'Music', N'Arts', 48)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SUB', N ↙
    'Applied Electricity', N'Science', 49)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SUB', N ↙
    'Basic Electricity', N'Science', 50)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SUB', N ↙
    'Electronics', N'Science', 51)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SUB', N ↙
    'Statistics', N'Science', 52)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SUB', N ↙
    'Introduction to Business Management', N'Social Science', 53)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SUB', N ↙
    'Shorthand', N'Social Science', 54)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SUB', N ↙
    'Typewriting', N'Social Science', 55)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'ENT', N ↙
    'Staff', N'', 58)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'ENT', N ↙
    'Students', NULL, 59)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SDT', N ↙
    'New Student', NULL, 60)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SDT', N ↙
    'Returning Student', NULL, 61)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SFT', N ↙
    'Academic Staff', NULL, 62)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SFT', N ↙
    'Non Academic Staff', NULL, 63)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'TKT', N ↙
    'Application', NULL, 64)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'TKT', N ↙
    'Library', NULL, 65)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'TKT', N ↙
    'Registration', NULL, 66)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'TKT', N ↙
    'Fines', N'k', 67)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'TKT', N ↙
    'Examinations', NULL, 68)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'TKT', N ↙
    'Hostel Accomodation', NULL, 69)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SEM', N ↙
    'First ', N'', 72)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SEM', N ↙
    'Second', N'', 73)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SEM', N ↙
    'Summer', NULL, 74)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SSS', N ↙
    'Active', NULL, 75)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SSS', N ↙
    'InActive', NULL, 76)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'GD', N  ↙
    'Male', NULL, 77)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'GD', N  ↙
    'Female', NULL, 78)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'CS', N  ↙
    'Single', NULL, 79)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'CS', N  ↙
    'Married', N'', 80)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'CS', N  ↙
    'Divorced', N'', 81)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'CS', N  ↙
    'Widow', NULL, 82)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'CS', N  ↙
    'Widower', NULL, 83)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'HSC', N ↙
    'Disable', NULL, 84)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'HSC', N ↙
    'Normal', NULL, 85)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'ACT', N ↙
    'Add New Record', N'', 86)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'ACT', N ↙
    'Update Existing Record', N'', 87)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'ACT', N ↙
    'Delete Record', N'', 88)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'ACT', N ↙
    'View Record', N'', 89)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'ACT', N ↙
    'Lock Record', N'Lock record so that it cannot be edited', 90)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'ACT', N ↙
    'Grant Approval', N'', 91)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'ACT', N ↙
    'Reject Approval', N'', 94)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'MOS', N ↙
    'Full Time', N'F', 96)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'MOS', N ↙
    'Part Time', N'P', 97)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'MOS', N ↙
    'Weekend', N'W', 98)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'MOS', N ↙
    'Sandwich / Long Vacation', N'S', 99)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'MOS', N ↙
    'Occasional', N'O', 100)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'MOS', N ↙
    'Exchange', N'X', 101)
```

```
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'MOS', N ↙
    'Correspondence', N'C', 102)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'MOS', N ↙
    'Distance Learning', N'D', 103)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'AIV', N ↙
    'B.A.', N'Bachelor of Arts', 104)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'AIV', N ↙
    'M.A. ', N'Master of Arts
', 105)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'AIV', N ↙
    'B.B.S. ', N'Bachelor of Business Science
', 106)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'AIV', N ↙
    'B.C.L.', N'Bachelor of Civil Law
', 107)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'AIV', N ↙
    'B.Lit., B.Litt., or Lit.B. ', N'Bachelor of Letters (or Literature)
', 108)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'AIV', N ↙
    'L.L.B. ', N'Bachelor of Laws

', 109)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'AIV', N ↙
    'B.Sc.', N'Bachelor of Science
', 110)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'AIV', N ↙
    'D.Sc.', N'Doctor of Science
', 111)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'AIV', N ↙
    'J.D.', N'Doctor of Law
', 112)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'AIV', N ↙
    'L.H.D.', N'Doctor of Humanities
', 113)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'AIV', N ↙
    'L.L.D.', N'Doctor of Laws', 114)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'AIV', N ↙
    'M.B.A.', N'Master of Business Administration
', 115)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'AIV', N ↙
    'M.S. or M.Sc.', N'Master of Science
', 116)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'AIV', N ↙
    'Ph.D.', N'Doctor of Philosophy
', 117)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'AIV', N ↙
    'Ph.G.', N'Graduate in Pharmacy', 118)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'AIV', N ↙
    'OND', N'Ordinary National Diploma', 119)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'AIV', N ↙
    'ND', N'National Diploma', 120)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'AIV', N ↙
    'HND', N'Higher National Diploma', 121)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'AIV', N ↙
    'NCE', N'National Certificate of Education', 122)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'AIV', N ↙
    'HCE', N'Higher Certificate in Education', 123)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'PRG', N ↙
    'Degree', N'', 124)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'PRG', N ↙
    'Doctorate Degree', N'', 125)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'PRG', N ↙
    'Masters Degree', N'', 126)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'PRG', N ↙
    'National Diploma', N'', 127)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'PRG', N ↙
    'Higher National Diploma', N'', 128)
```

```
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'PRG', N ↙
    'Remedials', N'', 129)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SSS', N ↙
    'Enabled', N'', 130)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SSS', N ↙
    'Disabled', N'', 131)
GO
print 'Processed 100 total records'
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SSS', N ↙
    'Locked', N'', 132)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'SSS', N ↙
    'Un Locked', N'', 133)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'AIV', N ↙
    'REMS', N'Remedials', 134)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'CSS', N ↙
    'Compulsory', N'', 135)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'CSS', N ↙
    'Required', N'', 136)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'CSS', N ↙
    'Elective', N'', 137)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'CSS', N ↙
    'Pre-Requisite', N'', 138)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'CSS', N ↙
    'Concurrent', N'', 139)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'DU', N  ↙
    'Year(s)', N'yr', 140)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'DU', N  ↙
    'Month(s)', N'mm', 141)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'DU', N  ↙
    'Week(s)', N'wk', 142)
INSERT [SetUp].[Descriptions] ([ParametersCode], [Name], [Notes], [ID]) VALUES (N'DU', N  ↙
    'Day(s)', N'dy', 143)
SET IDENTITY_INSERT [SetUp].[Descriptions] OFF
/****** Object:  UserDefinedFunction [Academics].[GetSemesterStatusDescription]    Script ↙
    Date: 08/16/2011 01:24:33 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE FUNCTION [Academics].[GetSemesterStatusDescription](@Code NVARCHAR(50)) RETURNS    ↙
    NVARCHAR(50)
AS
BEGIN
        DECLARE @Name NVARCHAR(50)=NULL
        SET @Name=(SELECT [Name] FROM [SetUp].[Descriptions] WHERE ([Code]=@Code))
        RETURN ISNULL(@Name,'N/A')
END
GO
/****** Object:  Table [Academics].[Semesters]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Academics].[Semesters](
    [Code] [bigint] IDENTITY(1,1) NOT NULL,
    [SemesterCode] [nvarchar](50) NULL,
    [SessionCode] [bigint] NULL,
    [StartDate] [datetime2](7) NULL,
    [EndDate] [datetime2](7) NULL,
    [RegistrationClosingDate] [datetime2](7) NULL,
    [SemesterStatus] [nvarchar](50) NULL,
    [TotalDays]  AS ([Academics].[GetTotalDays]([StartDate],[EndDate])),
    [Applicable] [bit] NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
    [SemesterStatusDescription]  AS ([Academics].[GetSemesterStatusDescription](   ↙
    [SemesterStatus])),
 CONSTRAINT [PK_Semesters] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,     ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
SET IDENTITY_INSERT [Academics].[Semesters] ON
INSERT [Academics].[Semesters] ([Code], [SemesterCode], [SessionCode], [StartDate],  ↙
    [EndDate], [RegistrationClosingDate], [SemesterStatus], [Applicable], [UniversityCode]↙
    , [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn],  ↙
    [DeletedBy]) VALUES (1, N'SEM-72', 1, CAST(0x07000000000092340B AS DateTime2), CAST  ↙
    (0x07000000000015350B AS DateTime2), CAST(0x070000000000D8340B AS DateTime2), N'SSS-75↙
    ', 1, N'ACU', CAST(0x07000000000092340B AS DateTime2), N'chiriksmat', NULL, NULL, 0,  ↙
    NULL, NULL)
SET IDENTITY_INSERT [Academics].[Semesters] OFF
/****** Object:  StoredProcedure [Academics].[SPSemestersSelect]    Script Date: 08/16/  ↙
    2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Academics.Semesters  ↙
    Table

CREATE PROC [Academics].[SPSemestersSelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Deleted BIT=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Academics].[Semesters] WHERE (([Code] = @Code) AND (   ↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted)))
            BEGIN
            SELECT * FROM [Academics].[Semesters]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Academics].[Semesters]
            WHERE
                (
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Academics].[SPSemestersSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Academics].[SPSemestersInsertUpdate]    Script Date: 08/↙
    16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Academics.Semesters Table

CREATE PROC [Academics].[SPSemestersInsertUpdate]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @SessionCode BIGINT=NULL,
    @SemesterCode NVARCHAR(50)=NULL,
    @StartDate DATETIME2(7)=NULL,
    @EndDate DATETIME2(7)=NULL,
    @RegistrationClosingDate DATETIME2(7)=NULL,
    @SemesterStatus NVARCHAR(50)=NULL,
    @Applicable BIT=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Academics].[Semesters] WHERE ([Code] = @Code) AND ( ↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
                INSERT INTO [Academics].[Semesters]
                (
                    [UniversityCode],
                    [SessionCode],
                    [SemesterCode],
                    [StartDate],
                    [EndDate],
                    [RegistrationClosingDate],
                    [SemesterStatus],
                    [Applicable],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]

                )
                VALUES
                (
                    @UniversityCode,
                    @SessionCode,
                    @SemesterCode,
                    @StartDate,
                    @EndDate,
                    @RegistrationClosingDate,
                    @SemesterStatus,
                    @Applicable,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted

                )
            END
        ELSE
            BEGIN
                UPDATE [Academics].[Semesters]
                SET
                    [UniversityCode]=@UniversityCode,
                    [SessionCode]=@SessionCode,
                    [SemesterCode]=@SemesterCode,
                    [StartDate]=@StartDate,
                    [EndDate]=@EndDate,
                    [RegistrationClosingDate]=@RegistrationClosingDate,
                    [SemesterStatus]=@SemesterStatus,
                    [Applicable]=@Applicable,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted

                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()
```

```
RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Academics].[SPSemestersInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Academics].[SPSemestersDeletePermanently]    Script Date↙
    : 08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Academics.Semesters    ↙
    Table

CREATE PROC [Academics].[SPSemestersDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Academics].[Semesters] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Academics].[Semesters]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Academics].[SPSemestersDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Academics].[SPSemestersDelete]    Script Date: 08/16/   ↙
    2011 01:24:27 ******/
SET ANSI_NULLS ON
```

```
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Academics.  ↙
    Semesters Table

CREATE PROC [Academics].[SPSemestersDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Academics].[Semesters] WHERE ([Code] = @Code) AND       ↙
    (Deleted=@Deleted))
            BEGIN
            UPDATE [Academics].[Semesters]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Academics].[SPSemestersDelete] TO PUBLIC
GO
/****** Object:  UserDefinedFunction [SetUp].[FnGetDescriptionName]    Script Date: 08/16/↙
    2011 01:24:33 ******/
SET ANSI_NULLS ON
GO
```

```
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Godwin Mathias
--Reviewer          :   Godwin Mathias
--Date Created      :   04/28/2011
--Last Updated      :   04/28/2011
--Last Updated By   :   Godwin Mathias
--Description       :   Function for retrieving discounted amount from [SetUp].[Modules]  ↙
    Table

CREATE FUNCTION [SetUp].[FnGetDescriptionName](@Code NVARCHAR(50)) RETURNS NVARCHAR(256)
AS
BEGIN
        DECLARE @DescriptionName Nvarchar(256)
        SET @DescriptionName = (SELECT [Name] FROM [SetUp].[Descriptions] WHERE Code=@    ↙
    Code)
        RETURN @DescriptionName
END
GO
/****** Object:  Table [SetUp].[SubCourses]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [SetUp].[SubCourses](
    [Code] [bigint] IDENTITY(1,1) NOT NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [FacultyCode] [bigint] NULL,
    [DepartmentCode] [bigint] NULL,
    [CourseCode] [bigint] NULL,
    [ProgramCode] [bigint] NULL,
    [SemesterCode] [nvarchar](50) NULL,
    [StatusCode] [nvarchar](50) NULL,
    [LevelCode] [bigint] NULL,
    [Practical] [int] NULL,
    [Laboratory] [int] NULL,
    [Credit] [int] NULL,
    [Title] [nvarchar](256) NULL,
    [PrefixCode] [nvarchar](50) NULL,
    [SubNo] [int] NULL,
    [SubCode]  AS ([PrefixCode]+CONVERT([nvarchar](4),[SubNo],(0))),
    [Notes] [nvarchar](500) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
    [StatusDescription]  AS ([SetUp].[FnGetDescriptionName]([StatusCode])),
 CONSTRAINT [PK_SubCourses] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,             ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
SET IDENTITY_INSERT [SetUp].[SubCourses] ON
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (1, N'ACU', 1, 8, 1, 1, N'SEM-72', N'CSS-135', 100, 0, 0, 2, N'Effective Communication↙
     Skills I', N'GES', 1102, N'', CAST(0x0708BAF1528A97340B AS DateTime2), N'', CAST     ↙
    (0x0719B7B5A68798340B AS DateTime2), N'', 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],   ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (2, N'ACU', 1, 8, 1, 1, N'SEM-72', N'CSS-135', 100, 0, 0, 1, N'Use Of Library', N'GES'↙
    , 1102, N'', CAST(0x07B45B90BC8E97340B AS DateTime2), N'', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],   ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (3, N'ACU', 1, 8, 1, 1, N'SEM-72', N'CSS-136', 100, 0, 0, 0, N'Use of French I', N'GES↙
    ', 1103, N'', CAST(0x0778F77D328F97340B AS DateTime2), N'', CAST(0x07C2BF3C798B98340B ↙
    AS DateTime2), N'', 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],   ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (4, N'ACU', 1, 8, 1, 1, N'SEM-72', N'CSS-135', 100, 0, 0, 2, N'The English Language', ↙
    N'ENG', 1101, N'', CAST(0x07E23829638F97340B AS DateTime2), N'', CAST    ↙
    (0x071386E12A8798340B AS DateTime2), N'', 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],   ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (5, N'ACU', 1, 8, 1, 1, N'SEM-72', N'CSS-135', 100, 0, 0, 2, N'Introduction to    ↙
    Nigerian Literature in English I', N'ENG', 1102, N'', CAST(0x071E2E5D858F97340B AS   ↙
    DateTime2), N'', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],   ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (6, N'ACU', 1, 8, 1, 1, N'SEM-72', N'CSS-135', 100, 0, 0, 2, N'Elements of Drama', N  ↙
    'ENG', 1103, N'', CAST(0x070231DF948F97340B AS DateTime2), N'', NULL, NULL, 0, NULL,  ↙
    NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],   ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (7, N'ACU', 1, 8, 1, 1, N'SEM-72', N'CSS-135', 100, 0, 0, 3, N'Poetry in English', N  ↙
    'ENG', 1104, N'', CAST(0x070D854EA88F97340B AS DateTime2), N'', NULL, NULL, 0, NULL,  ↙
    NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],   ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (8, N'ACU', 1, 8, 1, 1, N'SEM-72', N'CSS-135', 100, 0, 0, 2, N'Theatre Workshop', N   ↙
    'ENG', 1105, N'', CAST(0x071B44FAB68F97340B AS DateTime2), N'', NULL, NULL, 0, NULL,  ↙
    NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],   ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (9, N'ACU', 1, 8, 1, 1, N'SEM-73', N'CSS-135', 100, 0, 0, 2, N'Effective Communication↙
     Skills II', N'GES', 1201, N'', CAST(0x073073021F9097340B AS DateTime2), N'', NULL,   ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],   ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (10, N'ACU', 1, 8, 1, 1, N'SEM-73', N'CSS-136', 100, 0, 0, 0, N'Use of French II', N  ↙
    'GES', 1202, N'', CAST(0x07A4C706529097340B AS DateTime2), N'', NULL, NULL, 0, NULL,  ↙
    NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],   ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (11, N'ACU', 1, 8, 1, 1, N'SEM-73', N'CSS-135', 100, 0, 0, 3, N'Introduction to   ↙
    Computers', N'GES', 1203, N'', CAST(0x0739F795679097340B AS DateTime2), N'', NULL,   ↙
    NULL, 0, NULL, NULL)
```

```
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (12, N'ACU', 1, 8, 1, 1, N'SEM-73', N'CSS-135', 100, 0, 0, 2, N'Spoken English', N'ENG↙
    ', 1201, N'', CAST(0x0785AE0C759097340B AS DateTime2), N'', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (13, N'ACU', 1, 8, 1, 1, N'SEM-73', N'CSS-135', 100, 0, 0, 2, N'Introduction to ↙
    Nigerian Literature in English II', N'ENG', 1202, N'', CAST(0x078032C9B19097340B AS ↙
    DateTime2), N'', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (14, N'ACU', 1, 8, 1, 1, N'SEM-73', N'CSS-135', 100, 0, 0, 3, N'Introductory English ↙
    Grammar and Composition', N'ENG', 1203, N'', CAST(0x07E5D99DEE9097340B AS DateTime2), ↙
    N'', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (15, N'ACU', 1, 8, 1, 1, N'SEM-73', N'CSS-135', 100, 0, 0, 3, N'Prose Literature', N ↙
    'ENG', 1204, N'', CAST(0x07B72BA7FD9097340B AS DateTime2), N'', NULL, NULL, 0, NULL, ↙
    NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (16, N'ACU', 1, 8, 1, 1, N'SEM-73', N'CSS-135', 100, 0, 0, 2, N'Drama in English', N ↙
    'ENG', 1205, N'', CAST(0x077930F6239197340B AS DateTime2), N'', NULL, NULL, 0, NULL, ↙
    NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (17, N'ACU', 1, 8, 1, 1, N'SEM-72', N'CSS-135', 200, 0, 0, 3, N'Computer Applications'↙
    , N'GES', 2101, N'', CAST(0x0743AAB6799197340B AS DateTime2), N'', NULL, NULL, 0, NULL↙
    , NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (18, N'ACU', 1, 8, 1, 1, N'SEM-72', N'CSS-135', 200, 0, 0, 2, N'Introduction to ↙
    Philosophy & Logic', N'GES', 2102, N'', CAST(0x0718AC43E89197340B AS DateTime2), N'', ↙
    NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (19, N'ACU', 1, 8, 1, 1, N'SEM-72', N'CSS-136', 200, 0, 0, 0, N'Entrepreneurial ↙
    Education I', N'GES', 2103, N'', CAST(0x07CD8C220A9297340B AS DateTime2), N'', NULL, ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (20, N'ACU', 1, 8, 1, 1, N'SEM-72', N'CSS-135', 200, 0, 0, 3, N'Introduction to ↙
    General Phonetics and Phonology', N'ENG', 2101, N'', CAST(0x07EA10C5899297340B AS ↙
    DateTime2), N'', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (21, N'ACU', 1, 8, 1, 1, N'SEM-72', N'CSS-135', 200, 0, 0, 3, N'History of the English↙
     Language', N'ENG', 2102, N'', CAST(0x07ADAF88A09297340B AS DateTime2), N'', NULL, ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (22, N'ACU', 1, 8, 1, 1, N'SEM-72', N'CSS-135', 200, 0, 0, 3, N'The English Yesterday ↙
    and Today', N'ENG', 2103, N'', CAST(0x07B3BD30B79297340B AS DateTime2), N'', NULL, ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (23, N'ACU', 1, 8, 1, 1, N'SEM-72', N'CSS-137', 200, 0, 0, 3, N'Creative Writing', N ↙
    'ENG', 2104, N'', CAST(0x07066A1BCD9297340B AS DateTime2), N'', NULL, NULL, 0, NULL, ↙
    NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (24, N'ACU', 1, 8, 1, 1, N'SEM-72', N'CSS-135', 200, 0, 0, 3, N'Forms of Literature I:↙
     Poetry', N'ENG', 2105, N'', CAST(0x07E6BE06E99297340B AS DateTime2), N'', NULL, NULL,↙
     0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (25, N'ACU', 1, 8, 1, 1, N'SEM-72', N'CSS-137', 200, 0, 0, 2, N'Introduction to Oral ↙
    Literature', N'ENG', 2106, N'', CAST(0x07324489FE9297340B AS DateTime2), N'', NULL, ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (26, N'ACU', 1, 8, 1, 1, N'SEM-73', N'CSS-135', 200, 0, 0, 2, N'Nigerian Peoples and ↙
    Culture', N'GES', 2201, N'', CAST(0x0724A2CE2C9397340B AS DateTime2), N'', NULL, NULL,↙
     0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (27, N'ACU', 1, 8, 1, 1, N'SEM-73', N'CSS-135', 200, 0, 0, 2, N'Introduction to ↙
    History & Philosophy of Science', N'GES', 2202, N'', CAST(0x07AA13B35F9397340B AS ↙
    DateTime2), N'', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (28, N'ACU', 1, 8, 1, 1, N'SEM-73', N'CSS-136', 200, 0, 0, 0, N'Entrepreneurial ↙
    Education II', N'GES', 2203, N'', CAST(0x0791D33C7A9397340B AS DateTime2), N'', NULL, ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (29, N'ACU', 1, 8, 1, 1, N'SEM-73', N'CSS-135', 200, 0, 0, 3, N'Introductory English ↙
    Morphology and Syntax', N'ENG', 2201, N'', CAST(0x070586139B9397340B AS DateTime2), N ↙
    '', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (30, N'ACU', 1, 8, 1, 1, N'SEM-73', N'CSS-135', 200, 0, 0, 3, N'The African Novel', N ↙
    'ENG', 2202, N'', CAST(0x07758FA2B09397340B AS DateTime2), N'', NULL, NULL, 0, NULL, ↙
    NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],     ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (31, N'ACU', 1, 8, 1, 1, N'SEM-73', N'CSS-135', 200, 0, 0, 3, N'Language and Society',↙
     N'ENG', 2203, N'', CAST(0x074FA4F1BD9397340B AS DateTime2), N'', NULL, NULL, 0, NULL,↙
     NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],     ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (32, N'ACU', 1, 8, 1, 1, N'SEM-73', N'CSS-135', 200, 0, 0, 3, N'Forms of Literature II↙
    : Prose Fiction', N'ENG', 2204, N'', CAST(0x0702D032F89397340B AS DateTime2), N'',    ↙
    NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],     ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (33, N'ACU', 1, 8, 1, 1, N'SEM-73', N'CSS-135', 200, 0, 0, 3, N'Forms of Literature   ↙
    III: Drama', N'ENG', 2205, N'', CAST(0x07C5232D419497340B AS DateTime2), N'', NULL,   ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],     ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (34, N'ACU', 1, 8, 1, 1, N'SEM-72', N'CSS-135', 300, 0, 0, 3, N'Phonology of English',↙
     N'ENG', 3101, N'', CAST(0x079D83038D9497340B AS DateTime2), N'', NULL, NULL, 0, NULL,↙
     NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],     ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (35, N'ACU', 1, 8, 1, 1, N'SEM-72', N'CSS-137', 300, 0, 0, 3, N'Introduction to       ↙
    Applied Linguistics', N'ENG', 3102, N'', CAST(0x075DAE69A59497340B AS DateTime2), N'',↙
     NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],     ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (36, N'ACU', 1, 8, 1, 1, N'SEM-72', N'CSS-135', 300, 0, 0, 3, N'Discourse Analysis', N↙
    'ENG', 3103, N'', CAST(0x070DC96EB79497340B AS DateTime2), N'', NULL, NULL, 0, NULL,  ↙
    NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],     ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (37, N'ACU', 1, 8, 1, 1, N'SEM-72', N'CSS-135', 300, 0, 0, 3, N'Modern African Poetry'↙
    , N'ENG', 3104, N'', CAST(0x07DFD02BC69497340B AS DateTime2), N'', NULL, NULL, 0, NULL↙
    , NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],     ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (38, N'ACU', 1, 8, 1, 1, N'SEM-72', N'CSS-137', 300, 0, 0, 3, N'The English Language  ↙
    in Nigeria', N'ENG', 3105, N'', CAST(0x078334B5DA9497340B AS DateTime2), N'', NULL,   ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],     ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (39, N'ACU', 1, 8, 1, 1, N'SEM-72', N'CSS-135', 300, 0, 0, 3, N'Modern Africa Drama', ↙
    N'ENG', 3106, N'', CAST(0x07D46B66EC9497340B AS DateTime2), N'', NULL, NULL, 0, NULL, ↙
    NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],     ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (40, N'ACU', 1, 8, 1, 1, N'SEM-72', N'CSS-137', 300, 0, 0, 3, N'Contemporary English  ↙
    Usage', N'ENG', 3107, N'', CAST(0x073461E7069597340B AS DateTime2), N'', NULL, NULL, 0↙
    , NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (41, N'ACU', 1, 8, 1, 1, N'SEM-72', N'CSS-135', 300, 0, 0, 3, N'American Literature', ↙
    N'ENG', 3108, N'', CAST(0x07C1B2CD149597340B AS DateTime2), N'', NULL, NULL, 0, NULL, ↙
    NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (42, N'ACU', 1, 8, 1, 1, N'SEM-72', N'CSS-135', 300, 0, 0, 3, N'Advanced Basic Grammar↙
    ; the Group & the Sentence', N'ENG', 3109, N'', CAST(0x07EB8716369597340B AS ↙
    DateTime2), N'', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (43, N'ACU', 1, 8, 1, 1, N'SEM-73', N'CSS-135', 300, 0, 0, 3, N'Supra-Segmental ↙
    Phonology of English', N'ENG', 3201, N'', CAST(0x0762635C619597340B AS DateTime2), N''↙
    , NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (44, N'ACU', 1, 8, 1, 1, N'SEM-73', N'CSS-135', 300, 0, 0, 3, N'Introduction to ↙
    Semantics', N'ENG', 3202, N'', CAST(0x075238BD779597340B AS DateTime2), N'', NULL, ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (45, N'ACU', 1, 8, 1, 1, N'SEM-73', N'CSS-137', 300, 0, 0, 3, N'Language and Style', N↙
    'ENG', 3203, N'', CAST(0x076F3DA58B9597340B AS DateTime2), N'', NULL, NULL, 0, NULL, ↙
    NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (46, N'ACU', 1, 8, 1, 1, N'SEM-73', N'CSS-135', 300, 0, 0, 3, N'Research Methods in ↙
    English Usage', N'ENG', 3204, N'', CAST(0x07EC5271C39597340B AS DateTime2), N'', NULL,↙
     NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (47, N'ACU', 1, 8, 1, 1, N'SEM-73', N'CSS-137', 300, 0, 0, 3, N'Translation Theory', N↙
    'ENG', 3205, N'', CAST(0x079F0C10D69597340B AS DateTime2), N'', NULL, NULL, 0, NULL, ↙
    NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (48, N'ACU', 1, 8, 1, 1, N'SEM-73', N'CSS-135', 300, 0, 0, 3, N'Modern African Novel',↙
     N'ENG', 3206, N'', CAST(0x070325B0E69597340B AS DateTime2), N'', NULL, NULL, 0, NULL,↙
     NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (49, N'ACU', 1, 8, 1, 1, N'SEM-73', N'CSS-135', 300, 0, 0, 3, N'Literature and Gender'↙
    , N'ENG', 3207, N'', CAST(0x075CF0BDFF9597340B AS DateTime2), N'', NULL, NULL, 0, NULL↙
    , NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (50, N'ACU', 1, 8, 1, 1, N'SEM-73', N'CSS-137', 300, 0, 0, 3, N'African-American      ↙
    Literature', N'ENG', 3208, N'', CAST(0x071458DE199697340B AS DateTime2), N'', NULL,   ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (51, N'ACU', 1, 8, 1, 1, N'SEM-73', N'CSS-137', 300, 0, 0, 3, N'African-American      ↙
    Literature', N'ENG', 3208, N'', CAST(0x0742A8021D9697340B AS DateTime2), N'', NULL,   ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (52, N'ACU', 1, 8, 1, 1, N'SEM-72', N'CSS-135', 400, 0, 0, 3, N'English Literary      ↙
    Stylistics', N'ENG', 4101, N'', CAST(0x0785AFC0AA9697340B AS DateTime2), N'', NULL,   ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (53, N'ACU', 1, 8, 1, 1, N'SEM-72', N'CSS-135', 400, 0, 0, 3, N'Multilingualism', N   ↙
    'ENG', 4102, N'', CAST(0x074A672FBF9697340B AS DateTime2), N'', NULL, NULL, 0, NULL,  ↙
    NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (54, N'ACU', 1, 8, 1, 1, N'SEM-72', N'CSS-137', 400, 0, 0, 3, N'English for Specific  ↙
    Purposes', N'ENG', 4103, N'', CAST(0x07DC3B5BD29697340B AS DateTime2), N'', NULL, NULL↙
    , 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (55, N'ACU', 1, 8, 1, 1, N'SEM-72', N'CSS-137', 400, 0, 0, 3, N'Language and National ↙
    Development', N'ENG', 4104, N'', CAST(0x07DB44B6E79697340B AS DateTime2), N'', NULL,  ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (56, N'ACU', 1, 8, 1, 1, N'SEM-72', N'CSS-135', 400, 0, 0, 3, N'The English Novel', N ↙
    'ENG', 4105, N'', CAST(0x07C53B1FF89697340B AS DateTime2), N'', NULL, NULL, 0, NULL,  ↙
    NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (57, N'ACU', 1, 8, 1, 1, N'SEM-72', N'CSS-135', 400, 0, 0, 3, N'Research Methods II', ↙
    N'ENG', 4106, N'', CAST(0x075FAE6A099797340B AS DateTime2), N'', NULL, NULL, 0, NULL, ↙
    NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (58, N'ACU', 1, 8, 1, 1, N'SEM-72', N'CSS-135', 400, 0, 0, 3, N'Shakespeare and Other ↙
    Dramatists', N'ENG', 4107, N'', CAST(0x075A5728209797340B AS DateTime2), N'', NULL,   ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (59, N'ACU', 1, 8, 1, 1, N'SEM-72', N'CSS-137', 400, 0, 0, 3, N'Caribbean Literature',↙
     N'ENG', 4108, N'', CAST(0x077C114A319797340B AS DateTime2), N'', NULL, NULL, 0, NULL,↙
     NULL)
```

```
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (60, N'ACU', 1, 8, 1, 1, N'SEM-73', N'CSS-135', 400, 0, 0, 3, N'Literary History and  ↙
    Theory', N'ENG', 4201, N'', CAST(0x0715E1FF4E9797340B AS DateTime2), N'', NULL, NULL, ↙
    0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (61, N'ACU', 1, 8, 1, 1, N'SEM-73', N'CSS-137', 400, 0, 0, 3, N'Pragmatics', N'ENG',  ↙
    4202, N'', CAST(0x0722B46B5D9797340B AS DateTime2), N'', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (62, N'ACU', 1, 8, 1, 1, N'SEM-73', N'CSS-135', 400, 0, 0, 6, N'Project/Long Essay', N↙
    'ENG', 4203, N'', CAST(0x07B984126D9797340B AS DateTime2), N'', NULL, NULL, 0, NULL,  ↙
    NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (63, N'ACU', 1, 8, 1, 1, N'SEM-73', N'CSS-137', 400, 0, 0, 3, N'Advanced English      ↙
    Phonology', N'ENG', 4204, N'', CAST(0x072A99F1809797340B AS DateTime2), N'', NULL,    ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (64, N'ACU', 1, 8, 1, 1, N'SEM-73', N'CSS-137', 400, 0, 0, 3, N'Speech Writing', N'ENG↙
    ', 4205, N'', CAST(0x07FB0701B99797340B AS DateTime2), N'', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (65, N'ACU', 1, 8, 1, 1, N'SEM-73', N'CSS-137', 400, 0, 0, 3, N'Literature of other   ↙
    cultures', N'ENG', 4206, N'', CAST(0x0718320FCD9797340B AS DateTime2), N'', NULL, NULL↙
    , 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (66, N'ACU', 1, 8, 1, 1, N'SEM-73', N'CSS-135', 400, 0, 0, 3, N'Pre-Colonial          ↙
    Literature', N'ENG', 4207, N'', CAST(0x071904B3DD9797340B AS DateTime2), N'', NULL,   ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (67, N'ACU', 2, 14, 8, 17, N'SEM-72', N'CSS-135', 100, 1, 1, 2, N'Introduction to     ↙
    Computing', N'CSC', 1101, N'', CAST(0x0767B730039997340B AS DateTime2), N'', NULL,    ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (68, N'ACU', 2, 14, 8, 17, N'SEM-72', N'CSS-135', 100, 0, 0, 3, N'Discrete Structures'↙
    , N'CSC', 1102, N'', CAST(0x0758C3E4339997340B AS DateTime2), N'', NULL, NULL, 0, NULL↙
    , NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (69, N'ACU', 2, 14, 8, 17, N'SEM-72', N'CSS-136', 100, 0, 0, 3, N'Elementary          ↙
    Mathematics I', N'MTH', 1101, N'', CAST(0x076F559A4D9997340B AS DateTime2), N'', NULL,↙
```

```
     NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (70, N'ACU', 2, 14, 8, 17, N'SEM-72', N'CSS-136', 100, 0, 0, 3, N'General Physics I', ↙
    N'PHY', 1101, N'', CAST(0x07D4D06B6E9997340B AS DateTime2), N'', NULL, NULL, 0, NULL, ↙
    NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (71, N'ACU', 2, 14, 8, 17, N'SEM-72', N'CSS-136', 100, 0, 0, 1, N'Experimental Physics↙
     IA', N'PHY', 1107, N'', CAST(0x0791BBD18D9997340B AS DateTime2), N'', NULL, NULL, 0, ↙
    NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (72, N'ACU', 2, 14, 8, 17, N'SEM-72', N'CSS-136', 100, 0, 0, 3, N'General Physical    ↙
    Chemistry', N'CHM', 1103, N'', CAST(0x071694BE9D9997340B AS DateTime2), N'', NULL,    ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (73, N'ACU', 2, 14, 8, 17, N'SEM-72', N'CSS-136', 100, 0, 0, 1, N'Practical Chemistry ↙
    I', N'CHM', 1105, N'', CAST(0x077E0CFCAC9997340B AS DateTime2), N'', NULL, NULL, 0,   ↙
    NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (74, N'ACU', 2, 14, 8, 17, N'SEM-72', N'CSS-136', 100, 0, 0, 3, N'Introduction to     ↙
    Statistics', N'STA', 1101, N'', CAST(0x07742F69BD9997340B AS DateTime2), N'', NULL,   ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (75, N'ACU', 2, 14, 8, 17, N'SEM-72', N'CSS-135', 100, 0, 0, 2, N'Effective           ↙
    Communication Skills I', N'GES', 1101, N'', CAST(0x073ABCD02F9A97340B AS DateTime2), N↙
    '', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (76, N'ACU', 2, 14, 8, 17, N'SEM-72', N'CSS-135', 100, 0, 0, 1, N'Use of Library', N  ↙
    'GES', 1102, N'', CAST(0x075C04753F9A97340B AS DateTime2), N'', NULL, NULL, 0, NULL,  ↙
    NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (77, N'ACU', 2, 14, 8, 17, N'SEM-72', N'CSS-136', 100, 0, 0, 0, N'Use of French I', N ↙
    'GES', 1103, N'', CAST(0x07B7BDF1539A97340B AS DateTime2), N'', NULL, NULL, 0, NULL,  ↙
    NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (78, N'ACU', 2, 14, 8, 17, N'SEM-73', N'CSS-135', 100, 0, 0, 2, N'Computer Programming↙
     Concepts', N'CSC', 1201, N'', CAST(0x07AA2A34759A97340B AS DateTime2), N'', NULL,    ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (79, N'ACU', 2, 14, 8, 17, N'SEM-73', N'CSS-135', 100, 0, 0, 2, N'Computer Workshop  ↙
    Practice I', N'ICT', 1207, N'', CAST(0x077EAA35899A97340B AS DateTime2), N'', NULL,  ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (80, N'ACU', 2, 14, 8, 17, N'SEM-73', N'CSS-136', 100, 0, 0, 3, N'Elementary         ↙
    Mathematics II', N'MTH', 1202, N'', CAST(0x07BDE6219A9A97340B AS DateTime2), N'', NULL↙
    , NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (81, N'ACU', 2, 14, 8, 17, N'SEM-73', N'CSS-136', 100, 0, 0, 3, N'Elementary         ↙
    Mathematics III', N'MTH', 1203, N'', CAST(0x074F3483A49A97340B AS DateTime2), N'',    ↙
    NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (82, N'ACU', 2, 14, 8, 17, N'SEM-73', N'CSS-136', 100, 0, 0, 3, N'General Physics II',↙
     N'PHY', 1202, N'', CAST(0x0749704CB49A97340B AS DateTime2), N'', NULL, NULL, 0, NULL,↙
     NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (83, N'ACU', 2, 14, 8, 17, N'SEM-73', N'CSS-136', 100, 0, 0, 1, N'Experimental Physics↙
     IB', N'PHY', 1208, N'', CAST(0x07D9F464C39A97340B AS DateTime2), N'', NULL, NULL, 0, ↙
    NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (84, N'ACU', 2, 14, 8, 17, N'SEM-73', N'CSS-136', 100, 0, 0, 2, N'Introduction to    ↙
    Electronics', N'PHY', 1204, N'', CAST(0x07A339DCD39A97340B AS DateTime2), N'', NULL,  ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (85, N'ACU', 2, 14, 8, 17, N'SEM-73', N'CSS-136', 100, 0, 0, 3, N'Probability I', N  ↙
    'STA', 1201, N'', CAST(0x07CF72BEE69A97340B AS DateTime2), N'', NULL, NULL, 0, NULL,  ↙
    NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (86, N'ACU', 2, 14, 8, 17, N'SEM-73', N'CSS-135', 100, 0, 0, 2, N'Effective          ↙
    Communication Skills II', N'GES', 1201, N'', CAST(0x07E52E23FB9A97340B AS DateTime2), ↙
    N'', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (87, N'ACU', 2, 14, 8, 17, N'SEM-73', N'CSS-136', 100, 0, 0, 0, N'Use of French II', N↙
    'GES', 1202, N'', CAST(0x07B0A005099B97340B AS DateTime2), N'', NULL, NULL, 0, NULL,  ↙
    NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (88, N'ACU', 2, 14, 8, 17, N'SEM-72', N'CSS-135', 200, 0, 0, 3, N'Computer Networks I'↙
    , N'ICT', 2101, N'', CAST(0x0741410E299B97340B AS DateTime2), N'', NULL, NULL, 0, NULL↙
    , NULL)
```

```
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],       ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (89, N'ACU', 2, 14, 8, 17, N'SEM-72', N'CSS-135', 200, 0, 0, 3, N'Digital Computer  ↙
    Logic I', N'ICT', 2102, N'', CAST(0x075B35DF3A9B97340B AS DateTime2), N'', NULL, NULL,↙
     0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],       ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (90, N'ACU', 2, 14, 8, 17, N'SEM-72', N'CSS-135', 200, 0, 0, 3, N'Introduction to   ↙
    Computer Architecture', N'ICT', 2103, N'', CAST(0x07005A6C4D9B97340B AS DateTime2), N ↙
    '', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],       ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (91, N'ACU', 2, 14, 8, 17, N'SEM-72', N'CSS-135', 200, 0, 0, 2, N'Structured        ↙
    Programming I', N'CSC', 2101, N'', CAST(0x079A3B2A5F9B97340B AS DateTime2), N'', NULL,↙
     NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],       ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (92, N'ACU', 2, 14, 8, 17, N'SEM-72', N'CSS-135', 200, 0, 0, 3, N'Data Structures and ↙
    Algorithms', N'CSC', 2102, N'', CAST(0x07CC02A2759B97340B AS DateTime2), N'', NULL,  ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],       ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (93, N'ACU', 2, 14, 8, 17, N'SEM-72', N'CSS-136', 200, 0, 0, 3, N'Linear Algebra', N ↙
    'MTH', 2102, N'', CAST(0x076AD895869B97340B AS DateTime2), N'', NULL, NULL, 0, NULL, ↙
    NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],       ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (94, N'ACU', 2, 14, 8, 17, N'SEM-72', N'CSS-136', 200, 0, 0, 3, N'Electronics', N'PHY'↙
    , 2103, N'', CAST(0x0741261A929B97340B AS DateTime2), N'', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],       ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (95, N'ACU', 2, 14, 8, 17, N'SEM-72', N'CSS-135', 200, 0, 0, 2, N'Introduction to   ↙
    Logic and Philosophy', N'GES', 2102, N'', CAST(0x07E4CBAAA99B97340B AS DateTime2), N''↙
    , NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],       ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (96, N'ACU', 2, 14, 8, 17, N'SEM-72', N'CSS-136', 200, 0, 0, 0, N'Entrepreneurial   ↙
    Education I', N'GES', 2103, N'', CAST(0x074B83E4BA9B97340B AS DateTime2), N'', NULL, ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],       ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (97, N'ACU', 2, 14, 8, 17, N'SEM-73', N'CSS-135', 200, 0, 0, 3, N'Computer Networks II↙
    ', N'ICT', 2201, N'', CAST(0x073B1BECEE9B97340B AS DateTime2), N'', NULL, NULL, 0,  ↙
    NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],       ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (98, N'ACU', 2, 14, 8, 17, N'SEM-73', N'CSS-135', 200, 0, 0, 3, N'Digital Computer  ↙
    Logic II', N'ICT', 2202, N'', CAST(0x0731F40CFF9B97340B AS DateTime2), N'', NULL, NULL↙
    , 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],   ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],          ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES   ↙
    (99, N'ACU', 2, 14, 8, 17, N'SEM-73', N'CSS-135', 200, 0, 0, 2, N'Computer Workshop   ↙
    Practice II', N'ICT', 2207, N'', CAST(0x07D7D6920E9C97340B AS DateTime2), N'', NULL,   ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],   ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],          ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES   ↙
    (100, N'ACU', 2, 14, 8, 17, N'SEM-73', N'CSS-135', 200, 0, 0, 3, N'Principles of      ↙
    Compilers', N'CSC', 2201, N'', CAST(0x0783FD61219C97340B AS DateTime2), N'', NULL,     ↙
    NULL, 0, NULL, NULL)
GO
print 'Processed 100 total records'
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],   ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],          ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES   ↙
    (101, N'ACU', 2, 14, 8, 17, N'SEM-73', N'CSS-135', 200, 0, 0, 3, N'Operating Systems  ↙
    Principles', N'CSC', 2202, N'', CAST(0x07BAC054349C97340B AS DateTime2), N'', NULL,    ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],   ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],          ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES   ↙
    (102, N'ACU', 2, 14, 8, 17, N'SEM-73', N'CSS-135', 200, 0, 0, 2, N'Low Level Languages↙
    ', N'CSC', 2203, N'', CAST(0x07A7E94D529C97340B AS DateTime2), N'', NULL, NULL, 0,     ↙
    NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],   ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],          ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES   ↙
    (103, N'ACU', 2, 14, 8, 17, N'SEM-73', N'CSS-135', 200, 0, 0, 3, N'Object Oriented    ↙
    Programming', N'CSC', 2206, N'', CAST(0x07BDA2A7659C97340B AS DateTime2), N'', NULL,   ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],   ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],          ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES   ↙
    (104, N'ACU', 2, 14, 8, 17, N'SEM-73', N'CSS-136', 200, 0, 0, 0, N'Entrepreneurial    ↙
    Education I', N'GES', 2203, N'', CAST(0x07166EB57E9C97340B AS DateTime2), N'', NULL,   ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],   ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],          ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES   ↙
    (105, N'ACU', 2, 14, 8, 17, N'SEM-73', N'CSS-135', 200, 0, 0, 3, N'Nigerian Peoples   ↙
    and Culture', N'GES', 2201, N'', CAST(0x0796658A9B9C97340B AS DateTime2), N'', NULL,   ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],   ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],          ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES   ↙
    (106, N'ACU', 2, 14, 8, 17, N'SEM-73', N'CSS-135', 200, 0, 0, 2, N'Introduction to    ↙
    Social Science', N'GES', 2204, N'', CAST(0x0766F155B09C97340B AS DateTime2), N'', NULL↙
    , NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],   ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],          ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES   ↙
    (107, N'ACU', 2, 14, 8, 17, N'SEM-72', N'CSS-135', 300, 0, 0, 3, N'Introduction to    ↙
    Linux Operating System', N'ICT', 3101, N'', CAST(0x073B6516D49C97340B AS DateTime2), N↙
    '', NULL, NULL, 0, NULL, NULL)
```

```
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (108, N'ACU', 2, 14, 8, 17, N'SEM-72', N'CSS-135', 300, 0, 0, 3, N'Introduction to ↙
    Microsoft SQL-2000', N'ICT', 3102, N'', CAST(0x0781A344E59C97340B AS DateTime2), N'', ↙
    NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (109, N'ACU', 2, 14, 8, 17, N'SEM-72', N'CSS-135', 300, 0, 0, 3, N'Web Page Design and↙
     Management', N'ICT', 3103, N'', CAST(0x077B1F97FE9C97340B AS DateTime2), N'', NULL, ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (110, N'ACU', 2, 14, 8, 17, N'SEM-72', N'CSS-135', 300, 0, 0, 3, N'Systems Programming↙
     Principles', N'ICT', 3106, N'', CAST(0x07B660951A9D97340B AS DateTime2), N'', NULL, ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (111, N'ACU', 2, 14, 8, 17, N'SEM-72', N'CSS-135', 300, 0, 0, 2, N'Computer Workshop ↙
    Practice III', N'ICT', 3107, N'', CAST(0x07D054B0789D97340B AS DateTime2), N'', NULL, ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (112, N'ACU', 2, 14, 8, 17, N'SEM-72', N'CSS-135', 300, 0, 0, 3, N'Java Programming ↙
    Language', N'CSC', 3101, N'', CAST(0x07A8FBA29D9D97340B AS DateTime2), N'', NULL, NULL↙
    , 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (113, N'ACU', 2, 14, 8, 17, N'SEM-72', N'CSS-135', 300, 0, 0, 3, N'Microprocessor ↙
    Architecture', N'ICT', 3109, N'', CAST(0x072B3378B39D97340B AS DateTime2), N'', NULL, ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (114, N'ACU', 2, 14, 8, 17, N'SEM-72', N'CSS-135', 300, 0, 0, 2, N'Microprocessor ↙
    Laboratory', N'ICT', 3121, N'', CAST(0x07C749E6BF9D97340B AS DateTime2), N'', NULL, ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (115, N'ACU', 2, 14, 8, 17, N'SEM-73', N'CSS-135', 300, 0, 0, 6, N'Student Industrial ↙
    Work Experience', N'ICT', 3299, N'', CAST(0x07C0295BDB9D97340B AS DateTime2), N'', ↙
    NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (116, N'ACU', 2, 14, 8, 17, N'SEM-72', N'CSS-135', 400, 0, 0, 3, N'Computer Graphics',↙
     N'ICT', 4101, N'', CAST(0x078BAFE8FC9D97340B AS DateTime2), N'', NULL, NULL, 0, NULL,↙
     NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode], ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical], ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn], ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES ↙
    (117, N'ACU', 2, 14, 8, 17, N'SEM-72', N'CSS-135', 400, 0, 0, 3, N'Web Programming', N↙
    'ICT', 4102, N'', CAST(0x07A4CA5D089E97340B AS DateTime2), N'', NULL, NULL, 0, NULL,  ↙
    NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (118, N'ACU', 2, 14, 8, 17, N'SEM-72', N'CSS-135', 400, 0, 0, 2, N'Topics in ICT', N  ↙
    'ICT', 4107, N'', CAST(0x07DEEF6B139E97340B AS DateTime2), N'', NULL, NULL, 0, NULL,  ↙
    NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (119, N'ACU', 2, 14, 8, 17, N'SEM-72', N'CSS-135', 400, 0, 0, 3, N'Data Communications↙
    ', N'ICT', 4108, N'', CAST(0x075A6935279E97340B AS DateTime2), N'', NULL, NULL, 0,    ↙
    NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (120, N'ACU', 2, 14, 8, 17, N'SEM-72', N'CSS-135', 400, 0, 0, 2, N'Individual Research↙
     Seminar', N'ICT', 4199, N'', CAST(0x0701B9AF3D9E97340B AS DateTime2), N'', NULL, NULL↙
    , 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (121, N'ACU', 2, 14, 8, 17, N'SEM-72', N'CSS-135', 400, 0, 0, 3, N'Modelling and      ↙
    Simulation', N'CSC', 4105, N'', CAST(0x079368B5509E97340B AS DateTime2), N'', NULL,   ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (122, N'ACU', 2, 14, 8, 17, N'SEM-73', N'CSS-135', 400, 0, 0, 3, N'Principles of      ↙
    Software Engineering', N'ICT', 4201, N'', CAST(0x073B4DE9079F97340B AS DateTime2), N''↙
    , NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (123, N'ACU', 2, 14, 8, 17, N'SEM-73', N'CSS-135', 400, 0, 0, 3, N'ICT Policy and Data↙
     Security', N'ICT', 4205, N'', CAST(0x07D340381E9F97340B AS DateTime2), N'', NULL,    ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (124, N'ACU', 2, 14, 8, 17, N'SEM-73', N'CSS-135', 400, 0, 0, 2, N'Computer Workshop  ↙
    Practice IV', N'ICT', 4207, N'', CAST(0x076A036F429F97340B AS DateTime2), N'', NULL,  ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (125, N'ACU', 2, 14, 8, 17, N'SEM-73', N'CSS-135', 400, 0, 0, 6, N'Research Project', ↙
    N'ICT', 4299, N'', CAST(0x072E952A4F9F97340B AS DateTime2), N'', NULL, NULL, 0, NULL, ↙
    NULL)
INSERT [SetUp].[SubCourses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [CourseCode], [ProgramCode], [SemesterCode], [StatusCode], [LevelCode], [Practical],  ↙
    [Laboratory], [Credit], [Title], [PrefixCode], [SubNo], [Notes], [CreatedOn],         ↙
    [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES  ↙
    (126, N'ACU', 2, 14, 8, 17, N'SEM-73', N'CSS-135', 400, 0, 0, 3, N'Automata Theory,   ↙
    Compatibility and Formal Languages', N'CSC', 4201, N'', CAST(0x07551CDE659F97340B AS  ↙
    DateTime2), N'', NULL, NULL, 0, NULL, NULL)
SET IDENTITY_INSERT [SetUp].[SubCourses] OFF
```

```
/****** Object:  StoredProcedure [SetUp].[SPSubCoursesSelect]    Script Date: 08/16/2011  ↙
    01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from SetUp.SubCourses Table

CREATE PROC [SetUp].[SPSubCoursesSelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @FacultyCode BIGINT=NULL,
    @DepartmentCode BIGINT=NULL,
    @CourseCode BIGINT=NULL,
    @ProgramCode BIGINT=NULL,
    @SemesterCode NVARCHAR(50)=NULL,
    @LevelCode BIGINT=NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS
        (
        SELECT * FROM [SetUp].[SubCourses]
        WHERE ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([FacultyCode] = @FacultyCode) AND
                ([DepartmentCode]=@DepartmentCode) AND
                ([CourseCode] =@CourseCode) AND
                ([ProgramCode]=@ProgramCode) AND
                ([SemesterCode] =@SemesterCode) AND
                ([LevelCode]=@LevelCode) AND
                ([Deleted] = @Deleted)
               )
            BEGIN
            SELECT * FROM [SetUp].[SubCourses]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([FacultyCode] = @FacultyCode) AND
                ([DepartmentCode]=@DepartmentCode) AND
                ([CourseCode] =@CourseCode) AND
                ([ProgramCode]=@ProgramCode) AND
                ([SemesterCode] =@SemesterCode) AND
                ([LevelCode]=@LevelCode) AND
                ([Deleted] = @Deleted)
               )
            END
        ELSE
            BEGIN
            SELECT * FROM [SetUp].[SubCourses]
            WHERE
                (
                ([UniversityCode] = @UniversityCode) AND
                ([FacultyCode] = @FacultyCode) AND
                ([DepartmentCode]=@DepartmentCode) AND
                ([CourseCode] =@CourseCode) AND
                ([ProgramCode]=@ProgramCode) AND
                ([SemesterCode] =@SemesterCode) AND
                ([LevelCode]=@LevelCode) AND
                ([Deleted] = @Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPSubCoursesSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPSubCoursesInsertUpdate]    Script Date: 08/16/↙
    2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author             :   Ademola Adebo
--Reviewer           :   Godwin Mathias
--Date Created       :   07/08/2011
--Last Updated       :   07/08/2011
--Last Updated By    :   Ademola Adebo
--Description        :   Stored procedure for saving values into SetUp.SubCourses Table

CREATE PROC [SetUp].[SPSubCoursesInsertUpdate]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @FacultyCode BIGINT=NULL,
    @DepartmentCode BIGINT=NULL,
    @CourseCode BIGINT=NULL,
    @ProgramCode BIGINT=NULL,
    @SemesterCode NVARCHAR(50)=NULL,
    @StatusCode NVARCHAR(50)=NULL,
    @LevelCode BIGINT=NULL,
    @Practical INT=NULL,
    @Laboratory INT=NULL,
    @Credit INT=NULL,
    @Title NVARCHAR(256)=NULL,
    @PrefixCode NVARCHAR(50)=NULL,
    @SubNo INT=NULL,
    @Notes NVARCHAR(500)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[SubCourses]  WHERE ([Code] = @Code) AND (    ↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
                INSERT INTO [SetUp].[SubCourses]
                (
                    [UniversityCode],
                    [FacultyCode],
                    [DepartmentCode],
                    [CourseCode],
                    [ProgramCode],
                    [SemesterCode],
                    [StatusCode],
                    [LevelCode],
                    [Practical],
                    [Laboratory],
                    [Credit],
                    [Title],
                    [PrefixCode],
                    [SubNo],
                    [Notes],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]

                )
                VALUES
                (
                    @UniversityCode,
                    @FacultyCode,
                    @DepartmentCode,
                    @CourseCode,
                    @ProgramCode,
                    @SemesterCode,
                    @StatusCode,
                    @LevelCode,
                    @Practical,
                    @Laboratory,
                    @Credit,
                    @Title,
                    @PrefixCode,
                    @SubNo,
                    @Notes,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [SetUp].[SubCourses]
                SET
                    [UniversityCode]=@UniversityCode,
                    [FacultyCode]=@FacultyCode,
                    [DepartmentCode]=@DepartmentCode,
                    [CourseCode]=@CourseCode,
                    [ProgramCode]=@ProgramCode,
                    [SemesterCode]=@SemesterCode,
                    [StatusCode]=@StatusCode,
                    [LevelCode]=@LevelCode,
                    [Practical]=@Practical,
                    [Laboratory]=@Laboratory,
                    [Credit]=@Credit,
                    [Title]=@Title,
                    [PrefixCode]=@PrefixCode,
                    [SubNo]=@SubNo,
                    [Notes]=@Notes,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPSubCoursesInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPSubCoursesDeletePermanently]    Script Date:  ↙
    08/16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/12/2011
--Last Updated      :   07/12/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from SetUp.SubCourses Table

CREATE PROC [SetUp].[SPSubCoursesDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[SubCourses] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [SetUp].[SubCourses]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPSubCoursesDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPSubCoursesDelete]    Script Date: 08/16/2011  ↙
    01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/28/2011
--Last Updated        :   07/28/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for deleting values temporarily from SetUp.    ↙
    SubCourses Table

CREATE PROC [SetUp].[SPSubCoursesDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[SubCourses] WHERE (([Code] = @Code) AND     ↙
    (Deleted=@Deleted)))
            BEGIN
            UPDATE [SetUp].[SubCourses]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
```

```sql
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPSubCoursesDelete] TO PUBLIC
GO
/****** Object:  Table [SetUp].[Faculties]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [SetUp].[Faculties](
    [Code] [bigint] IDENTITY(1,1) NOT NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [Acronym] [nvarchar](50) NULL,
    [Description] [nvarchar](500) NULL,
    [Notes] [nvarchar](1000) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Faculties] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,          ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
SET IDENTITY_INSERT [SetUp].[Faculties] ON
INSERT [SetUp].[Faculties] ([Code], [UniversityCode], [Acronym], [Description], [Notes],  ↙
    [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn],         ↙
    [DeletedBy]) VALUES (1, N'ACU', N'FH', N'Humanities', N'hjj', CAST                     ↙
    (0x0700000000007B340B AS DateTime2), N'ademola', CAST(0x07D92B181EA195340B AS          ↙
    DateTime2), N'', 0, NULL, NULL)
INSERT [SetUp].[Faculties] ([Code], [UniversityCode], [Acronym], [Description], [Notes],  ↙
    [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn],         ↙
    [DeletedBy]) VALUES (2, N'ACU', N'FNS', N'Natural Sciences', N'By Godwin Mathias',     ↙
    CAST(0x0700000000007B340B AS DateTime2), N'ademola', CAST(0x075A51FA2BA195340B AS      ↙
    DateTime2), N'', 0, NULL, NULL)
INSERT [SetUp].[Faculties] ([Code], [UniversityCode], [Acronym], [Description], [Notes],  ↙
    [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn],         ↙
    [DeletedBy]) VALUES (3, N'ACU', N'FL', N'Law', NULL, CAST(0x0700000000007B340B AS      ↙
    DateTime2), N'ademola', NULL, NULL, 0, NULL, NULL)
```

```
INSERT [SetUp].[Faculties] ([Code], [UniversityCode], [Acronym], [Description], [Notes], ↙
    [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], ↙
    [DeletedBy]) VALUES (4, N'ACU', N'FSMS', N'Social and Management Sciences', NULL, CAST↙
    (0x0700000000007B340B AS DateTime2), N'ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Faculties] ([Code], [UniversityCode], [Acronym], [Description], [Notes], ↙
    [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], ↙
    [DeletedBy]) VALUES (6, N'ACU', N'FES', N'Environmental Studies', NULL, CAST ↙
    (0x0700000000007B340B AS DateTime2), N'ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Faculties] ([Code], [UniversityCode], [Acronym], [Description], [Notes], ↙
    [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], ↙
    [DeletedBy]) VALUES (7, N'ACU', N'FHS', N'Health Sciences', NULL, CAST ↙
    (0x0700000000007B340B AS DateTime2), N'ademola', NULL, NULL, 0, NULL, NULL)
SET IDENTITY_INSERT [SetUp].[Faculties] OFF
/****** Object:  Table [SetUp].[Departments]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [SetUp].[Departments](
    [Code] [bigint] IDENTITY(1,1) NOT NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [FacultyCode] [bigint] NULL,
    [Acronym] [nvarchar](50) NULL,
    [Description] [nvarchar](500) NULL,
    [Notes] [nvarchar](1000) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Departments] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF, ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
SET IDENTITY_INSERT [SetUp].[Departments] ON
INSERT [SetUp].[Departments] ([Code], [UniversityCode], [FacultyCode], [Acronym], ↙
    [Description], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], ↙
    [Deleted], [DeletedOn], [DeletedBy]) VALUES (8, N'ACU', 1, N'DENG', N'English', NULL, ↙
    CAST(0x0700000000007B340B AS DateTime2), N'ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Departments] ([Code], [UniversityCode], [FacultyCode], [Acronym], ↙
    [Description], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], ↙
    [Deleted], [DeletedOn], [DeletedBy]) VALUES (9, N'ACU', 1, N'DHIS', N'History and ↙
    International Studies', NULL, CAST(0x0700000000007B340B AS DateTime2), N'ademola', ↙
    NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Departments] ([Code], [UniversityCode], [FacultyCode], [Acronym], ↙
    [Description], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], ↙
    [Deleted], [DeletedOn], [DeletedBy]) VALUES (10, N'ACU', 1, N'DREL', N'Religions', ↙
    NULL, CAST(0x0700000000007B340B AS DateTime2), N'ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Departments] ([Code], [UniversityCode], [FacultyCode], [Acronym], ↙
    [Description], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], ↙
    [Deleted], [DeletedOn], [DeletedBy]) VALUES (11, N'ACU', 2, N'DBS', N'Biological ↙
    Sciences', NULL, CAST(0x0700000000007B340B AS DateTime2), N'ademola', NULL, NULL, 0, ↙
    NULL, NULL)
INSERT [SetUp].[Departments] ([Code], [UniversityCode], [FacultyCode], [Acronym], ↙
    [Description], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], ↙
    [Deleted], [DeletedOn], [DeletedBy]) VALUES (12, N'ACU', 2, N'DCS', N'Chemical ↙
    Sciences', NULL, CAST(0x0700000000007B340B AS DateTime2), N'ademola', NULL, NULL, 0, ↙
    NULL, NULL)
INSERT [SetUp].[Departments] ([Code], [UniversityCode], [FacultyCode], [Acronym], ↙
    [Description], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], ↙
    [Deleted], [DeletedOn], [DeletedBy]) VALUES (13, N'ACU', 2, N'DES', N'Earth Sciences',↙
     NULL, CAST(0x0700000000007B340B AS DateTime2), N'ademola', NULL, NULL, 0, NULL, NULL)
```

```
INSERT [SetUp].[Departments] ([Code], [UniversityCode], [FacultyCode], [Acronym], ↙
    [Description], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], ↙
    [Deleted], [DeletedOn], [DeletedBy]) VALUES (14, N'ACU', 2, N'DPS', N'Physical ↙
    Sciences', NULL, CAST(0x0700000000007B340B AS DateTime2), N'ademola', NULL, NULL, 0, ↙
    NULL, NULL)
INSERT [SetUp].[Departments] ([Code], [UniversityCode], [FacultyCode], [Acronym], ↙
    [Description], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], ↙
    [Deleted], [DeletedOn], [DeletedBy]) VALUES (15, N'ACU', 4, N'DACT', N'Accounting', ↙
    NULL, CAST(0x0700000000007B340B AS DateTime2), N'ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Departments] ([Code], [UniversityCode], [FacultyCode], [Acronym], ↙
    [Description], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], ↙
    [Deleted], [DeletedOn], [DeletedBy]) VALUES (16, N'ACU', 4, N'DBA', N'Business ↙
    Administration', NULL, CAST(0x0700000000007B340B AS DateTime2), N'ademola', NULL, NULL↙
    , 0, NULL, NULL)
INSERT [SetUp].[Departments] ([Code], [UniversityCode], [FacultyCode], [Acronym], ↙
    [Description], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], ↙
    [Deleted], [DeletedOn], [DeletedBy]) VALUES (17, N'ACU', 4, N'DECNS', N'Economics', ↙
    NULL, CAST(0x0700000000007B340B AS DateTime2), N'ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Departments] ([Code], [UniversityCode], [FacultyCode], [Acronym], ↙
    [Description], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], ↙
    [Deleted], [DeletedOn], [DeletedBy]) VALUES (18, N'ACU', 4, N'DCMS', N'Communication ↙
    and Media Studies', NULL, CAST(0x0700000000007B340B AS DateTime2), N'ademola', NULL, ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[Departments] ([Code], [UniversityCode], [FacultyCode], [Acronym], ↙
    [Description], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], ↙
    [Deleted], [DeletedOn], [DeletedBy]) VALUES (19, N'ACU', 3, N'DLAW', N'Law', NULL, ↙
    CAST(0x0700000000007B340B AS DateTime2), N'ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Departments] ([Code], [UniversityCode], [FacultyCode], [Acronym], ↙
    [Description], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], ↙
    [Deleted], [DeletedOn], [DeletedBy]) VALUES (20, N'ACU', 6, N'DARCH', N'Architecture',↙
     NULL, CAST(0x0700000000007B340B AS DateTime2), N'ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Departments] ([Code], [UniversityCode], [FacultyCode], [Acronym], ↙
    [Description], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], ↙
    [Deleted], [DeletedOn], [DeletedBy]) VALUES (21, N'ACU', 6, N'DEM', N'Estate ↙
    Management', NULL, CAST(0x0700000000007B340B AS DateTime2), N'ademola', NULL, NULL, 0,↙
     NULL, NULL)
INSERT [SetUp].[Departments] ([Code], [UniversityCode], [FacultyCode], [Acronym], ↙
    [Description], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], ↙
    [Deleted], [DeletedOn], [DeletedBy]) VALUES (22, N'ACU', 6, N'DURP', N'Urban and ↙
    Regional Planning', NULL, CAST(0x0700000000007B340B AS DateTime2), N'ademola', NULL, ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[Departments] ([Code], [UniversityCode], [FacultyCode], [Acronym], ↙
    [Description], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], ↙
    [Deleted], [DeletedOn], [DeletedBy]) VALUES (23, N'ACU', 6, N'DQS', N'Quantity ↙
    Surveying', NULL, CAST(0x0700000000007B340B AS DateTime2), N'ademola', NULL, NULL, 0, ↙
    NULL, NULL)
INSERT [SetUp].[Departments] ([Code], [UniversityCode], [FacultyCode], [Acronym], ↙
    [Description], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], ↙
    [Deleted], [DeletedOn], [DeletedBy]) VALUES (24, N'ACU', 7, N'DMBBS', N'Medicine and ↙
    Surgery', NULL, CAST(0x0700000000007B340B AS DateTime2), N'ademola', NULL, NULL, 0, ↙
    NULL, NULL)
INSERT [SetUp].[Departments] ([Code], [UniversityCode], [FacultyCode], [Acronym], ↙
    [Description], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], ↙
    [Deleted], [DeletedOn], [DeletedBy]) VALUES (25, N'ACU', 7, N'DPHARM', N'Pharmacy', ↙
    NULL, CAST(0x0700000000007B340B AS DateTime2), N'ademola', NULL, NULL, 0, NULL, NULL)
SET IDENTITY_INSERT [SetUp].[Departments] OFF
/****** Object:  Table [SetUp].[Courses]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [SetUp].[Courses](
    [Code] [bigint] IDENTITY(1,1) NOT NULL,
    [UniversityCode] [nvarchar](50) NOT NULL,
    [FacultyCode] [bigint] NOT NULL,
    [DepartmentCode] [bigint] NULL,
    [Acronym] [nvarchar](50) NULL,
    [Description] [nvarchar](300) NULL,
    [Notes] [nvarchar](1000) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Courses] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,       ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
SET IDENTITY_INSERT [SetUp].[Courses] ON
INSERT [SetUp].[Courses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [Acronym], [Description], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn],         ↙
    [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES (1, N'ACU', 1, 8, N'ENG', N ↙
    'English', N'', CAST(0x0700000000007B340B AS DateTime2), N'ademola', NULL, NULL, 0,  ↙
    NULL, NULL)
INSERT [SetUp].[Courses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [Acronym], [Description], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn],         ↙
    [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES (2, N'ACU', 1, 8, N'HIS', N ↙
    'History and International Studies', NULL, CAST(0x0700000000007B340B AS DateTime2), N ↙
    'ademola', NULL, NULL, 1, CAST(0x078029DB557596340B AS DateTime2), N'')
INSERT [SetUp].[Courses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [Acronym], [Description], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn],         ↙
    [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES (3, N'ACU', 2, 11, N'MCB', N↙
    'Microbiology', NULL, CAST(0x0700000000007B340B AS DateTime2), N'ademola', NULL, NULL,↙
     0, NULL, NULL)
INSERT [SetUp].[Courses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [Acronym], [Description], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn],         ↙
    [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES (4, N'ACU', 2, 12, N'BCH', N↙
    'Biochemistry', NULL, CAST(0x0700000000007B340B AS DateTime2), N'ademola', NULL, NULL,↙
     0, NULL, NULL)
INSERT [SetUp].[Courses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [Acronym], [Description], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn],         ↙
    [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES (5, N'ACU', 2, 12, N'ICM', N↙
    'Industrial Chemistry', NULL, CAST(0x0700000000007B340B AS DateTime2), N'ademola',   ↙
    NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Courses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [Acronym], [Description], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn],         ↙
    [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES (6, N'ACU', 2, 13, N'GEY', N↙
    'Geology', NULL, CAST(0x0700000000007B340B AS DateTime2), N'ademola', NULL, NULL, 0,  ↙
    NULL, NULL)
INSERT [SetUp].[Courses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [Acronym], [Description], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn],         ↙
    [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES (7, N'ACU', 2, 14, N'CSC', N↙
    'Computer Sciences', NULL, CAST(0x0700000000007B340B AS DateTime2), N'ademola', NULL, ↙
    NULL, 0, NULL, NULL)
INSERT [SetUp].[Courses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [Acronym], [Description], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn],         ↙
    [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES (8, N'ACU', 2, 14, N'ICT', N↙
    'Information and Communication Technology', NULL, CAST(0x0700000000007B340B AS       ↙
    DateTime2), N'ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Courses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [Acronym], [Description], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn],         ↙
    [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES (9, N'ACU', 2, 14, N'PHY', N↙
    'Physics with Electronics', NULL, CAST(0x0700000000007B340B AS DateTime2), N'ademola',↙
     NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Courses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],   ↙
    [Acronym], [Description], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn],         ↙
    [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES (10, N'ACU', 4, 15, N'ACC', ↙
    N'Accounting', NULL, CAST(0x0700000000007B340B AS DateTime2), N'ademola', NULL, NULL, ↙
    0, NULL, NULL)
```

```
INSERT [SetUp].[Courses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],      ↙
    [Acronym], [Description], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn],           ↙
    [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES (11, N'ACU', 4, 16, N'BUS', ↙
    N'Business Administration', NULL, CAST(0x0700000000007B340B AS DateTime2), N'ademola',↙
     NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Courses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],      ↙
    [Acronym], [Description], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn],           ↙
    [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES (12, N'ACU', 4, 17, N'ECN', ↙
    N'Economics', NULL, CAST(0x0700000000007B340B AS DateTime2), N'ademola', NULL, NULL, 0↙
    , NULL, NULL)
INSERT [SetUp].[Courses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],      ↙
    [Acronym], [Description], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn],           ↙
    [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES (13, N'ACU', 4, 18, N'MCM', ↙
    N'Mass Communications', NULL, CAST(0x0700000000007B340B AS DateTime2), N'ademola',    ↙
    NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Courses] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],      ↙
    [Acronym], [Description], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn],           ↙
    [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES (16, N'ACU', 1, 9, N'HIS', N↙
    'History and International Studies', N'', CAST(0x07E58BA86B7596340B AS DateTime2), N''↙
    , NULL, NULL, 0, NULL, NULL)
SET IDENTITY_INSERT [SetUp].[Courses] OFF
/****** Object:  Table [SetUp].[Programs]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [SetUp].[Programs](
    [Code] [bigint] IDENTITY(1,1) NOT NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [FacultyCode] [bigint] NULL,
    [DepartmentCode] [bigint] NULL,
    [CourseCode] [bigint] NULL,
    [AwardCode] [nvarchar](50) NULL,
    [ProgCode] [nvarchar](50) NULL,
    [Notes] [nvarchar](1000) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
    [ProgName]  AS ([SetUp].[FnGetDescriptionName]([ProgCode])),
    [AwardInViewName]  AS ([SetUp].[FnGetDescriptionName]([AwardCode])),
    [Description]  AS ([SetUp].[FnGetProgramDescription]([Code])),
 CONSTRAINT [PK_Programs] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,             ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
SET IDENTITY_INSERT [SetUp].[Programs] ON
INSERT [SetUp].[Programs] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],     ↙
    [CourseCode], [AwardCode], [ProgCode], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn]↙
    , [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES (1, N'ACU', 1, 8, 1, N'AIV↙
    -104', N'PRG-124', N'', CAST(0x078A2F08953597340B AS DateTime2), N'', NULL, NULL, 0,   ↙
    NULL, NULL)
INSERT [SetUp].[Programs] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],     ↙
    [CourseCode], [AwardCode], [ProgCode], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn]↙
    , [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES (2, N'ACU', 1, 8, 1, N'AIV↙
    -105', N'PRG-126', N'', CAST(0x0746FEC7EF3597340B AS DateTime2), N'', NULL, NULL, 0,   ↙
    NULL, NULL)
INSERT [SetUp].[Programs] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],     ↙
    [CourseCode], [AwardCode], [ProgCode], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn]↙
    , [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES (3, N'ACU', 1, 9, 16, N   ↙
    'AIV-104', N'PRG-124', N'', CAST(0x07A5630C993897340B AS DateTime2), N'', NULL, NULL, ↙
    0, NULL, NULL)
INSERT [SetUp].[Programs] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],     ↙
    [CourseCode], [AwardCode], [ProgCode], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn]↙
    , [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES (4, N'ACU', 1, 9, 16, N   ↙
    'AIV-105', N'PRG-126', N'', CAST(0x072276A8A93897340B AS DateTime2), N'', NULL, NULL, ↙
    0, NULL, NULL)
INSERT [SetUp].[Programs] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],     ↙
    [CourseCode], [AwardCode], [ProgCode], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn]↙
    , [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES (5, N'ACU', 2, 11, 3, N   ↙
    'AIV-110', N'PRG-124', N'', CAST(0x072B7041C23897340B AS DateTime2), N'', NULL, NULL, ↙
    0, NULL, NULL)
INSERT [SetUp].[Programs] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],     ↙
    [CourseCode], [AwardCode], [ProgCode], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn]↙
    , [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES (6, N'ACU', 2, 11, 3, N   ↙
    'AIV-116', N'PRG-126', N'', CAST(0x074E2F9DCF3897340B AS DateTime2), N'', NULL, NULL, ↙
    0, NULL, NULL)
INSERT [SetUp].[Programs] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],     ↙
    [CourseCode], [AwardCode], [ProgCode], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn]↙
    , [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES (7, N'ACU', 2, 12, 5, N   ↙
    'AIV-110', N'PRG-124', N'', CAST(0x07B4F2611D3997340B AS DateTime2), N'', NULL, NULL, ↙
    0, NULL, NULL)
INSERT [SetUp].[Programs] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],     ↙
    [CourseCode], [AwardCode], [ProgCode], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn]↙
    , [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES (8, N'ACU', 2, 12, 5, N   ↙
    'AIV-116', N'PRG-126', N'', CAST(0x07762D52303997340B AS DateTime2), N'', NULL, NULL, ↙
    0, NULL, NULL)
INSERT [SetUp].[Programs] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],     ↙
    [CourseCode], [AwardCode], [ProgCode], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn]↙
    , [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES (9, N'ACU', 2, 12, 4, N   ↙
    'AIV-110', N'PRG-124', N'', CAST(0x078C95A74D3997340B AS DateTime2), N'', NULL, NULL, ↙
    0, NULL, NULL)
INSERT [SetUp].[Programs] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],     ↙
    [CourseCode], [AwardCode], [ProgCode], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn]↙
    , [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES (10, N'ACU', 2, 12, 4, N  ↙
    'AIV-116', N'PRG-126', N'', CAST(0x072E3AAB5D3997340B AS DateTime2), N'', NULL, NULL, ↙
    0, NULL, NULL)
INSERT [SetUp].[Programs] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],     ↙
    [CourseCode], [AwardCode], [ProgCode], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn]↙
    , [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES (11, N'ACU', 2, 13, 6, N  ↙
    'AIV-110', N'PRG-124', N'', CAST(0x07DD59396A3997340B AS DateTime2), N'', NULL, NULL, ↙
    0, NULL, NULL)
INSERT [SetUp].[Programs] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],     ↙
    [CourseCode], [AwardCode], [ProgCode], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn]↙
    , [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES (12, N'ACU', 2, 13, 6, N  ↙
    'AIV-116', N'PRG-126', N'', CAST(0x07F1E49A783997340B AS DateTime2), N'', NULL, NULL, ↙
    0, NULL, NULL)
INSERT [SetUp].[Programs] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],     ↙
    [CourseCode], [AwardCode], [ProgCode], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn]↙
    , [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES (13, N'ACU', 2, 11, 3, N  ↙
    'AIV-134', N'PRG-129', N'', CAST(0x0795F0B1ED3997340B AS DateTime2), N'', NULL, NULL, ↙
    0, NULL, NULL)
INSERT [SetUp].[Programs] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],     ↙
    [CourseCode], [AwardCode], [ProgCode], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn]↙
    , [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES (14, N'ACU', 2, 12, 4, N  ↙
    'AIV-134', N'PRG-129', N'', CAST(0x07AC5D41073A97340B AS DateTime2), N'', NULL, NULL, ↙
    0, NULL, NULL)
INSERT [SetUp].[Programs] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],     ↙
    [CourseCode], [AwardCode], [ProgCode], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn]↙
    , [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES (15, N'ACU', 2, 12, 5, N  ↙
    'AIV-134', N'PRG-129', N'', CAST(0x07B5C64C203A97340B AS DateTime2), N'', NULL, NULL, ↙
    0, NULL, NULL)
INSERT [SetUp].[Programs] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],     ↙
    [CourseCode], [AwardCode], [ProgCode], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn]↙
    , [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES (16, N'ACU', 2, 13, 6, N  ↙
    'AIV-134', N'PRG-129', N'', CAST(0x072129FD373A97340B AS DateTime2), N'', NULL, NULL, ↙
    0, NULL, NULL)
INSERT [SetUp].[Programs] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],     ↙
    [CourseCode], [AwardCode], [ProgCode], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn]↙
    , [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES (17, N'ACU', 2, 14, 8, N  ↙
    'AIV-110', N'PRG-124', N'', CAST(0x07C01EC2939897340B AS DateTime2), N'', NULL, NULL, ↙
    0, NULL, NULL)
INSERT [SetUp].[Programs] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],     ↙
    [CourseCode], [AwardCode], [ProgCode], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn]↙
    , [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES (18, N'ACU', 2, 14, 8, N  ↙
    'AIV-116', N'PRG-126', N'', CAST(0x0700BF81A89897340B AS DateTime2), N'', NULL, NULL, ↙
    0, NULL, NULL)
SET IDENTITY_INSERT [SetUp].[Programs] OFF
/****** Object:  Table [SetUp].[Durations]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [SetUp].[Durations](
    [Code] [bigint] IDENTITY(1,1) NOT NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [FacultyCode] [bigint] NULL,
    [DepartmentCode] [bigint] NULL,
    [CourseCode] [bigint] NULL,
    [ProgramCode] [bigint] NULL,
    [ModeOfStudy] [nvarchar](50) NULL,
    [DurationValue] [bigint] NULL,
    [DurationUnit] [nvarchar](50) NULL,
    [Notes] [nvarchar](1000) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
    [MOSDescription]  AS ([SetUp].[FnGetDescriptionName]([ModeOfStudy])),
    [DUDescription]  AS ([SetUp].[FnGetDescriptionName]([DurationUnit])),
 CONSTRAINT [PK_Durations] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,             ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
SET IDENTITY_INSERT [SetUp].[Durations] ON
INSERT [SetUp].[Durations] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],    ↙
    [CourseCode], [ProgramCode], [ModeOfStudy], [DurationValue], [DurationUnit], [Notes], ↙
    [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn],         ↙
    [DeletedBy]) VALUES (1, N'ACU', 1, 8, 1, 1, N'MOS-96', 4, N'DU-140', N'', CAST        ↙
    (0x071D2BD1389E98340B AS DateTime2), N'', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Durations] ([Code], [UniversityCode], [FacultyCode], [DepartmentCode],    ↙
    [CourseCode], [ProgramCode], [ModeOfStudy], [DurationValue], [DurationUnit], [Notes], ↙
    [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn],         ↙
    [DeletedBy]) VALUES (2, N'ACU', 1, 9, 16, 3, N'MOS-96', 4, N'DU-140', N'', CAST       ↙
    (0x0777D9DE3EA098340B AS DateTime2), N'', NULL, NULL, 0, NULL, NULL)
SET IDENTITY_INSERT [SetUp].[Durations] OFF
/****** Object:  StoredProcedure [SetUp].[SPDurationsSelect]    Script Date: 08/16/2011 01↙
    :24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for retrieving values from SetUp.Durations Table
```

```sql
CREATE PROC [SetUp].[SPDurationsSelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @FacultyCode BIGINT=NULL,
    @DepartmentCode BIGINT=NULL,
    @CourseCode BIGINT=NULL,
    @ProgramCode BIGINT=NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Durations] WHERE ([Code] = @Code) AND ([UniversityCode] = @UniversityCode) AND ([FacultyCode]=@FacultyCode) AND ([DepartmentCode]=@DepartmentCode) AND ([CourseCode]=@CourseCode) AND ([ProgramCode]=@ProgramCode) AND ([Deleted] = @Deleted))
            BEGIN
            SELECT * FROM [SetUp].[Durations]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([FacultyCode]=@FacultyCode) AND
                ([DepartmentCode]=@DepartmentCode) AND
                ([CourseCode]=@CourseCode) AND
                ([ProgramCode]=@ProgramCode) AND
                ([Deleted] = @Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [SetUp].[Durations]
            WHERE
                (
                ([UniversityCode] = @UniversityCode) AND
                ([FacultyCode]=@FacultyCode) AND
                ([DepartmentCode]=@DepartmentCode) AND
                ([CourseCode]=@CourseCode) AND
                ([ProgramCode]=@ProgramCode) AND
                ([Deleted] = @Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH
```

```
GRANT EXECUTE ON [SetUp].[SPDurationsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPDurationsInsertUpdate]    Script Date: 08/16/ ↙
    2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for saving values into SetUp.Durations Table

CREATE PROC [SetUp].[SPDurationsInsertUpdate]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @FacultyCode BIGINT=NULL,
    @DepartmentCode BIGINT=NULL,
    @CourseCode BIGINT=NULL,
    @ProgramCode BIGINT=NULL,
    @ModeOfStudy NVARCHAR(50)=NULL,
    @DurationValue BIGINT=NULL,
    @DurationUnit NVARCHAR(50)=NULL,
    @Notes NVARCHAR(1000)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[Durations] WHERE ([Code] = @Code) AND (      ↙
    [UniversityCode] = @UniversityCode) AND ([FacultyCode]=@FacultyCode) AND (           ↙
    [DepartmentCode]=@DepartmentCode) AND ([CourseCode]=@CourseCode) AND ([ProgramCode]=@ ↙
    ProgramCode) AND ([Deleted] = @Deleted))
            BEGIN
                INSERT INTO [SetUp].[Durations]
                (
                    [UniversityCode],
                    [FacultyCode],
                    [DepartmentCode],
                    [CourseCode],
                    [ProgramCode],
                    [ModeOfStudy],
                    [DurationValue],
                    [DurationUnit],
                    [Notes],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]

                )
                VALUES
                (
                    @UniversityCode,
                    @FacultyCode,
                    @DepartmentCode,
                    @CourseCode,
                    @ProgramCode,
                    @ModeOfStudy,
                    @DurationValue,
                    @DurationUnit,
                    @Notes,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [SetUp].[Durations]
                SET
                    [UniversityCode]=@UniversityCode,
                    [FacultyCode]=@FacultyCode,
                    [DepartmentCode]=@DepartmentCode,
                    [CourseCode]=@CourseCode,
                    [ProgramCode]=@ProgramCode,
                    [ModeOfStudy]=@ModeOfStudy,
                    [DurationValue]=@DurationValue,
                    [DurationUnit]=@DurationUnit,
                    [Notes]=@Notes,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPDurationsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPDurationsDeletePermanently]    Script Date: 08↙
    /16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/12/2011
--Last Updated      :   07/12/2011
--Last Updated By   :   Ademola Adebo
```

```
--Description       :   Stored procedure for deleting values from SetUp.Durations Table

CREATE PROC [SetUp].[SPDurationsDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Durations] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [SetUp].[Durations]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPDurationsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPDurationsDelete]    Script Date: 08/16/2011 01↙
    :24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/27/2011
--Last Updated      :   07/27/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from SetUp.      ↙
    Durations Table

CREATE PROC [SetUp].[SPDurationsDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
```

```
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[Durations] WHERE (([Code] = @Code) AND    ↙
    (Deleted=@Deleted)))
            BEGIN
            UPDATE [SetUp].[Durations]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPDurationsDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPProgramsSelect]    Script Date: 08/16/2011 01:↙
    24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author             :   Ademola Adebo
--Reviewer           :   Godwin Mathias
--Date Created       :   07/08/2011
--Last Updated       :   07/08/2011
--Last Updated By    :   Ademola Adebo
--Description        :   Stored procedure for retrieving values from SetUp.Programs Table

CREATE PROC [SetUp].[SPProgramsSelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @FacultyCode BIGINT=NULL,
    @DepartmentCode BIGINT=NULL,
    @CourseCode BIGINT=NULL,
    @Deleted BIT=NULL
)
AS
```

```
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Programs] WHERE ([Code] = @Code) AND (   ↙
    [UniversityCode] = @UniversityCode) AND ([FacultyCode]=@FacultyCode) AND (     ↙
    [DepartmentCode]=@DepartmentCode) AND ([CourseCode]=@CourseCode) AND ([Deleted] = @  ↙
    Deleted))
            BEGIN
            SELECT * FROM [SetUp].[Programs]
            WHERE
            (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([FacultyCode]=@FacultyCode) AND
                ([DepartmentCode]=@DepartmentCode) AND
                ([CourseCode]=@CourseCode) AND
                ([Deleted] = @Deleted)
            )
            END
        ELSE
            BEGIN
            SELECT * FROM [SetUp].[Programs]
            WHERE
            (
                ([UniversityCode] = @UniversityCode) AND
                ([FacultyCode]=@FacultyCode) AND
                ([DepartmentCode]=@DepartmentCode) AND
                ([CourseCode]=@CourseCode) AND
                ([Deleted] = @Deleted)
            )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPProgramsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPProgramsInsertUpdate]    Script Date: 08/16/  ↙
    2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
```

```
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into SetUp.Programs Table

CREATE PROC [SetUp].[SPProgramsInsertUpdate]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @FacultyCode BIGINT=NULL,
    @DepartmentCode BIGINT=NULL,
    @CourseCode BIGINT=NULL,
    @AwardCode NVARCHAR(50)=NULL,
    @ProgCode NVARCHAR(50)=NULL,
    @Notes NVARCHAR(1000)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[Programs]  WHERE ([Code] = @Code) AND (      ↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
                INSERT INTO [SetUp].[Programs]
                (
                    [UniversityCode],
                    [FacultyCode],
                    [DepartmentCode],
                    [CourseCode],
                    [AwardCode],
                    [ProgCode],
                    [Notes],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]

                )
                VALUES
                (
                    @UniversityCode,
                    @FacultyCode,
                    @DepartmentCode,
                    @CourseCode,
                    @AwardCode,
                    @ProgCode,
                    @Notes,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [SetUp].[Programs]
                SET
                    [UniversityCode]=@UniversityCode,
                    [FacultyCode]=@FacultyCode,
                    [DepartmentCode]=@DepartmentCode,
                    [CourseCode]=@CourseCode,
                    [AwardCode]=@AwardCode,
```

```sql
                    [ProgCode]=@ProgCode,
                    [Notes]=@Notes,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPProgramsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPProgramsDeletePermanently]    Script Date: 08/16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/12/2011
--Last Updated        :   07/12/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for deleting values from SetUp.Programs Table

CREATE PROC [SetUp].[SPProgramsDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Programs] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [SetUp].[Programs]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY
```

```
BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPProgramsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPProgramsDelete]    Script Date: 08/16/2011 01:↙
    24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/28/2011
--Last Updated      :   07/28/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from SetUp.      ↙
    Programs Table

CREATE PROC [SetUp].[SPProgramsDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[Programs] WHERE (([Code] = @Code) AND       ↙
    (Deleted=@Deleted)))
            BEGIN
            UPDATE [SetUp].[Programs]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
```

```
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPProgramsDelete] TO PUBLIC
GO
/****** Object:  Table [Personals].[Sports]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Personals].[Sports](
    [Code] [int] IDENTITY(1,1) NOT NULL,
    [SportCode] [nvarchar](50) NULL,
    [AccountCode] [nvarchar](50) NULL,
    [ScreenCode] [nvarchar](50) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Sport] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,           ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Personals].[Sponsors]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Personals].[Sponsors](
    [Code] [int] IDENTITY(1,1) NOT NULL,
    [SponsorCode] [nvarchar](50) NULL,
    [AccountCode] [nvarchar](50) NULL,
    [ScreenCode] [nvarchar](50) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Sponsor] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,           ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Finance].[TransactionsDetail]    Script Date: 08/16/2011 01:24:32 ↙
    ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Finance].[TransactionsDetail](
    [Code] [bigint] NOT NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [SessionCode] [bigint] NULL,
    [SemesterCode] [nvarchar](50) NULL,
    [InvoiceCode] [decimal](18, 0) NULL,
    [ItemDescription] [nvarchar](500) NULL,
    [Quantity] [int] NULL,
    [Unit] [nvarchar](50) NULL,
    [UnitCost] [decimal](18, 2) NULL,
    [DiscountPercentage] [decimal](18, 2) NULL,
    [DiscountAmount] [decimal](18, 2) NULL,
    [TaxType] [nvarchar](50) NULL,
    [TaxValue] [decimal](18, 2) NULL,
    [TotalCost] [decimal](18, 2) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_InvoicesDetail] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,          ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [SetUp].[SubjectRequirements]    Script Date: 08/16/2011 01:24:32 *↙
    *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [SetUp].[SubjectRequirements](
    [Code] [bigint] IDENTITY(1,1) NOT NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [FacultyCode] [bigint] NULL,
    [DepartmentCode] [bigint] NULL,
    [CourseCode] [bigint] NULL,
    [ProgramCode] [bigint] NULL,
    [SubjectCode] [nvarchar](50) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_SubjectRequirements] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,          ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [SetUp].[Countries]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
```

```sql
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [SetUp].[Countries](
    [Code] [nvarchar](50) NOT NULL,
    [Name] [nvarchar](256) NOT NULL,
 CONSTRAINT [PK_Countries] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,        ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
INSERT [SetUp].[Countries] ([Code], [Name]) VALUES (N'AFG', N'Afghanistan')
INSERT [SetUp].[Countries] ([Code], [Name]) VALUES (N'ARG', N'Argentina')
INSERT [SetUp].[Countries] ([Code], [Name]) VALUES (N'AUS', N'Australia')
INSERT [SetUp].[Countries] ([Code], [Name]) VALUES (N'BF', N'Burkina faso')
INSERT [SetUp].[Countries] ([Code], [Name]) VALUES (N'BRA', N'Brazil')
INSERT [SetUp].[Countries] ([Code], [Name]) VALUES (N'CAM', N'Cameroun')
INSERT [SetUp].[Countries] ([Code], [Name]) VALUES (N'CD', N'Cote D'' Ivore')
INSERT [SetUp].[Countries] ([Code], [Name]) VALUES (N'ENG', N'England')
INSERT [SetUp].[Countries] ([Code], [Name]) VALUES (N'FRA', N'France')
INSERT [SetUp].[Countries] ([Code], [Name]) VALUES (N'GER', N'Germany')
INSERT [SetUp].[Countries] ([Code], [Name]) VALUES (N'GHA', N'Ghana')
INSERT [SetUp].[Countries] ([Code], [Name]) VALUES (N'ITA', N'Italy')
INSERT [SetUp].[Countries] ([Code], [Name]) VALUES (N'JAM', N'Jamaica')
INSERT [SetUp].[Countries] ([Code], [Name]) VALUES (N'MEX', N'Mexico')
INSERT [SetUp].[Countries] ([Code], [Name]) VALUES (N'NGN', N'Nigeria ')
INSERT [SetUp].[Countries] ([Code], [Name]) VALUES (N'RSA', N'South Africa')
INSERT [SetUp].[Countries] ([Code], [Name]) VALUES (N'TZ', N'Tanzania Republic')
INSERT [SetUp].[Countries] ([Code], [Name]) VALUES (N'USA', N'United States of America')
/****** Object:  Table [Personals].[ComputerSkills]    Script Date: 08/16/2011 01:24:32 **↙
    ****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Personals].[ComputerSkills](
    [Code] [int] IDENTITY(1,1) NOT NULL,
    [AccountCode] [nvarchar](50) NULL,
    [ScreenCode] [nvarchar](50) NULL,
    [PackageCode] [nvarchar](50) NULL,
    [LevelOfKnowledgeCode] [nvarchar](50) NULL,
    [Notes] [nvarchar](256) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_ComputerSkills] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,        ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Personals].[Companies]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Personals].[Companies](
    [Code] [int] NOT NULL,
    [AccountCode] [nvarchar](50) NULL,
    [ScreenCode] [nvarchar](50) NULL,
    [CompanyName] [nvarchar](256) NOT NULL,
    [ContactName] [nvarchar](150) NULL,
    [ContactTitle] [nvarchar](50) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Companies] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,        ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Personals].[Choices]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Personals].[Choices](
    [Code] [int] IDENTITY(1,1) NOT NULL,
    [FirstChoice] [nvarchar](50) NULL,
    [SecondChoice] [nvarchar](50) NULL,
    [AccountCode] [nvarchar](50) NULL,
    [ScreenCode] [nvarchar](50) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Choices] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,        ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Personals].[Certificates]    Script Date: 08/16/2011 01:24:32 ****↙
    **/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Personals].[Certificates](
    [Code] [int] IDENTITY(1,1) NOT NULL,
    [AccountCode] [nvarchar](50) NULL,
    [ScreenCode] [nvarchar](50) NULL,
    [AwardingBody] [nvarchar](50) NULL,
    [YearObtained] [nvarchar](4) NOT NULL,
    [Title] [nvarchar](256) NULL,
    [Certificate] [nvarchar](50) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Certificates] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,        ↙
```

```
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Personals].[BioDatas]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Personals].[BioDatas](
    [Code] [nvarchar](50) NOT NULL,
    [Surname] [nvarchar](50) NULL,
    [MiddleName] [nvarchar](50) NULL,
    [FirstName] [nvarchar](50) NOT NULL,
    [GenderCode] [nvarchar](50) NULL,
    [CivilStatus] [nvarchar](50) NULL,
    [DateOfBirth] [datetime] NULL,
    [CountryCode] [nvarchar](50) NULL,
    [StateCode] [nvarchar](50) NULL,
    [LGACode] [nvarchar](50) NULL,
    [PlaceOfBirth] [nvarchar](50) NULL,
    [Height] [nvarchar](50) NULL,
    [Weight] [nvarchar](50) NULL,
    [HealthStatusCode] [nvarchar](50) NULL,
    [ScreenCode] [nvarchar](50) NULL,
    [PositionInFamily] [int] NULL,
    [NoOfChildren] [int] NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
    [TicketCode] [numeric](18, 0) NULL,
    [PinCode] [numeric](18, 0) NULL,
 CONSTRAINT [PK_BioDatas] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,        ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  StoredProcedure [dbo].[aspnet_Setup_RestorePermissions]    Script Date:   ↙
    08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_Setup_RestorePermissions]
    @name   sysname
AS
BEGIN
    DECLARE @object sysname
    DECLARE @protectType char(10)
    DECLARE @action varchar(60)
    DECLARE @grantee sysname
    DECLARE @cmd nvarchar(500)
    DECLARE c1 cursor FORWARD_ONLY FOR
        SELECT Object, ProtectType, [Action], Grantee FROM #aspnet_Permissions where     ↙
    Object = @name

    OPEN c1

    FETCH c1 INTO @object, @protectType, @action, @grantee
    WHILE (@@fetch_status = 0)
    BEGIN
        SET @cmd = @protectType + ' ' + @action + ' on ' + @object + ' TO [' + @grantee + ↙
    ']'
        EXEC (@cmd)
        FETCH c1 INTO @object, @protectType, @action, @grantee
    END

    CLOSE c1
    DEALLOCATE c1
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_Setup_RemoveAllRoleMembers]    Script Date:↙
     08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_Setup_RemoveAllRoleMembers]
    @name   sysname
AS
BEGIN
    CREATE TABLE #aspnet_RoleMembers
    (
        Group_name      sysname,
        Group_id        smallint,
        Users_in_group  sysname,
        User_id         smallint
    )

    INSERT INTO #aspnet_RoleMembers
    EXEC sp_helpuser @name

    DECLARE @user_id smallint
    DECLARE @cmd nvarchar(500)
    DECLARE c1 cursor FORWARD_ONLY FOR
        SELECT User_id FROM #aspnet_RoleMembers

    OPEN c1

    FETCH c1 INTO @user_id
    WHILE (@@fetch_status = 0)
    BEGIN
        SET @cmd = 'EXEC sp_droprolemember ' + '''' + @name + ''', ''' + USER_NAME(@      ↙
    user_id) + ''''
        EXEC (@cmd)
        FETCH c1 INTO @user_id
    END

    CLOSE c1
    DEALLOCATE c1
END
GO
/****** Object:  Table [dbo].[aspnet_SchemaVersions]    Script Date: 08/16/2011 01:24:32 *↙
    *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[aspnet_SchemaVersions](
    [Feature] [nvarchar](128) NOT NULL,
    [CompatibleSchemaVersion] [nvarchar](128) NOT NULL,
    [IsCurrentVersion] [bit] NOT NULL,
PRIMARY KEY CLUSTERED
(
    [Feature] ASC,
    [CompatibleSchemaVersion] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,           ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
```

```
GO
INSERT [dbo].[aspnet_SchemaVersions] ([Feature], [CompatibleSchemaVersion],        ↙
    [IsCurrentVersion]) VALUES (N'common', N'1', 1)
INSERT [dbo].[aspnet_SchemaVersions] ([Feature], [CompatibleSchemaVersion],        ↙
    [IsCurrentVersion]) VALUES (N'health monitoring', N'1', 1)
INSERT [dbo].[aspnet_SchemaVersions] ([Feature], [CompatibleSchemaVersion],        ↙
    [IsCurrentVersion]) VALUES (N'membership', N'1', 1)
INSERT [dbo].[aspnet_SchemaVersions] ([Feature], [CompatibleSchemaVersion],        ↙
    [IsCurrentVersion]) VALUES (N'personalization', N'1', 1)
INSERT [dbo].[aspnet_SchemaVersions] ([Feature], [CompatibleSchemaVersion],        ↙
    [IsCurrentVersion]) VALUES (N'profile', N'1', 1)
INSERT [dbo].[aspnet_SchemaVersions] ([Feature], [CompatibleSchemaVersion],        ↙
    [IsCurrentVersion]) VALUES (N'role manager', N'1', 1)
/****** Object:  Table [dbo].[aspnet_Applications]    Script Date: 08/16/2011 01:24:32 ***↙
    ***/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[aspnet_Applications](
    [ApplicationName] [nvarchar](256) NOT NULL,
    [LoweredApplicationName] [nvarchar](256) NOT NULL,
    [ApplicationId] [uniqueidentifier] NOT NULL,
    [Description] [nvarchar](256) NULL,
PRIMARY KEY NONCLUSTERED
(
    [ApplicationId] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,          ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY],
UNIQUE NONCLUSTERED
(
    [LoweredApplicationName] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,          ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY],
UNIQUE NONCLUSTERED
(
    [ApplicationName] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,          ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
INSERT [dbo].[aspnet_Applications] ([ApplicationName], [LoweredApplicationName],       ↙
    [ApplicationId], [Description]) VALUES (N'Xcellentnovelty Solace', N'xcellentnovelty ↙
    solace', N'3436a128-b3cb-4d6a-a7f2-782b22a6d04d', NULL)
/****** Object:  Table [SetUp].[AreasOfSpecializations]    Script Date: 08/16/2011 01:24: ↙
    32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [SetUp].[AreasOfSpecializations](
    [Code] [int] IDENTITY(1,1) NOT NULL,
    [DescriptionsCode] [nvarchar](50) NOT NULL,
    [Specialization] [nvarchar](256) NULL,
 CONSTRAINT [PK_AreasOfSpecializations] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,          ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Personals].[AreasOfExpertises]    Script Date: 08/16/2011 01:24:32↙
     ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
```

```
CREATE TABLE [Personals].[AreasOfExpertises](
    [Code] [int] IDENTITY(1,1) NOT NULL,
    [AccountCode] [nvarchar](50) NULL,
    [ScreenCode] [nvarchar](50) NULL,
    [JobCategoryCode] [nvarchar](50) NULL,
    [AOSCode] [nvarchar](50) NULL,
    [YearOfExperience] [nvarchar](50) NULL,
    [LeveOfExperience] [nvarchar](50) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_AreasOfExpertises] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,        ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Personals].[Addresses]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Personals].[Addresses](
    [Code] [int] IDENTITY(1,1) NOT NULL,
    [AddressTypeCode] [nvarchar](50) NULL,
    [AccountCode] [nvarchar](50) NULL,
    [Description] [nvarchar](256) NULL,
    [ScreenCode] [nvarchar](50) NULL,
    [CountryCode] [nvarchar](50) NULL,
    [StateCode] [nvarchar](50) NULL,
    [LgCode] [nvarchar](50) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Addresses] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,        ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  UserDefinedFunction [Academics].[GetSessionDescription]    Script Date:  ↙
    08/16/2011 01:24:33 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE FUNCTION [Academics].[GetSessionDescription](@CurrentYear int,@NextYear int)      ↙
    RETURNS NVARCHAR(50)
AS
BEGIN
        RETURN ISNULL((CONVERT(NVARCHAR(5), @CurrentYear)+'/'+CONVERT(NVARCHAR(5), @     ↙
    NextYear)),'N/A')
END
GO
/****** Object:  Table [Personals].[MedicalConditions]    Script Date: 08/16/2011 01:24:32↙
     ******/
SET ANSI_NULLS ON
```

```
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Personals].[MedicalConditions](
    [Code] [int] IDENTITY(1,1) NOT NULL,
    [DiseaseCode] [nvarchar](50) NULL,
    [Conditions] [nvarchar](500) NULL,
    [AccountCode] [nvarchar](50) NULL,
    [ScreenCode] [nvarchar](50) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_MedicalConditions] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,       ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [SetUp].[Lockings]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [SetUp].[Lockings](
    [Code] [bigint] NOT NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [FacultyCode] [bigint] NULL,
    [DepartmentCode] [bigint] NULL,
    [CourseCode] [bigint] NULL,
    [ProgramCode] [bigint] NULL,
    [SessionCode] [nvarchar](50) NULL,
    [SemesterCode] [nvarchar](50) NULL,
    [LevelCode] [int] NULL,
    [AccountCode] [nvarchar](50) NULL,
    [EntityCode] [nvarchar](50) NULL,
    [Locked] [bit] NULL,
    [LockedOn] [datetime] NULL,
    [LockedBy] [nvarchar](50) NULL,
    [UnlockedOn] [datetime] NULL,
    [UnlockedBy] [nvarchar](50) NULL,
    [LastLockedOn] [datetime] NULL,
    [LastLockedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Lockings_1] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,       ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Personals].[LanguageSkills]    Script Date: 08/16/2011 01:24:32 **↙
    ****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Personals].[LanguageSkills](
    [Code] [int] IDENTITY(1,1) NOT NULL,
    [LanguageCode] [nvarchar](50) NULL,
    [AccountCode] [nvarchar](50) NULL,
    [DateX] [datetime] NULL,
    [ScreenCode] [nvarchar](50) NULL,
    [ReadingDegree] [nvarchar](50) NULL,
    [SpeakingDegree] [nvarchar](50) NULL,
    [WritingDegree] [nvarchar](50) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_LanguageSkills] PRIMARY KEY CLUSTERED 
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,          ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Personals].[JAMBs]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Personals].[JAMBs](
    [Code] [int] IDENTITY(1,1) NOT NULL,
    [RegNo] [nvarchar](50) NULL,
    [Score] [int] NULL,
    [AccountCode] [nvarchar](50) NULL,
    [ScreenCode] [nvarchar](50) NULL,
    [Year] [int] NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_JAMB] PRIMARY KEY CLUSTERED 
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,          ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Personals].[Hobbies]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Personals].[Hobbies](
    [Code] [int] IDENTITY(1,1) NOT NULL,
    [AccountCode] [nvarchar](50) NOT NULL,
    [ScreenCode] [nvarchar](50) NULL,
    [HobbyCode] [nvarchar](50) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Hobbies] PRIMARY KEY CLUSTERED 
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,          ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Personals].[Guardians]    Script Date: 08/16/2011 01:24:32 ******/
```

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Personals].[Guardians](
    [Code] [int] IDENTITY(1,1) NOT NULL,
    [GuardianCode] [nvarchar](50) NULL,
    [TitleCode] [nvarchar](50) NULL,
    [GuardianName] [nvarchar](50) NULL,
    [OccupationCode] [nvarchar](50) NULL,
    [AccountCode] [nvarchar](50) NULL,
    [ScreenCode] [nvarchar](50) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Guardians] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,       ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Personals].[Guarantors]    Script Date: 08/16/2011 01:24:32 ******↙
    /
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Personals].[Guarantors](
    [Code] [int] NOT NULL,
    [TitleCode] [nvarchar](50) NULL,
    [Name] [nvarchar](150) NULL,
    [OccupationCode] [nvarchar](50) NULL,
    [AccountCode] [nvarchar](50) NULL,
    [ScreenCode] [nvarchar](50) NULL,
    [Undertaking] [nvarchar](500) NOT NULL,
    [Notes] [nvarchar](500) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Guarantors] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,       ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Assessment].[Definition]    Script Date: 08/16/2011 01:24:32 *****↙
    */
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Assessment].[Definition](
    [Code] [bigint] IDENTITY(1,1) NOT NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [FacultyCode] [bigint] NULL,
    [DepartmentCode] [bigint] NULL,
    [CourseCode] [bigint] NULL,
    [ProgramCode] [bigint] NULL,
    [SessionCode] [bigint] NULL,
    [SemesterCode] [bigint] NULL,
    [AssessmentTypeCode] [nvarchar](50) NULL,
    [Value] [decimal](18, 2) NULL,
    [NoOfAssessment] [int] NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Definition] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,       ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Personals].[Declarations]    Script Date: 08/16/2011 01:24:32 ****↙
    **/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Personals].[Declarations](
    [Code] [int] IDENTITY(1,1) NOT NULL,
    [AccountCode] [nvarchar](50) NULL,
    [ScreenCode] [nvarchar](50) NULL,
    [Notes] [nvarchar](1000) NULL,
    [GuardianCode] [int] NOT NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Declarations] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,       ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[aspnet_WebEvent_Events]    Script Date: 08/16/2011 01:24:32 ↙
    ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
SET ANSI_PADDING ON
GO
CREATE TABLE [dbo].[aspnet_WebEvent_Events](
    [EventId] [char](32) NOT NULL,
    [EventTimeUtc] [datetime] NOT NULL,
    [EventTime] [datetime] NOT NULL,
    [EventType] [nvarchar](256) NOT NULL,
    [EventSequence] [decimal](19, 0) NOT NULL,
    [EventOccurrence] [decimal](19, 0) NOT NULL,
    [EventCode] [int] NOT NULL,
    [EventDetailCode] [int] NOT NULL,
    [Message] [nvarchar](1024) NULL,
    [ApplicationPath] [nvarchar](256) NULL,
    [ApplicationVirtualPath] [nvarchar](256) NULL,
    [MachineName] [nvarchar](256) NOT NULL,
    [RequestUrl] [nvarchar](1024) NULL,
    [ExceptionType] [nvarchar](256) NULL,
    [Details] [ntext] NULL,
PRIMARY KEY CLUSTERED
(
    [EventId] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,        ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
GO
SET ANSI_PADDING OFF
GO
/****** Object:  Table [SetUp].[CourseRegulation]    Script Date: 08/16/2011 01:24:32 ****↙
    **/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [SetUp].[CourseRegulation](
    [Code] [bigint] IDENTITY(1,1) NOT NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [FacultyCode] [bigint] NULL,
    [DepartmentCode] [bigint] NULL,
    [CourseCode] [bigint] NULL,
    [ProgramCode] [bigint] NULL,
    [LevelCode] [bigint] NULL,
    [SemesterCode] [nvarchar](50) NULL,
    [CreditLowerBound] [int] NULL,
    [CreditUpperBound] [int] NULL,
    [ModeOfStudyCode] [nvarchar](50) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_CourseRegulation] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,        ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [SetUp].[EntryRequirements]    Script Date: 08/16/2011 01:24:32 ***↙
    ***/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [SetUp].[EntryRequirements](
    [Code] [bigint] NOT NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [FacultyCode] [bigint] NULL,
    [DepartmentCode] [bigint] NULL,
    [CourseCode] [bigint] NULL,
    [ProgramCode] [bigint] NULL,
    [EntryMode] [nvarchar](50) NULL,
    [ModeOfStudy] [nvarchar](50) NULL,
    [NoOfCredits] [int] NULL,
    [NoOfSittings] [int] NULL,
    [RequirementType] [nvarchar](50) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_EntryRequirements] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,           ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Personals].[EmploymentHistories]    Script Date: 08/16/2011 01:24:↙
    32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Personals].[EmploymentHistories](
    [Code] [int] IDENTITY(1,1) NOT NULL,
    [AccountCode] [nvarchar](50) NULL,
    [ScreenCode] [nvarchar](50) NULL,
    [TitleCode] [nvarchar](50) NULL,
    [JobCategoryCode] [nvarchar](50) NULL,
    [JobDescCode] [nvarchar](50) NULL,
    [LevelOfXpertiseCode] [nvarchar](50) NULL,
    [WorkPercentageCode] [nvarchar](50) NULL,
    [FromYearCode] [nvarchar](50) NULL,
    [ToYearCode] [nvarchar](50) NULL,
    [FromMonthCode] [nvarchar](50) NULL,
    [ToMonthCode] [nvarchar](50) NULL,
    [FromSAS] [nvarchar](50) NULL,
    [FromSASCurrencyCode] [nvarchar](50) NULL,
    [FromSASTotals] [decimal](18, 0) NULL,
    [FromEAS] [nvarchar](50) NULL,
    [FromEASCurrencyCode] [nvarchar](50) NULL,
    [FromEASTotals] [decimal](18, 0) NULL,
    [Supervisor] [nvarchar](150) NULL,
    [SupervisorTitle] [nvarchar](50) NULL,
    [Employer] [nvarchar](256) NULL,
    [EmployerAddress] [nvarchar](256) NULL,
    [EmployerBusinessNatureCode] [nvarchar](50) NOT NULL,
    [EmployerURL] [nvarchar](256) NULL,
    [DutiesDescription] [nvarchar](500) NULL,
    [KeyAchievements] [nvarchar](500) NULL,
    [ContactRefrenceCode] [nvarchar](50) NULL,
    [NoOfPeopleSupervised] [int] NULL,
    [ReasonForLeaving] [nvarchar](256) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_EmploymentHistories] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,           ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Personals].[Emergencies]    Script Date: 08/16/2011 01:24:32 *****↙
    */
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Personals].[Emergencies](
    [Code] [int] IDENTITY(1,1) NOT NULL,
    [AccountCode] [nvarchar](50) NULL,
    [ScreenCode] [nvarchar](50) NULL,
    [TitleCode] [nvarchar](50) NULL,
    [Name] [nvarchar](150) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Emergencies] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,    ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Personals].[Emails]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Personals].[Emails](
    [Code] [int] IDENTITY(1,1) NOT NULL,
    [AccountCode] [nvarchar](50) NULL,
    [Email] [nvarchar](50) NULL,
    [ScreenCode] [nvarchar](50) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Emails] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,    ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [SetUp].[ProgramRequirements]    Script Date: 08/16/2011 01:24:32 *↙
    *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [SetUp].[ProgramRequirements](
    [Code] [bigint] IDENTITY(1,1) NOT NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [FacultyCode] [bigint] NULL,
    [DepartmentCode] [bigint] NULL,
    [CourseCode] [bigint] NULL,
    [ProgramCode] [bigint] NULL,
    [MinimumCredit] [int] NULL,
    [MaximumCredit] [int] NULL,
    [TotalDuration] [bigint] NULL,
    [DurationUnit] [nvarchar](50) NULL,
    [EntryMode] [nvarchar](50) NULL,
    [ModeOfStudy] [nvarchar](50) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_ProgramRequirements] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,      ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Personals].[Photos]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
SET ANSI_PADDING ON
GO
CREATE TABLE [Personals].[Photos](
    [Code] [int] NOT NULL,
    [ModuleCode] [nvarchar](50) NULL,
    [ScreenCode] [nvarchar](50) NULL,
    [ImageTypeCode] [nvarchar](50) NULL,
    [AccountCode] [nvarchar](50) NULL,
    [ByteThumb] [varbinary](max) NULL,
    [BytePoster] [varbinary](max) NULL,
    [ByteFull] [varbinary](max) NULL,
    [ByteOriginal] [varbinary](max) NULL,
    [Notes] [nvarchar](256) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Photos] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,      ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
SET ANSI_PADDING OFF
GO
/****** Object:  Table [Personals].[Phones]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Personals].[Phones](
    [Code] [int] IDENTITY(1,1) NOT NULL,
    [AccountCode] [nvarchar](50) NULL,
    [PhoneTypeCode] [nvarchar](50) NULL,
    [ScreenCode] [nvarchar](50) NULL,
    [PostalCode] [nvarchar](5) NULL,
    [PhoneNumber] [nvarchar](50) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Phones] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,      ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
```

```
GO
/****** Object:  Table [SetUp].[Permissions]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [SetUp].[Permissions](
    [Code] [bigint] IDENTITY(1,1) NOT NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [ModuleCode] [nvarchar](50) NULL,
    [ScreenCode] [nvarchar](50) NOT NULL,
    [RoleCode] [uniqueidentifier] NULL,
    [ActionCode] [nvarchar](50) NOT NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Permissions] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,           ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Personals].[Qualifications]    Script Date: 08/16/2011 01:24:32 **↙
    ****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Personals].[Qualifications](
    [Code] [int] IDENTITY(1,1) NOT NULL,
    [AccountCode] [nvarchar](50) NULL,
    [ScreenCode] [nvarchar](50) NULL,
    [EducationTypeCode] [nvarchar](50) NULL,
    [QualificationTypeCode] [nvarchar](50) NULL,
    [FromYear] [nvarchar](50) NULL,
    [ToYear] [nvarchar](50) NULL,
    [FromMonth] [nvarchar](50) NULL,
    [ToMonth] [nvarchar](50) NULL,
    [AwardingBody] [nvarchar](256) NULL,
    [Certificate] [nvarchar](50) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Qualifications] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,           ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Personals].[Referees]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Personals].[Referees](
    [Code] [int] NOT NULL,
    [TitleCode] [nvarchar](50) NULL,
    [Name] [nvarchar](150) NULL,
    [OccupationCode] [nvarchar](50) NULL,
    [AccountCode] [nvarchar](50) NULL,
    [ScreenCode] [nvarchar](50) NULL,
    [KnowingMode] [nvarchar](256) NULL,
    [ContactReferenceCode] [nvarchar](50) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Referees] PRIMARY KEY CLUSTERED 
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,          ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Personals].[Others]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Personals].[Others](
    [Code] [int] IDENTITY(1,1) NOT NULL,
    [AccountCode] [nvarchar](50) NULL,
    [Notes] [nvarchar](500) NULL,
    [ScreenCode] [nvarchar](50) NULL,
    [OthersCode] [nvarchar](50) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Others] PRIMARY KEY CLUSTERED 
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,          ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Personals].[OLevels]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Personals].[OLevels](
    [Code] [int] IDENTITY(1,1) NOT NULL,
    [SubjectCode] [nvarchar](50) NULL,
    [AccountCode] [nvarchar](50) NULL,
    [ScreenCode] [nvarchar](50) NULL,
    [GradeCode] [int] NULL,
    [GradeDescCode] [nvarchar](50) NOT NULL,
    [ExamTypeCode] [nvarchar](50) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_OLevels] PRIMARY KEY CLUSTERED 
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,        ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Personals].[NextOfKins]    Script Date: 08/16/2011 01:24:32 ******↙
    /
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Personals].[NextOfKins](
    [Code] [int] IDENTITY(1,1) NOT NULL,
    [RelTypeCode] [nvarchar](50) NULL,
    [TitleCode] [nvarchar](50) NULL,
    [Name] [nvarchar](50) NULL,
    [OccupationCode] [nvarchar](50) NULL,
    [AccountCode] [nvarchar](50) NULL,
    [ScreenCode] [nvarchar](50) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_NextOfKins] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,        ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [SetUp].[Schedules]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [SetUp].[Schedules](
    [Code] [bigint] NOT NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [FacultyCode] [bigint] NULL,
    [DepartmentCode] [bigint] NULL,
    [CourseCode] [bigint] NULL,
    [ProgramCode] [bigint] NULL,
    [SemesterCode] [nvarchar](50) NULL,
    [LevelCode] [int] NULL,
    [SubCode] [bigint] NULL,
    [ModeOfStudy] [nvarchar](50) NULL,
    [StaffCode] [nvarchar](50) NULL,
    [DateX] [datetime2](7) NULL,
    [StartTime] [nvarchar](50) NULL,
    [StopTime] [nvarchar](50) NULL,
    [ScheduleType] [nvarchar](50) NULL,
    [Notes] [nvarchar](500) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Schedules] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,        ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
```

```
) ON [PRIMARY]
GO
/****** Object:  Table [Personals].[Religions]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Personals].[Religions](
    [Code] [int] IDENTITY(1,1) NOT NULL,
    [AccountCode] [nvarchar](50) NULL,
    [BioDataCode] [nvarchar](50) NOT NULL,
    [ReligionCode] [nvarchar](50) NULL,
    [Clergy] [nvarchar](150) NULL,
    [PlaceOfWorship] [nvarchar](256) NULL,
    [PlaceOfWorshipAddress] [nvarchar](256) NULL,
    [ClergyPhoneNumber] [nvarchar](50) NOT NULL,
    [ClergyEmail] [nvarchar](50) NULL,
    [PlaceOfWorshipPhoneNumber] [nvarchar](50) NULL,
    [PlaceOfWorshipEmail] [nvarchar](50) NULL,
    [ScreenCode] [nvarchar](50) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Religions] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,          ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  StoredProcedure [SetUp].[SPCountriesSelect]    Script Date: 08/16/2011 01↙
    :24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for retrieving values from SetUp.Countries Table

CREATE PROC [SetUp].[SPCountriesSelect]
(
    @Code NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Countries] WHERE ([Code] = @Code))
            BEGIN
            SELECT * FROM [SetUp].[Countries]
            WHERE
                (
                ([Code] = @Code)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [SetUp].[Countries]
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPCountriesSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPCountriesInsertUpdate]    Script Date: 08/16/ ↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into SetUp.Countries Table

CREATE PROC [SetUp].[SPCountriesInsertUpdate]
(
    @Code NVARCHAR(50)=NULL,
    @Name NVARCHAR(256)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[Countries]  WHERE ([Code] = @Code))
            BEGIN
                INSERT INTO [SetUp].[Countries]
                (
                    [Code],
                    [Name]
                )
                VALUES
                (
                    @Code,
                    @Name
                )
            END
        ELSE
            BEGIN
                UPDATE [SetUp].[Countries]
                SET
```

```
                    [Name]=@Name
                WHERE
                    ([Code] = @Code)
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPCountriesInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPCountriesDeletePermanently]    Script Date: 08↙
    /16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from SetUp.Countries Table

CREATE PROC [SetUp].[SPCountriesDeletePermanently]
(
    @Code NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Countries] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [SetUp].[Countries]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPCountriesDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPComputerSkillsSelect]    Script Date: 08/ ↙
    16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Personal.          ↙
    ComputerSkills Table

CREATE PROC [Personals].[SPComputerSkillsSelect]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[ComputerSkills] WHERE (([Code] = @Code) AND ↙
    ([AccountCode]=@AccountCode) AND ([ScreenCode]=@ScreenCode) AND ([Deleted]=@Deleted)))
            BEGIN
            SELECT * FROM [Personals].[ComputerSkills]
            WHERE
                (
                ([Code] = @Code) AND
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Personals].[ComputerSkills]
            WHERE
                (
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
COMMIT TRAN
```

```
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPComputerSkillsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPComputerSkillsInsertUpdate]    Script Date↙
    : 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for saving values into Personals.ComputerSkills  ↙
    Table

CREATE PROC [Personals].[SPComputerSkillsInsertUpdate]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @PackageCode NVARCHAR(50)=NULL,
    @LevelOfKnowledgeCode NVARCHAR(50)=NULL,
    @Notes NVARCHAR(256)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Personals].[ComputerSkills] WHERE (([Code] = @Code) ↙
    AND ([AccountCode]=@AccountCode) AND ([Deleted]=@Deleted)))
            BEGIN
                INSERT INTO [Personals].[ComputerSkills]
                (
                    [AccountCode],
                    [ScreenCode],
                    [PackageCode],
                    [LevelOfKnowledgeCode],
                    [Notes],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    @AccountCode,
                    @ScreenCode,
                    @PackageCode,
                    @LevelOfKnowledgeCode,
                    @Notes,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [Personals].[ComputerSkills]
                SET
                    [AccountCode]=@AccountCode,
                    [ScreenCode]=@ScreenCode,
                    [PackageCode]=@PackageCode,
                    [LevelOfKnowledgeCode]=@LevelOfKnowledgeCode,
                    [Notes]=@Notes,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([AccountCode]=@AccountCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPComputerSkillsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPComputerSkillsDeletePermanently]    Script Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
```

```
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Personal.ComputerSkills ↙
    Table

CREATE PROC [Personals].[SPComputerSkillsDeletePermanently]
(
    @Code INT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[ComputerSkills] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Personals].[ComputerSkills]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPComputerSkillsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPComputerSkillsDelete]    Script Date: 08/ ↙
    16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Personal.   ↙
    ComputerSkills Table

CREATE PROC [Personals].[SPComputerSkillsDelete]
(
    @Code INT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[ComputerSkills] WHERE (([Code] = @Code) AND ↙
    (Deleted=@Deleted)))
            BEGIN
            UPDATE [Personals].[ComputerSkills]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPComputerSkillsDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPCompaniesSelect]    Script Date: 08/16/   ↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Personal.Companies    ↙
    Table

CREATE PROC [Personals].[SPCompaniesSelect]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @Deleted BIT=NULL
```

```
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Companies] WHERE (([Code] = @Code) AND (   ↙
    [AccountCode]=@AccountCode) AND ([ScreenCode]=@ScreenCode) AND ([Deleted]=@Deleted)))
            BEGIN
            SELECT * FROM [Personals].[Companies]
            WHERE
                (
                ([Code] = @Code) AND
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Personals].[Companies]
            WHERE
                (
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPCompaniesSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPCompaniesInsertUpdate]    Script Date: 08/↙
    16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for saving values into Personal.Companies Table

CREATE PROC [Personals].[SPCompaniesInsertUpdate]
```

```
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @CompanyName NVARCHAR(256)=NULL,
    @ContactName NVARCHAR(150)=NULL,
    @ContactTitle NVARCHAR(50)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Personals].[Companies] WHERE (([Code] = @Code) AND (↙
    [AccountCode]=@AccountCode) AND ([Deleted]=@Deleted)))
            BEGIN
                INSERT INTO [Personals].[Companies]
                (
                    [AccountCode],
                    [ScreenCode],
                    [CompanyName],
                    [ContactName],
                    [ContactTitle],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    @AccountCode,
                    @ScreenCode,
                    @CompanyName,
                    @ContactName,
                    @ContactTitle,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [Personals].[Companies]
                SET
                    [AccountCode]=@AccountCode,
                    [ScreenCode]=@ScreenCode,
                    [CompanyName]=@CompanyName,
                    [ContactName]=@ContactName,
                    [ContactTitle]=@ContactTitle,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([AccountCode]=@AccountCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN
```

```
DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPCompaniesInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPCompaniesDeletePermanently]    Script Date↙
    : 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Personal.Companies Table

CREATE PROC [Personals].[SPCompaniesDeletePermanently]
(
    @Code INT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Companies] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Personals].[Companies]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPCompaniesDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPCompaniesDelete]    Script Date: 08/16/ ↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Personal. ↙
    Companies Table

CREATE PROC [Personals].[SPCompaniesDelete]
(
    @Code INT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Companies] WHERE (([Code] = @Code) AND ↙
    (Deleted=@Deleted)))
            BEGIN
            UPDATE [Personals].[Companies]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()
```

```
RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPCompaniesDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPChoicesSelect]    Script Date: 08/16/2011 ↙
    01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Personal.Choices Table

CREATE PROC [Personals].[SPChoicesSelect]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Choices] WHERE (([Code] = @Code) AND (      ↙
    [AccountCode]=@AccountCode) AND ([ScreenCode]=@ScreenCode) AND ([Deleted]=@Deleted)))
            BEGIN
            SELECT * FROM [Personals].[Choices]
            WHERE
                (
                ([Code] = @Code) AND
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Personals].[Choices]
            WHERE
                (
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
```

```
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPChoicesSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPChoicesInsertUpdate]    Script Date: 08/16↙
    /2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Personal.Choices Table

CREATE PROC [Personals].[SPChoicesInsertUpdate]
(
    @Code INT=NULL,
    @FirstChoice NVARCHAR(50)=NULL,
    @SecondChoice NVARCHAR(50)=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Personals].[Choices] WHERE (([Code] = @Code) AND (   ↙
    [AccountCode]=@AccountCode) AND ([Deleted]=@Deleted)))
            BEGIN
                INSERT INTO [Personals].[Choices]
                (
                    [FirstChoice],
                    [SecondChoice],
                    [AccountCode],
                    [ScreenCode],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    @FirstChoice,
                    @SecondChoice,
                    @AccountCode,
                    @ScreenCode,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [Personals].[Choices]
                SET
                    [FirstChoice]=@FirstChoice,
                    [SecondChoice]=@SecondChoice,
                    [AccountCode]=@AccountCode,
                    [ScreenCode]=@ScreenCode,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([AccountCode]=@AccountCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPChoicesInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPChoicesDeletePermanently]    Script Date: ↙
    08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Personal.Choices Table

CREATE PROC [Personals].[SPChoicesDeletePermanently]
(
    @Code INT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Choices] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Personals].[Choices]
            WHERE
```

```sql
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPChoicesDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPChoicesDelete]    Script Date: 08/16/2011 ↙
    01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Personal.  ↙
    Choices Table

CREATE PROC [Personals].[SPChoicesDelete]
(
    @Code INT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Choices] WHERE (([Code] = @Code) AND        ↙
    (Deleted=@Deleted)))
            BEGIN
            UPDATE [Personals].[Choices]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPChoicesDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPCertificatesSelect]    Script Date: 08/16/↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Personal.Certificates ↙
    Table

CREATE PROC [Personals].[SPCertificatesSelect]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Certificates] WHERE (([Code] = @Code) AND ( ↙
    [AccountCode]=@AccountCode) AND ([ScreenCode]=@ScreenCode) AND ([Deleted]=@Deleted)))
            BEGIN
            SELECT * FROM [Personals].[Certificates]
            WHERE
                (
                ([Code] = @Code) AND
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Personals].[Certificates]
            WHERE
                (
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPCertificatesSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPCertificatesInsertUpdate]    Script Date: ↙
    08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Personal.Certificates    ↙
    Table

CREATE PROC [Personals].[SPCertificatesInsertUpdate]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @AwardingBody NVARCHAR(50)=NULL,
    @YearObtained NVARCHAR(4)=NULL,
    @Title NVARCHAR(256)=NULL,
    @Certificate NVARCHAR(50)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
```

```
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Personals].[Certificates] WHERE (([Code] = @Code)   ↙
    AND ([AccountCode]=@AccountCode) AND ([Deleted]=@Deleted)))
            BEGIN
                INSERT INTO [Personals].[Certificates]
                (
                    [AccountCode],
                    [ScreenCode],
                    [AwardingBody],
                    [YearObtained],
                    [Title],
                    [Certificate],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    @AccountCode,
                    @ScreenCode,
                    @AwardingBody,
                    @YearObtained,
                    @Title,
                    @Certificate,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [Personals].[Certificates]
                SET
                    [AccountCode]=@AccountCode,
                    [ScreenCode]=@ScreenCode,
                    [AwardingBody]=@AwardingBody,
                    [YearObtained]=@YearObtained,
                    [Title]=@Title,
                    [Certificate]=@Certificate,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([AccountCode]=@AccountCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()
```

```
RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPCertificatesInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPCertificatesDeletePermanently]    Script   ↙
    Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Personal.Certificates   ↙
    Table

CREATE PROC [Personals].[SPCertificatesDeletePermanently]
(
    @Code INT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Certificates] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Personals].[Certificates]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPCertificatesDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPCertificatesDelete]    Script Date: 08/16/↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
```

```
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Personal.   ↙
    Certificates Table

CREATE PROC [Personals].[SPCertificatesDelete]
(
    @Code INT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Certificates] WHERE (([Code] = @Code) AND   ↙
    (Deleted=@Deleted)))
            BEGIN
            UPDATE [Personals].[Certificates]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPCertificatesDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPBioDatasSelect]    Script Date: 08/16/2011↙
     01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
```

```sql
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for retrieving values from Personal.BioDatas     ↙
    Table

CREATE PROC [Personals].[SPBioDatasSelect]
(
    @Code NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[BioDatas] WHERE (([Code] = @Code) AND (      ↙
    [ScreenCode]=@ScreenCode) AND ([Deleted]=@Deleted)))
            BEGIN
            SELECT * FROM [Personals].[BioDatas]
            WHERE
                (
                ([Code] = @Code) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Personals].[BioDatas]
            WHERE
                (
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPBioDatasSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPBioDatasInsertUpdate]    Script Date: 08/ ↙
    16/2011 01:24:28 ******/
```

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Personals.BioDatas Table

CREATE PROC [Personals].[SPBioDatasInsertUpdate]
(
    @Code NVARCHAR(50)=NULL,
    @Surname NVARCHAR(50) = NULL,
    @MiddleName NVARCHAR(50)=NULL,
    @FirstName NVARCHAR(50)=NULL,
    @GenderCode NVARCHAR(50)=NULL,
    @CivilStatus NVARCHAR(50)=NULL,
    @DateOfBirth DATETIME=NULL,
    @CountryCode NVARCHAR(50)=NULL,
    @StateCode NVARCHAR(50)=NULL,
    @LGACode NVARCHAR(50)=NULL,
    @PlaceOfBirth NVARCHAR(50)=NULL,
    @Height NVARCHAR(50)=NULL,
    @Weight NVARCHAR(50)=NULL,
    @HealthStatusCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50)=NULL,
    @PositionInFamily INT=NULL,
    @NoOfChildren INT=NULL,
    @TicketCode NUMERIC(18,0)=NULL,
    @PinCode NUMERIC(18,0)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Personals].[BioDatas] WHERE (([Code] = @Code) AND ( ↙
    [ScreenCode]=@ScreenCode) AND ([Deleted]=@Deleted)))
            BEGIN
                INSERT INTO [Personals].[BioDatas]
                (
                    [Code],
                    [Surname],
                    [MiddleName],
                    [FirstName],
                    [GenderCode],
                    [CivilStatus],
                    [DateOfBirth],
                    [CountryCode],
                    [StateCode],
                    [LGACode],
                    [PlaceOfBirth],
                    [Height],
                    [Weight],
                    [HealthStatusCode],
                    [ScreenCode],
                    [PositionInFamily],
                    [NoOfChildren],
                    [TicketCode],
                    [PinCode],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    @Code,
                    @Surname,
                    @MiddleName,
                    @FirstName,
                    @GenderCode,
                    @CivilStatus,
                    @DateOfBirth,
                    @CountryCode,
                    @StateCode,
                    @LGACode,
                    @PlaceOfBirth,
                    @Height,
                    @Weight,
                    @HealthStatusCode,
                    @ScreenCode,
                    @PositionInFamily,
                    @NoOfChildren,
                    @TicketCode,
                    @PinCode,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [Personals].[BioDatas]
                SET
                    [Surname]=@Surname,
                    [MiddleName]=@MiddleName,
                    [FirstName]=@FirstName,
                    [GenderCode]=@GenderCode,
                    [CivilStatus]=@CivilStatus,
                    [DateOfBirth]=@DateOfBirth,
                    [CountryCode]=@CountryCode,
                    [StateCode]=@StateCode,
                    [LGACode]=@LGACode,
                    [PlaceOfBirth]=@PlaceOfBirth,
                    [Height]=@Height,
                    [Weight]=@Weight,
                    [HealthStatusCode]=@HealthStatusCode,
                    [ScreenCode]=@ScreenCode,
                    [PositionInFamily]=@PositionInFamily,
                    [NoOfChildren]=@NoOfChildren,
                    [TicketCode]=@TicketCode,
                    [PinCode]=@PinCode,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([ScreenCode]=@ScreenCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPBioDatasInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPBioDatasDeletePermanently]    Script Date:↙
     08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Personal.BioDatas Table

CREATE PROC [Personals].[SPBioDatasDeletePermanently]
(
    @Code NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[BioDatas] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Personals].[BioDatas]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()
```

```
RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPBioDatasDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPBioDatasDelete]    Script Date: 08/16/2011↙
     01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Personal.   ↙
    BioDatas Table

CREATE PROC [Personals].[SPBioDatasDelete]
(
    @Code NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[BioDatas] WHERE (([Code] = @Code) AND       ↙
    (Deleted=@Deleted)))
            BEGIN
            UPDATE [Personals].[BioDatas]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()
```

```
RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPBioDatasDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPAddressesSelect]    Script Date: 08/16/   ↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Personal.Addresses    ↙
    Table

CREATE PROC [Personals].[SPAddressesSelect]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Addresses] WHERE (([Code] = @Code) AND (    ↙
    [AccountCode]=@AccountCode) AND ([ScreenCode]=@ScreenCode) AND ([Deleted]=@Deleted)))
            BEGIN
            SELECT * FROM [Personals].[Addresses]
            WHERE
                (
                ([Code] = @Code) AND
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Personals].[Addresses]
            WHERE
                (
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
```

```sql
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPAddressesSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPAddressesInsertUpdate]    Script Date: 08/↙
    16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Personal.Addresses Table

CREATE PROC [Personals].[SPAddressesInsertUpdate]
(
    @Code INT=NULL,
    @AddressTypeCode NVARCHAR(50)=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @Description NVARCHAR(256)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @CountryCode NVARCHAR(50)=NULL,
    @StateCode NVARCHAR(50)=NULL,
    @LgCode NVARCHAR(50)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Personals].[Addresses] WHERE (([Code] = @Code) AND (↙
    [AccountCode]=@AccountCode) AND ([Deleted]=@Deleted)))
            BEGIN
                INSERT INTO [Personals].[Addresses]
                (
                    [AddressTypeCode],
                    [AccountCode],
                    [Description],
                    [ScreenCode],
                    [CountryCode],
                    [StateCode],
                    [LgCode],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    @AddressTypeCode,
                    @AccountCode,
                    @Description,
                    @ScreenCode,
                    @CountryCode,
                    @StateCode,
                    @LgCode,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [Personals].[Addresses]
                SET
                    [AddressTypeCode]=@AddressTypeCode,
                    [AccountCode]=@AccountCode,
                    [Description]=@Description,
                    [ScreenCode]=@ScreenCode,
                    [CountryCode]=@CountryCode,
                    [StateCode]=@StateCode,
                    [LgCode]=@LgCode,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([AccountCode]=@AccountCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPAddressesInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPAddressesDeletePermanently]    Script Date↙
    : 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for deleting values from Personal.Addresses Table

CREATE PROC [Personals].[SPAddressesDeletePermanently]
```

```
(
    @Code INT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Addresses] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Personals].[Addresses]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPAddressesDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPAddressesDelete]    Script Date: 08/16/ ↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Personal. ↙
    Addresses Table

CREATE PROC [Personals].[SPAddressesDelete]
(
    @Code INT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;
```

```
        IF EXISTS (SELECT * FROM [Personals].[Addresses] WHERE (([Code] = @Code) AND ↙
    (Deleted=@Deleted)))
            BEGIN
            UPDATE [Personals].[Addresses]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPAddressesDelete] TO PUBLIC
GO
/****** Object:  Table [SetUp].[Modules]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [SetUp].[Modules](
    [Code] [bigint] IDENTITY(1,1) NOT NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [Description] [nvarchar](256) NULL,
    [Url] [nvarchar](256) NULL,
    [Notes] [nvarchar](500) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Modules] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF, ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
SET IDENTITY_INSERT [SetUp].[Modules] ON
```

```
INSERT [SetUp].[Modules] ([Code], [UniversityCode], [Description], [Url], [Notes],       ↙
    [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn],        ↙
    [DeletedBy]) VALUES (1, N'ACU', N'Academics', N'http://127.0.0.1/UniversityPortalWeb/ ↙
    Academics', N'Academics Module', CAST(0x0700000000008C340B AS DateTime2), N'Ademola', ↙
    NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Modules] ([Code], [UniversityCode], [Description], [Url], [Notes],       ↙
    [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn],        ↙
    [DeletedBy]) VALUES (3, N'ACU', N'SetUp', N'http://127.0.0.1/UniversityPortalWeb/SetUp↙
    ', N'SetUp Module', CAST(0x079528BF74948C340B AS DateTime2), N'Ademola', NULL, NULL, 0↙
    , NULL, NULL)
INSERT [SetUp].[Modules] ([Code], [UniversityCode], [Description], [Url], [Notes],       ↙
    [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn],        ↙
    [DeletedBy]) VALUES (4, N'ACU', N'Assessment', N'http://127.0.0.1/UniversityPortalWeb/↙
    Assessment', N'Assessment Module', CAST(0x071BE2E0F2958C340B AS DateTime2), N'Ademola'↙
    , NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Modules] ([Code], [UniversityCode], [Description], [Url], [Notes],       ↙
    [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn],        ↙
    [DeletedBy]) VALUES (5, N'ACU', N'Registry', N'http://127.0.0.1/UniversityPortalWeb/  ↙
    Registry', N'Registry Module', CAST(0x0782B1A18A968C340B AS DateTime2), N'Ademola',   ↙
    NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Modules] ([Code], [UniversityCode], [Description], [Url], [Notes],       ↙
    [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn],        ↙
    [DeletedBy]) VALUES (6, N'ACU', N'Finance', N'http://127.0.0.1/UniversityPortalWeb/   ↙
    Finance', N'Finance Module', CAST(0x07611375A1968C340B AS DateTime2), N'Ademola', NULL↙
    , NULL, 0, NULL, NULL)
INSERT [SetUp].[Modules] ([Code], [UniversityCode], [Description], [Url], [Notes],       ↙
    [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn],        ↙
    [DeletedBy]) VALUES (7, N'ACU', N'Personals', N'http://127.0.0.1/UniversityPortalWeb/ ↙
    Personals', N'Personals Module', CAST(0x0783EAF8C1968C340B AS DateTime2), N'Ademola', ↙
    NULL, NULL, 0, NULL, NULL)
SET IDENTITY_INSERT [SetUp].[Modules] OFF
/****** Object:  Table [Registry].[Exams]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Registry].[Exams](
    [Code] [bigint] IDENTITY(1,1) NOT NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [TypeCode] [nvarchar](50) NULL,
    [AccountCode] [nvarchar](50) NULL,
    [FacultyCode] [bigint] NULL,
    [DepartmentCode] [bigint] NULL,
    [CourseCode] [bigint] NULL,
    [ProgramCode] [bigint] NULL,
    [AwardCode] [nvarchar](50) NULL,
    [EntryMode] [nvarchar](50) NULL,
    [StudyMode] [nvarchar](50) NULL,
    [LevelCode] [int] NULL,
    [DateApplied] [datetime2](7) NULL,
    [SessionCode] [bigint] NULL,
    [Notes] [nvarchar](256) NULL,
    [NoOfSitting] [int] NULL,
    [NoOfCredits] [int] NULL,
    [NoOfPasses] [int] NULL,
    [NoOfDistinctions] [int] NULL,
    [NoOfFails] [int] NULL,
    [Total] [int] NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_QualifiedForExams] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,          ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Finance].[FeeDefinition]    Script Date: 08/16/2011 01:24:32 *****↙
    */
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Finance].[FeeDefinition](
    [Code] [nvarchar](50) NOT NULL,
    [Description] [nvarchar](256) NULL,
    [FeeStatusCode] [nvarchar](50) NULL,
    [UniversityCode] [nvarchar](50) NULL,
 CONSTRAINT [PK_FeeDefinition] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,          ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [SetUp].[GradingSystem]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [SetUp].[GradingSystem](
    [Code] [bigint] NOT NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [ScoreLowerBound] [decimal](18, 2) NULL,
    [ScoreUpperBound] [decimal](18, 2) NULL,
    [Description] [nvarchar](2) NULL,
    [CGPALowerBound] [decimal](18, 2) NULL,
    [CGPAUpperBound] [decimal](18, 2) NULL,
    [Notes] [nvarchar](500) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_GradingSystem] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,          ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Finance].[AccountTypes]    Script Date: 08/16/2011 01:24:32 ******↙
    /
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Finance].[AccountTypes](
    [Code] [decimal](18, 0) NOT NULL,
    [AccountSubCode] [nvarchar](50) NULL,
    [DescriptionCode] [nvarchar](50) NULL,
    [Description] [nvarchar](256) NULL,
    [UniversityCode] [nvarchar](50) NULL,
 CONSTRAINT [PK_AccountTypes] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,          ↙
```

```sql
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Finance].[AccountSub]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Finance].[AccountSub](
    [Code] [int] IDENTITY(1,1) NOT NULL,
    [DescriptionCode] [nvarchar](50) NULL,
    [ParameterCode] [nvarchar](50) NULL,
    [Description] [nvarchar](256) NULL,
    [UniversityCode] [nvarchar](50) NULL,
 CONSTRAINT [PK_AccountSub] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,      ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Finance].[Accounts]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Finance].[Accounts](
    [Code] [decimal](18, 0) NOT NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [Description] [nvarchar](256) NULL,
    [AccountTypeCode] [decimal](18, 0) NULL,
    [AccountSubCode] [int] NULL,
 CONSTRAINT [PK_Accouints] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,      ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [SetUp].[Approvals]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [SetUp].[Approvals](
    [Code] [uniqueidentifier] NOT NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [FacultyCode] [bigint] NULL,
    [DepartmentCode] [bigint] NULL,
    [CourseCode] [bigint] NULL,
    [ProgramCode] [int] NULL,
    [SessionCode] [nvarchar](50) NULL,
    [ModuleCode] [nvarchar](50) NULL,
    [ScreenCode] [nvarchar](50) NULL,
    [SemesterCode] [nvarchar](50) NULL,
    [LevelCode] [int] NULL,
    [RoleCode] [uniqueidentifier] NULL,
    [AccountCode] [nvarchar](50) NULL,
    [EntityCode] [nvarchar](50) NULL,
    [ApprovalType] [nvarchar](50) NULL,
    [Order] [int] NULL,
    [Approved] [bit] NULL,
    [ApprovedOn] [datetime] NULL,
    [ApprovedBy] [nvarchar](50) NULL,
    [ApprovedNotes] [nvarchar](1000) NULL,
    [Requested] [bit] NULL,
    [RequestedOn] [datetime] NULL,
    [RequestedBy] [nvarchar](50) NULL,
    [RequestedNotes] [nvarchar](1000) NULL,
    [Rejected] [bit] NULL,
    [RejectedOn] [datetime] NULL,
    [RejectedBy] [nvarchar](50) NULL,
    [RejectedNotes] [nvarchar](1000) NULL,
 CONSTRAINT [PK_Approvals] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,           ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[aspnet_Paths]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[aspnet_Paths](
    [ApplicationId] [uniqueidentifier] NOT NULL,
    [PathId] [uniqueidentifier] NOT NULL,
    [Path] [nvarchar](256) NOT NULL,
    [LoweredPath] [nvarchar](256) NOT NULL,
PRIMARY KEY NONCLUSTERED
(
    [PathId] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,           ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  StoredProcedure [dbo].[aspnet_Personalization_GetApplicationId]    Script↙
     Date: 08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_Personalization_GetApplicationId] (
    @ApplicationName NVARCHAR(256),
    @ApplicationId UNIQUEIDENTIFIER OUT)
AS
BEGIN
    SELECT @ApplicationId = ApplicationId FROM dbo.aspnet_Applications WHERE LOWER(@      ↙
    ApplicationName) = LoweredApplicationName
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_CheckSchemaVersion]    Script Date: 08/16/ ↙
    2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_CheckSchemaVersion]
    @Feature                   nvarchar(128),
    @CompatibleSchemaVersion   nvarchar(128)
AS
BEGIN
    IF (EXISTS( SELECT  *
                FROM    dbo.aspnet_SchemaVersions
                WHERE   Feature = LOWER( @Feature ) AND
                        CompatibleSchemaVersion = @CompatibleSchemaVersion ))
        RETURN 0

    RETURN 1
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_Applications_CreateApplication]    Script   ↙
    Date: 08/16/2011 01:24:27 ******/
```

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_Applications_CreateApplication]
    @ApplicationName      nvarchar(256),
    @ApplicationId        uniqueidentifier OUTPUT
AS
BEGIN
    SELECT  @ApplicationId = ApplicationId FROM dbo.aspnet_Applications WHERE LOWER(@     ↙
    ApplicationName) = LoweredApplicationName

    IF(@ApplicationId IS NULL)
    BEGIN
        DECLARE @TranStarted   bit
        SET @TranStarted = 0

        IF( @@TRANCOUNT = 0 )
        BEGIN
            BEGIN TRANSACTION
            SET @TranStarted = 1
        END
        ELSE
            SET @TranStarted = 0

        SELECT  @ApplicationId = ApplicationId
        FROM dbo.aspnet_Applications WITH (UPDLOCK, HOLDLOCK)
        WHERE LOWER(@ApplicationName) = LoweredApplicationName

        IF(@ApplicationId IS NULL)
        BEGIN
            SELECT  @ApplicationId = NEWID()
            INSERT  dbo.aspnet_Applications (ApplicationId, ApplicationName,              ↙
    LoweredApplicationName)
            VALUES  (@ApplicationId, @ApplicationName, LOWER(@ApplicationName))
        END


        IF( @TranStarted = 1 )
        BEGIN
            IF(@@ERROR = 0)
            BEGIN
            SET @TranStarted = 0
            COMMIT TRANSACTION
            END
            ELSE
            BEGIN
                SET @TranStarted = 0
                ROLLBACK TRANSACTION
            END
        END
    END
END
GO
/****** Object:  Table [dbo].[aspnet_Users]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[aspnet_Users](
    [ApplicationId] [uniqueidentifier] NOT NULL,
    [UserId] [uniqueidentifier] NOT NULL,
    [UserName] [nvarchar](256) NOT NULL,
    [LoweredUserName] [nvarchar](256) NOT NULL,
    [MobileAlias] [nvarchar](16) NULL,
    [IsAnonymous] [bit] NOT NULL,
    [LastActivityDate] [datetime] NOT NULL,
PRIMARY KEY NONCLUSTERED
(
    [UserId] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,     ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
INSERT [dbo].[aspnet_Users] ([ApplicationId], [UserId], [UserName], [LoweredUserName],  ↙
    [MobileAlias], [IsAnonymous], [LastActivityDate]) VALUES (N'3436a128-b3cb-4d6a-a7f2-  ↙
    782b22a6d04d', N'b6a16061-1f52-40b4-a680-be68f8df7d26', N'chiriksmat', N'chiriksmat', ↙
    NULL, 0, CAST(0x00009EEC016CD544 AS DateTime))
/****** Object:  StoredProcedure [dbo].[aspnet_UnRegisterSchemaVersion]    Script Date: 08↙
    /16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_UnRegisterSchemaVersion]
    @Feature                   nvarchar(128),
    @CompatibleSchemaVersion   nvarchar(128)
AS
BEGIN
    DELETE FROM dbo.aspnet_SchemaVersions
        WHERE   Feature = LOWER(@Feature) AND @CompatibleSchemaVersion =                ↙
    CompatibleSchemaVersion
END
GO
/****** Object:  Table [dbo].[aspnet_Roles]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[aspnet_Roles](
    [ApplicationId] [uniqueidentifier] NOT NULL,
    [RoleId] [uniqueidentifier] NOT NULL,
    [RoleName] [nvarchar](256) NOT NULL,
    [LoweredRoleName] [nvarchar](256) NOT NULL,
    [Description] [nvarchar](256) NULL,
PRIMARY KEY NONCLUSTERED
(
    [RoleId] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,     ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  StoredProcedure [dbo].[aspnet_RegisterSchemaVersion]    Script Date: 08/ ↙
    16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_RegisterSchemaVersion]
    @Feature                   nvarchar(128),
    @CompatibleSchemaVersion   nvarchar(128),
    @IsCurrentVersion          bit,
    @RemoveIncompatibleSchema  bit
AS
BEGIN
    IF( @RemoveIncompatibleSchema = 1 )
    BEGIN
        DELETE FROM dbo.aspnet_SchemaVersions WHERE Feature = LOWER( @Feature )
    END
    ELSE
    BEGIN
        IF( @IsCurrentVersion = 1 )
        BEGIN
            UPDATE dbo.aspnet_SchemaVersions
            SET IsCurrentVersion = 0
            WHERE Feature = LOWER( @Feature )
        END
    END

    INSERT  dbo.aspnet_SchemaVersions( Feature, CompatibleSchemaVersion, IsCurrentVersion ↙
     )
    VALUES( LOWER( @Feature ), @CompatibleSchemaVersion, @IsCurrentVersion )
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_WebEvent_LogEvent]    Script Date: 08/16/ ↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_WebEvent_LogEvent]
        @EventId         char(32),
        @EventTimeUtc    datetime,
        @EventTime       datetime,
        @EventType       nvarchar(256),
        @EventSequence   decimal(19,0),
        @EventOccurrence decimal(19,0),
        @EventCode       int,
        @EventDetailCode int,
        @Message         nvarchar(1024),
        @ApplicationPath nvarchar(256),
        @ApplicationVirtualPath nvarchar(256),
        @MachineName    nvarchar(256),
        @RequestUrl      nvarchar(1024),
        @ExceptionType   nvarchar(256),
        @Details         ntext
AS
BEGIN
    INSERT
        dbo.aspnet_WebEvent_Events
        (
            EventId,
            EventTimeUtc,
            EventTime,
            EventType,
            EventSequence,
            EventOccurrence,
            EventCode,
            EventDetailCode,
            Message,
            ApplicationPath,
            ApplicationVirtualPath,
            MachineName,
            RequestUrl,
            ExceptionType,
            Details
        )
    VALUES
    (
        @EventId,
        @EventTimeUtc,
        @EventTime,
        @EventType,
        @EventSequence,
        @EventOccurrence,
        @EventCode,
        @EventDetailCode,
        @Message,
        @ApplicationPath,
        @ApplicationVirtualPath,
        @MachineName,
        @RequestUrl,
        @ExceptionType,
        @Details
    )
END
GO
/****** Object:  Table [SetUp].[SubjectRequirementDetails]    Script Date: 08/16/2011 01: ↙
    24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [SetUp].[SubjectRequirementDetails](
    [Code] [bigint] IDENTITY(1,1) NOT NULL,
    [SubjectRequirementCode] [bigint] NULL,
    [GradeCode] [nvarchar](50) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_SubjectRequirementDetails] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,          ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  View [dbo].[vw_aspnet_Applications]    Script Date: 08/16/2011 01:24:34 *↙
    *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE VIEW [dbo].[vw_aspnet_Applications]
  AS SELECT [dbo].[aspnet_Applications].[ApplicationName], [dbo].[aspnet_Applications].   ↙
    [LoweredApplicationName], [dbo].[aspnet_Applications].[ApplicationId], [dbo].        ↙
    [aspnet_Applications].[Description]
  FROM [dbo].[aspnet_Applications]
GO
/****** Object:  Table [Registry].[Students]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Registry].[Students](
    [Code] [bigint] IDENTITY(1,1) NOT NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [FacultyCode] [bigint] NULL,
    [DepartmentCode] [bigint] NULL,
    [ProgramCode] [bigint] NULL,
    [CourseCode] [bigint] NULL,
    [LevelCode] [int] NULL,
    [MatricNo] [nvarchar](50) NOT NULL,
    [AccountCode] [nvarchar](50) NULL,
    [EntryMode] [nvarchar](50) NULL,
    [StudyMode] [nvarchar](50) NULL,
    [SessionCode] [bigint] NULL,
    [AdmittedOn] [date] NULL,
    [AwardCode] [nvarchar](50) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Students] PRIMARY KEY CLUSTERED
(
    [Code] ASC,
    [MatricNo] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,           ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [SetUp].[States]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [SetUp].[States](
    [Code] [nvarchar](50) NOT NULL,
    [Name] [nvarchar](256) NOT NULL,
    [CountriesCode] [nvarchar](50) NOT NULL,
 CONSTRAINT [PK_States] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,           ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
INSERT [SetUp].[States] ([Code], [Name], [CountriesCode]) VALUES (N'AB', N'Abia', N'NGN')
INSERT [SetUp].[States] ([Code], [Name], [CountriesCode]) VALUES (N'AD', N'Adamawa', N'NGN↙
    ')
INSERT [SetUp].[States] ([Code], [Name], [CountriesCode]) VALUES (N'CR', N'Cross River', N↙
    'NGN')
INSERT [SetUp].[States] ([Code], [Name], [CountriesCode]) VALUES (N'CT', N'Cape Town', N  ↙
    'RSA')
INSERT [SetUp].[States] ([Code], [Name], [CountriesCode]) VALUES (N'JB', N'Johannesburg', ↙
    N'RSA')
INSERT [SetUp].[States] ([Code], [Name], [CountriesCode]) VALUES (N'LG', N'Lagos', N'NGN')
INSERT [SetUp].[States] ([Code], [Name], [CountriesCode]) VALUES (N'OD', N'Ondo', N'NGN')
INSERT [SetUp].[States] ([Code], [Name], [CountriesCode]) VALUES (N'OY', N'Oyo', N'NGN')
INSERT [SetUp].[States] ([Code], [Name], [CountriesCode]) VALUES (N'TR', N'Taraba State', ↙
    N'NGN')
/****** Object:  Table [SetUp].[Staff]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [SetUp].[Staff](
    [Code] [nvarchar](50) NOT NULL,
    [AccountCode] [nvarchar](50) NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [FacultyCode] [bigint] NULL,
    [DepartmentCode] [bigint] NULL,
    [StaffType] [nvarchar](50) NULL,
    [ContractType] [nvarchar](50) NULL,
    [ComDate] [datetime2](7) NULL,
    [CessDate] [datetime2](7) NULL,
    [StatusCode] [nvarchar](50) NULL,
    [StatusReason] [nvarchar](500) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Staff] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,       ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  StoredProcedure [Finance].[SPTransactionsDetailSelect]    Script Date: 08↙
    /16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Finance.             ↙
    TransactionsDetail Table

CREATE PROC [Finance].[SPTransactionsDetailSelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Deleted BIT=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance].[TransactionsDetail] WHERE (([Code] = @Code)   ↙
    AND ([UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted)))
            BEGIN
            SELECT * FROM [Finance].[TransactionsDetail]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Finance].[TransactionsDetail]
            WHERE
                (
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPTransactionsDetailSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPTransactionsDetailInsertUpdate]    Script   ↙
    Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Finance.TransactionsDetail↙
     Table

CREATE PROC [Finance].[SPTransactionsDetailInsertUpdate]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @SessionCode BIGINT=NULL,
    @SemesterCode NVARCHAR(50)=NULL,
    @InvoiceCode DECIMAL(18,0)=NULL,
    @ItemDescription NVARCHAR(500)=NULL,
    @Quantity INT=NULL,
    @Unit NVARCHAR(50)=NULL,
    @UnitCost DECIMAL(18,2)=NULL,
    @DiscountPercentage DECIMAL(18,2)=NULL,
    @DiscountAmount DECIMAL(18,2)=NULL,
    @TaxType NVARCHAR(50)=NULL,
    @TaxValue DECIMAL(18,2)=NULL,
    @TotalCost DECIMAL(18,2)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Finance] .[TransactionsDetail] WHERE ([Code] = @    ↙
    Code) AND ([UniversityCode] = @UniversityCode))
            BEGIN
                INSERT INTO [Finance] .[TransactionsDetail]
                (
                    [UniversityCode],
                    [SessionCode],
                    [SemesterCode],
                    [InvoiceCode],
                    [ItemDescription],
                    [Quantity],
                    [Unit],
                    [UnitCost],
                    [DiscountPercentage],
                    [DiscountAmount],
                    [TaxType],
                    [TaxValue],
                    [TotalCost],
                    [CreatedOn],
                    [CreatedBy]

                )
                VALUES
                (
                    @UniversityCode,
                    @SessionCode,
                    @SemesterCode,
                    @InvoiceCode,
                    @ItemDescription,
                    @Quantity,
                    @Unit,
                    @UnitCost,
                    @DiscountPercentage,
                    @DiscountAmount,
                    @TaxType,
                    @TaxValue,
                    @TotalCost,
                    @CreatedOn,
                    @CreatedBy

                )
            END
        ELSE
            BEGIN
                UPDATE [Finance] .[TransactionsDetail]
                SET
                    [UniversityCode]=@UniversityCode,
                    [SessionCode]=@SessionCode,
                    [SemesterCode]=@SemesterCode,
                    [InvoiceCode]=@InvoiceCode,
                    [ItemDescription]=@ItemDescription,
                    [Quantity]=@Quantity,
                    [Unit]=@Unit,
                    [UnitCost]=@UnitCost,
                    [DiscountPercentage]=@DiscountPercentage,
                    [DiscountAmount]=@DiscountAmount,
                    [TaxType]=@TaxType,
                    [TaxValue]=@TaxValue,
                    [TotalCost]=@TotalCost,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy

                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()
```

```
RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPTransactionsDetailsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPTransactionsDetailDeletePermanently]         ↙
    Script Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Finance.                ↙
    TransactionsDetail Table

CREATE PROC [Finance].[SPTransactionsDetailDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance].[TransactionsDetail] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Finance].[TransactionsDetail]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPTransactionsDetailDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPTransactionsDetailDelete]    Script Date: 08↙
    /16/2011 01:24:28 ******/
SET ANSI_NULLS ON
```

```
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/28/2011
--Last Updated        :   07/28/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for deleting values temporarily from Finance.    ↙
    TransactionsDetail Table

CREATE PROC [Finance].[SPTransactionsDetailDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance].[TransactionsDetail] WHERE ([Code] = @Code) AND↙
     (Deleted=@Deleted))
            BEGIN
            UPDATE [Finance].[TransactionsDetail]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPTransactionsDetailDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPUniversitiesSelect]    Script Date: 08/16/2011↙
     01:24:29 ******/
SET ANSI_NULLS ON
GO
```

```
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from SetUp.Universities    ↙
    Table

CREATE PROC [SetUp].[SPUniversitiesSelect]
(
    @Code NVARCHAR(50)=NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Universities] WHERE ([Code] = @Code)AND (        ↙
    [Deleted] = @Deleted))
            BEGIN
            SELECT * FROM [SetUp].[Universities]
            WHERE
                (
                ([Code] = @Code) AND
                ([Deleted] = @Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [SetUp].[Universities]
            WHERE
                (
                ([Deleted] = @Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPUniversitiesSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPUniversitiesInsertUpdate]    Script Date: 08/ ↙
    16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
```

```
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into SetUp.Universities Table

CREATE PROC [SetUp].[SPUniversitiesInsertUpdate]
(
    @Code NVARCHAR(50)=NULL,
    @Description NVARCHAR(256)=NULL,
    @UniversityTypeCode NVARCHAR(50)=NULL,
    @BannerCode NVARCHAR(50)=NULL,
    @LogoCode NVARCHAR(50)=NULL,
    @Url NVARCHAR(256)=NULL,
    @CountryCode NVARCHAR(50)=NULL,
    @StateCode NVARCHAR(50)=NULL,
    @LgaCode NVARCHAR(50)=NULL,
    @Motto NVARCHAR(50)=NULL,
    @Notes NVARCHAR(MAX)=NULL,
    @EstablishedYear INT=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[Universities]  WHERE ([Code] = @Code) AND ( ↙
    [Deleted] = @Deleted))
            BEGIN
                INSERT INTO [SetUp].[Universities]
                (
                    [Code],
                    [Description],
                    [UniversityTypeCode],
                    [BannerCode],
                    [LogoCode],
                    [Url],
                    [CountryCode],
                    [StateCode],
                    [LgaCode],
                    [Motto],
                    [Notes],
                    [EstablishedYear],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]

                )
                VALUES
                (
                    @Code,
                    @Description,
                    @UniversityTypeCode,
                    @BannerCode,
                    @LogoCode,
                    @Url,
                    @CountryCode,
                    @StateCode,
                    @LgaCode,
                    @Motto,
                    @Notes,
                    @EstablishedYear,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [SetUp].[Universities]
                SET
                    [Description]=@Description,
                    [UniversityTypeCode]=@UniversityTypeCode,
                    [BannerCode]=@BannerCode,
                    [LogoCode]=@LogoCode,
                    [Url]=@Url,
                    [CountryCode]=@CountryCode,
                    [StateCode]=@StateCode,
                    [LgaCode]=@LgaCode,
                    [Motto]=@Motto,
                    [Notes]=@Notes,
                    [EstablishedYear]=@EstablishedYear,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    ([Code] = @Code)
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPUniversitiesInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPUniversitiesDeletePermanently]    Script Date:↙
     08/16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/12/2011
--Last Updated      :   07/12/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from SetUp.Universities Table
```

```
CREATE PROC [SetUp].[SPUniversitiesDeletePermanently]
(
    @Code NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Universities] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [SetUp].[Universities]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPUniversitiesDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPUniversitiesDelete]    Script Date: 08/16/2011↙
     01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from SetUp.     ↙
    Universities Table

CREATE PROC [SetUp].[SPUniversitiesDelete]
(
    @Code NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[Universities] WHERE (([Code] = @Code) AND   ↙
    (Deleted=@Deleted)))
            BEGIN
            UPDATE [SetUp].[Universities]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPUniversitiesDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPRefereesSelect]    Script Date: 08/16/2011↙
     01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Personal.Referees     ↙
    Table

CREATE PROC [Personals].[SPRefereesSelect]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
```

```
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Referees] WHERE (([Code] = @Code) AND (    ↙
    [AccountCode]=@AccountCode) AND ([ScreenCode]=@ScreenCode) AND ([Deleted]=@Deleted)))
            BEGIN
            SELECT * FROM [Personals].[Referees]
            WHERE
                (
                ([Code] = @Code) AND
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Personals].[Referees]
            WHERE
                (
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPRefereesSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPRefereesInsertUpdate]    Script Date: 08/ ↙
    16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for saving values into Personals.Referees Table

CREATE PROC [Personals].[SPRefereesInsertUpdate]
(
    @Code INT=NULL,
    @TitleCode NVARCHAR(50)=NULL,
    @Name NVARCHAR(150)=NULL,
    @OccupationCode NVARCHAR(50)=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50)=NULL,
    @KnowingMode NVARCHAR(256)=NULL,
    @ContactReferenceCode NVARCHAR(50)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Personals].[Referees] WHERE (([Code] = @Code) AND ( ↙
    [AccountCode]=@AccountCode) AND ([Deleted]=@Deleted)))
            BEGIN
                INSERT INTO [Personals].[Referees]
                (
                    [TitleCode],
                    [Name],
                    [OccupationCode],
                    [AccountCode],
                    [ScreenCode],
                    [KnowingMode],
                    [ContactReferenceCode],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    @TitleCode,
                    @Name,
                    @OccupationCode,
                    @AccountCode,
                    @ScreenCode,
                    @KnowingMode,
                    @ContactReferenceCode,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [Personals].[Referees]
                SET
                    [TitleCode]=@TitleCode,
                    [Name]=@Name,
                    [OccupationCode]=@OccupationCode,
                    [AccountCode]=@AccountCode,
                    [ScreenCode]=@ScreenCode,
                    [KnowingMode]=@KnowingMode,
                    [ContactReferenceCode]=@ContactReferenceCode,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([AccountCode]=@AccountCode))
            END
COMMIT TRAN
END TRY
```

```
BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPRefereesInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPRefereesDeletePermanently]    Script Date:↙
     08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for deleting values from Personal.Referees Table

CREATE PROC [Personals].[SPRefereesDeletePermanently]
(
    @Code INT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Referees] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Personals].[Referees]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
```

```
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPRefereesDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPRefereesDelete]    Script Date: 08/16/2011↙
     01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Personal.  ↙
    Referees Table

CREATE PROC [Personals].[SPRefereesDelete]
(
    @Code INT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Referees] WHERE (([Code] = @Code) AND      ↙
    (Deleted=@Deleted)))
            BEGIN
            UPDATE [Personals].[Referees]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPRefereesDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPSchedulesSelect]    Script Date: 08/16/2011 01↙
    :24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from SetUp.Schedules Table

CREATE PROC [SetUp].[SPSchedulesSelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Schedules] WHERE ([Code] = @Code) AND (         ↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
            SELECT * FROM [SetUp].[Schedules]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [SetUp].[Schedules]
            WHERE
                (
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);
```

```
SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPSchedulesSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPSchedulesInsertUpdate]    Script Date: 08/16/ ↙
    2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into SetUp.Schedules Table

CREATE PROC [SetUp].[SPSchedulesInsertUpdate]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @FacultyCode BIGINT=NULL,
    @DepartmentCode BIGINT=NULL,
    @CourseCode BIGINT=NULL,
    @ProgramCode BIGINT=NULL,
    @LevelCode BIGINT=NULL,
    @SemesterCode NVARCHAR(50)=NULL,
    @SubCode BIGINT=NULL,
    @ModeOfStudy NVARCHAR(50)=NULL,
    @StaffCode NVARCHAR(50)=NULL,
    @DateX DATETIME2(7)=NULL,
    @StartTime NVARCHAR(50)=NULL,
    @StopTime NVARCHAR(50)=NULL,
    @ScheduleType NVARCHAR(50)=NULL,
    @Notes NVARCHAR(500)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[Schedules]  WHERE ([Code] = @Code) AND (    ↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
                INSERT INTO [SetUp].[Schedules]
                (
                    [UniversityCode],
                    [FacultyCode],
                    [DepartmentCode],
                    [CourseCode],
                    [ProgramCode],
                    [LevelCode],
                    [SemesterCode],
                    [SubCode],
                    [ModeOfStudy],
                    [StaffCode],
                    [DateX],
                    [StartTime],
                    [StopTime],
                    [ScheduleType],
                    [Notes],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]

                )
                VALUES
                (
                    @UniversityCode,
                    @FacultyCode,
                    @DepartmentCode,
                    @CourseCode,
                    @ProgramCode,
                    @LevelCode,
                    @SemesterCode,
                    @SubCode,
                    @ModeOfStudy,
                    @StaffCode,
                    @DateX,
                    @StartTime,
                    @StopTime,
                    @ScheduleType,
                    @Notes,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [SetUp].[Schedules]
                SET
                    [UniversityCode]=@UniversityCode,
                    [FacultyCode]=@FacultyCode,
                    [DepartmentCode]=@DepartmentCode,
                    [CourseCode]=@CourseCode,
                    [ProgramCode]=@ProgramCode,
                    [LevelCode]=@LevelCode,
                    [SemesterCode]=@SemesterCode,
                    [SubCode]=@SubCode,
                    [ModeOfStudy]=@ModeOfStudy,
                    [StaffCode]=@StaffCode,
                    [DateX]=@DateX,
                    [StartTime]=@StartTime,
                    [StopTime]=@StopTime,
                    [ScheduleType]=@ScheduleType,
                    [Notes]=@Notes,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
```

```
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPSchedulesInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPSchedulesDeletePermanently]    Script Date: 08↙
    /16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author             :   Ademola Adebo
--Reviewer           :   Godwin Mathias
--Date Created       :   07/12/2011
--Last Updated       :   07/12/2011
--Last Updated By    :   Ademola Adebo
--Description        :   Stored procedure for deleting values from SetUp.Schedules Table

CREATE PROC [SetUp].[SPSchedulesDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Schedules] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [SetUp].[Schedules]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPSchedulesDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPSchedulesDelete]    Script Date: 08/16/2011 01↙
    :24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/28/2011
--Last Updated      :   07/28/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from SetUp.      ↙
    Schedules Table

CREATE PROC [SetUp].[SPSchedulesDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[Schedules] WHERE (([Code] = @Code) AND      ↙
    (Deleted=@Deleted)))
            BEGIN
            UPDATE [SetUp].[Schedules]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPSchedulesDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPSportsSelect]    Script Date: 08/16/2011  ↙
    01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author             :   Ademola Adebo
--Reviewer           :   Godwin Mathias
--Date Created       :   07/08/2011
--Last Updated       :   07/08/2011
--Last Updated By    :   Ademola Adebo
--Description        :   Stored procedure for retrieving values from Personal.Sports Table

CREATE PROC [Personals].[SPSportsSelect]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Sports] WHERE (([Code] = @Code) AND (       ↙
    [AccountCode]=@AccountCode) AND ([ScreenCode]=@ScreenCode) AND ([Deleted]=@Deleted)))
            BEGIN
            SELECT * FROM [Personals].[Sports]
            WHERE
                (
                ([Code] = @Code) AND
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Personals].[Sports]
            WHERE
                (
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);
```

```sql
SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPSportsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPSportsInsertUpdate]    Script Date: 08/16/↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Personals.Sports Table

CREATE PROC [Personals].[SPSportsInsertUpdate]
(
    @Code INT=NULL,
    @SportCode NVARCHAR(50)=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Personals].[Sports] WHERE (([Code] = @Code) AND (   ↙
    [AccountCode]=@AccountCode) AND ([Deleted]=@Deleted)))
            BEGIN
                INSERT INTO [Personals].[Sports]
                (
                    [SportCode],
                    [AccountCode],
                    [ScreenCode],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    @SportCode,
                    @AccountCode,
                    @ScreenCode,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [Personals].[Sports]
                SET
                    [SportCode]=@SportCode,
                    [AccountCode]=@AccountCode,
                    [ScreenCode]=@ScreenCode,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([AccountCode]=@AccountCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPSportsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPSportsDeletePermanently]    Script Date:  ↙
    08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Personal.Sports Table

CREATE PROC [Personals].[SPSportsDeletePermanently]
(
    @Code INT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Sports] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Personals].[Sports]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPSportsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPSportsDelete]    Script Date: 08/16/2011  ↙
    01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Personal.  ↙
    Sports Table

CREATE PROC [Personals].[SPSportsDelete]
(
    @Code INT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Sports] WHERE (([Code] = @Code) AND (Deleted↙
    =@Deleted)))
            BEGIN
            UPDATE [Personals].[Sports]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPSportsDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPSponsorsSelect]    Script Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Personal.Sponsors Table

CREATE PROC [Personals].[SPSponsorsSelect]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Sponsors] WHERE (([Code] = @Code) AND ([AccountCode]=@AccountCode) AND ([ScreenCode]=@ScreenCode) AND ([Deleted]=@Deleted)))
            BEGIN
            SELECT * FROM [Personals].[Sponsors]
            WHERE
                (
                ([Code] = @Code) AND
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Personals].[Sponsors]
            WHERE
                (
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPSponsorsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPSponsorsInsertUpdate]    Script Date: 08/ ↙
    16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Personals.Sponsors Table

CREATE PROC [Personals].[SPSponsorsInsertUpdate]
(
    @Code INT=NULL,
    @SponsorCode NVARCHAR(50)=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Personals].[Sponsors] WHERE (([Code] = @Code) AND ( ↙
    [AccountCode]=@AccountCode) AND ([Deleted]=@Deleted)))
```

```
            BEGIN
                INSERT INTO [Personals].[Sponsors]
                (
                    [SponsorCode],
                    [AccountCode],
                    [ScreenCode],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    @SponsorCode,
                    @AccountCode,
                    @ScreenCode,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [Personals].[Sponsors]
                SET
                    [SponsorCode]=@SponsorCode,
                    [AccountCode]=@AccountCode,
                    [ScreenCode]=@ScreenCode,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([AccountCode]=@AccountCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPSponsorsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPSponsorsDeletePermanently]    Script Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
```

```
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Personal.Sponsors Table

CREATE PROC [Personals].[SPSponsorsDeletePermanently]
(
    @Code INT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Sponsors] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Personals].[Sponsors]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPSponsorsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPSponsorsDelete]    Script Date: 08/16/2011↙
     01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Personal.   ↙
    Sponsors Table

CREATE PROC [Personals].[SPSponsorsDelete]
(
    @Code INT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Sponsors] WHERE (([Code] = @Code) AND     ↙
    (Deleted=@Deleted)))
            BEGIN
            UPDATE [Personals].[Sponsors]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPSponsorsDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPQualificationsSelect]    Script Date: 08/ ↙
    16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Personal.           ↙
    Qualifications Table

CREATE PROC [Personals].[SPQualificationsSelect]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @Deleted BIT=NULL
)
```

```
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Qualifications] WHERE (([Code] = @Code) AND ↙
    ([AccountCode]=@AccountCode) AND ([ScreenCode]=@ScreenCode) AND ([Deleted]=@Deleted)))
            BEGIN
            SELECT * FROM [Personals].[Qualifications]
            WHERE
                (
                ([Code] = @Code) AND
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Personals].[Qualifications]
            WHERE
                (
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPQualificationsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPQualificationsInsertUpdate]    Script Date↙
    : 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Personals.Qualifications  ↙
    Table

CREATE PROC [Personals].[SPQualificationsInsertUpdate]
```

```sql
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @EducationTypeCode NVARCHAR(50)=NULL,
    @QualificationTypeCode NVARCHAR(50)=NULL,
    @FromYear NVARCHAR(50)=NULL,
    @ToYear NVARCHAR(50)=NULL,
    @FromMonth NVARCHAR(50)=NULL,
    @ToMonth NVARCHAR(50)=NULL,
    @AwardingBody NVARCHAR(256)=NULL,
    @Certificate NVARCHAR(50)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Personals].[Qualifications] WHERE (([Code] = @Code) ↙
    AND ([AccountCode]=@AccountCode) AND ([Deleted]=@Deleted)))
            BEGIN
                INSERT INTO [Personals].[Qualifications]
                (
                    [AccountCode],
                    [ScreenCode],
                    [EducationTypeCode],
                    [QualificationTypeCode],
                    [FromYear],
                    [ToYear],
                    [FromMonth],
                    [ToMonth],
                    [AwardingBody],
                    [Certificate],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    @AccountCode,
                    @ScreenCode,
                    @EducationTypeCode,
                    @QualificationTypeCode,
                    @FromYear,
                    @ToYear,
                    @FromMonth,
                    @ToMonth,
                    @AwardingBody,
                    @Certificate,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [Personals].[Qualifications]
                SET
                    [AccountCode]=@AccountCode,
                    [ScreenCode]=@ScreenCode,
                    [EducationTypeCode]=@EducationTypeCode,
                    [QualificationTypeCode]=@QualificationTypeCode,
                    [FromYear]=@FromYear,
                    [ToYear]=@ToYear,
                    [FromMonth]=@FromMonth,
                    [ToMonth]=@ToMonth,
                    [AwardingBody]=@AwardingBody,
                    [Certificate]=@Certificate,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([AccountCode]=@AccountCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPQualificationsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPQualificationsDeletePermanently]    Script↙
     Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author                :   Ademola Adebo
--Reviewer              :   Godwin Mathias
--Date Created          :   07/08/2011
--Last Updated          :   07/08/2011
--Last Updated By       :   Ademola Adebo
--Description           :   Stored procedure for deleting values from Personal.Qualifications ↙
    Table

CREATE PROC [Personals].[SPQualificationsDeletePermanently]
(
    @Code INT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Qualifications] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Personals].[Qualifications]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPQualificationsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPQualificationsDelete]    Script Date: 08/ ↙
    16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Personal.   ↙
    Qualifications Table

CREATE PROC [Personals].[SPQualificationsDelete]
(
    @Code INT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Qualifications] WHERE (([Code] = @Code) AND ↙
    (Deleted=@Deleted)))
            BEGIN
            UPDATE [Personals].[Qualifications]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPQualificationsDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPSubjectRequirementsSelect]    Script Date: 08/↙
    16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from SetUp.         ↙
    SubjectRequirements Table

CREATE PROC [SetUp].[SPSubjectRequirementsSelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[SubjectRequirements] WHERE ([Code] = @Code) AND ↙
    ([UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
            SELECT * FROM [SetUp].[SubjectRequirements]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [SetUp].[SubjectRequirements]
            WHERE
                (
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPSubjectRequirementsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPSubjectRequirementsInsertUpdate]    Script    ↙
    Date: 08/16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into SetUp.SubjectRequirements ↙
    Table

CREATE PROC [SetUp].[SPSubjectRequirementsInsertUpdate]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @FacultyCode BIGINT=NULL,
    @DepartmentCode BIGINT=NULL,
    @CourseCode BIGINT=NULL,
    @ProgramCode BIGINT=NULL,
    @SubjectCode NVARCHAR(50)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;
```

```
        IF NOT EXISTS (SELECT * FROM [SetUp].[SubjectRequirements]  WHERE ([Code] = @Code)↙
     AND ([UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
                INSERT INTO [SetUp].[SubjectRequirements]
                (
                    [UniversityCode],
                    [FacultyCode],
                    [DepartmentCode],
                    [CourseCode],
                    [ProgramCode],
                    [SubjectCode],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]

                )
                VALUES
                (
                    @UniversityCode,
                    @FacultyCode,
                    @DepartmentCode,
                    @CourseCode,
                    @ProgramCode,
                    @SubjectCode,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [SetUp].[SubjectRequirements]
                SET
                    [UniversityCode]=@UniversityCode,
                    [FacultyCode]=@FacultyCode,
                    [DepartmentCode]=@DepartmentCode,
                    [CourseCode]=@CourseCode,
                    [ProgramCode]=@ProgramCode,
                    [SubjectCode]=@SubjectCode,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);
```

```
END CATCH

GRANT EXECUTE ON [SetUp].[SPSubjectRequirementsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPSubjectRequirementsDeletePermanently]        ↙
    Script Date: 08/16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/12/2011
--Last Updated      :   07/12/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from SetUp.                 ↙
    SubjectRequirements Table

CREATE PROC [SetUp].[SPSubjectRequirementsDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[SubjectRequirements] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [SetUp].[SubjectRequirements]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPSubjectRequirementsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPSubjectRequirementsDelete]    Script Date: 08/↙
    16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
```

```
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/28/2011
--Last Updated      :   07/28/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from SetUp.      ↙
    SubjectRequirements Table

CREATE PROC [SetUp].[SPSubjectRequirementsDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[SubjectRequirements] WHERE (([Code] = @Code)↙
     AND (Deleted=@Deleted)))
            BEGIN
            UPDATE [SetUp].[SubjectRequirements]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPSubjectRequirementsDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPProgramRequirementsSelect]    Script Date: 08/↙
    16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
```

```
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from SetUp.      ↙
    ProgramRequirements Table

CREATE PROC [SetUp].[SPProgramRequirementsSelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[ProgramRequirements] WHERE ([Code] = @Code) AND ↙
    ([UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
            SELECT * FROM [SetUp].[ProgramRequirements]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [SetUp].[ProgramRequirements]
            WHERE
                (
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPProgramRequirementsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPProgramRequirementsInsertUpdate]    Script    ↙
    Date: 08/16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
```

```
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for saving values into SetUp.ProgramRequirements ↙
    Table

CREATE PROC [SetUp].[SPProgramRequirementsInsertUpdate]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @FacultyCode BIGINT=NULL,
    @DepartmentCode BIGINT=NULL,
    @CourseCode BIGINT=NULL,
    @ProgramCode BIGINT=NULL,
    @MinimumCredit INT=NULL,
    @MaximumCredit INT=NULL,
    @TotalDuration BIGINT=NULL,
    @DurationUnit NVARCHAR(50)=NULL,
    @EntryMode NVARCHAR(50)=NULL,
    @ModeOfStudy NVARCHAR(50)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[ProgramRequirements]  WHERE ([Code] = @Code)↙
     AND ([UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
                INSERT INTO [SetUp].[ProgramRequirements]
                (
                    [UniversityCode],
                    [FacultyCode],
                    [DepartmentCode],
                    [CourseCode],
                    [ProgramCode],
                    [MinimumCredit],
                    [MaximumCredit],
                    [TotalDuration],
                    [DurationUnit],
                    [EntryMode],
                    [ModeOfStudy],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]

                )
                VALUES
                (
                    @UniversityCode,
                    @FacultyCode,
                    @DepartmentCode,
                    @CourseCode,
                    @ProgramCode,
                    @MinimumCredit,
                    @MaximumCredit,
                    @TotalDuration,
                    @DurationUnit,
                    @EntryMode,
                    @ModeOfStudy,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [SetUp].[ProgramRequirements]
                SET
                    [UniversityCode]=@UniversityCode,
                    [FacultyCode]=@FacultyCode,
                    [DepartmentCode]=@DepartmentCode,
                    [CourseCode]=@CourseCode,
                    [ProgramCode]=@ProgramCode,
                    [MinimumCredit]=@MinimumCredit,
                    [MaximumCredit]=@MaximumCredit,
                    [TotalDuration]=@TotalDuration,
                    [DurationUnit]=@DurationUnit,
                    [EntryMode]=@EntryMode,
                    [ModeOfStudy]=@ModeOfStudy,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPProgramRequirementsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPProgramRequirementsDeletePermanently]        ↙
    Script Date: 08/16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/12/2011
--Last Updated      :   07/12/2011
--Last Updated By   :   Ademola Adebo
```

```
--Description       :   Stored procedure for deleting values from SetUp.        ↙
    ProgramRequirements Table

CREATE PROC [SetUp].[SPProgramRequirementsDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[ProgramRequirements] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [SetUp].[ProgramRequirements]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPProgramRequirementsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPProgramRequirementsDelete]    Script Date: 08/↙
    16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/28/2011
--Last Updated      :   07/28/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from SetUp.    ↙
    ProgramRequirements Table

CREATE PROC [SetUp].[SPProgramRequirementsDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
```

```
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[ProgramRequirements] WHERE (([Code] = @Code)↙
     AND (Deleted=@Deleted)))
            BEGIN
            UPDATE [SetUp].[ProgramRequirements]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPProgramRequirementsDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPPhotosSelect]    Script Date: 08/16/2011  ↙
    01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Personal.Photos Table

CREATE PROC [Personals].[SPPhotosSelect]
(
    @Code INT=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Photos] WHERE (([Code] = @Code) AND (        ↙
    [AccountCode]=@AccountCode) AND ([ScreenCode]=@ScreenCode) AND ([Deleted]=@Deleted)))
            BEGIN
            SELECT * FROM [Personals].[Photos]
            WHERE
                (
                ([Code] = @Code) AND
                ([ScreenCode]=@ScreenCode) AND
                ([AccountCode]=@AccountCode) AND
                ([Deleted]=@Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Personals].[Photos]
            WHERE
                (
                ([ScreenCode]=@ScreenCode) AND
                ([AccountCode]=@AccountCode) AND
                ([Deleted]=@Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPPhotosSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPPhotosInsertUpdate]    Script Date: 08/16/↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Personal.Photos Table

CREATE PROC [Personals].[SPPhotosInsertUpdate]
(
    @Code INT=NULL,
    @ModuleCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50)=NULL,
    @ImageTypeCode NVARCHAR(50)=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ByteThumb VARBINARY(MAX)=NULL,
    @BytePoster VARBINARY(MAX)=NULL,
    @ByteFull VARBINARY(MAX)=NULL,
    @ByteOriginal VARBINARY(MAX)=NULL,
    @Notes NVARCHAR(256)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Personals].[Photos] WHERE (([Code] = @Code) AND (    ↙
    [AccountCode]=@AccountCode) AND ([Deleted]=@Deleted)))
            BEGIN
                INSERT INTO [Personals].[Photos]
                (
                    [ModuleCode],
                    [ScreenCode],
                    [ImageTypeCode],
                    [AccountCode],
                    [ByteThumb],
                    [BytePoster],
                    [ByteFull],
                    [ByteOriginal],
                    [Notes],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    @ModuleCode,
                    @ScreenCode,
                    @ImageTypeCode,
                    @AccountCode,
                    @ByteThumb,
                    @BytePoster,
                    @ByteFull,
                    @ByteOriginal,
                    @Notes,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [Personals].[Photos]
                SET
                    [ModuleCode]=@ModuleCode,
                    [ScreenCode]=@ScreenCode,
                    [ImageTypeCode]=@ImageTypeCode,
                    [AccountCode]=@AccountCode,
                    [ByteThumb]=@ByteThumb,
                    [BytePoster]=@BytePoster,
                    [ByteFull]=@ByteFull,
                    [ByteOriginal]=@ByteOriginal,
                    [Notes]=@Notes,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([AccountCode]=@AccountCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPPhotosInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPPhotosDeletePermanently]    Script Date:  ↙
    08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Personal.Photos Table

CREATE PROC [Personals].[SPPhotosDeletePermanently]
(
    @Code INT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Photos] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Personals].[Photos]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
```

```
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPPhotosDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPPhotosDelete]    Script Date: 08/16/2011  ↙
    01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Personal.  ↙
    Photos Table

CREATE PROC [Personals].[SPPhotosDelete]
(
    @Code INT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Photos] WHERE (([Code] = @Code) AND (Deleted↙
    =@Deleted)))
            BEGIN
            UPDATE [Personals].[Photos]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN
```

```
DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPPhotosDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPPhonesSelect]    Script Date: 08/16/2011  ↙
    01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Personal.Phones Table

CREATE PROC [Personals].[SPPhonesSelect]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Phones] WHERE (([Code] = @Code) AND (      ↙
    [AccountCode]=@AccountCode) AND ([ScreenCode]=@ScreenCode) AND ([Deleted]=@Deleted)))
            BEGIN
            SELECT * FROM [Personals].[Phones]
            WHERE
                (
                ([Code] = @Code) AND
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Personals].[Phones]
            WHERE
                (
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
```

```sql
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPPhonesSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPPhonesInsertUpdate]    Script Date: 08/16/↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Personal.Phones Table

CREATE PROC [Personals].[SPPhonesInsertUpdate]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @PhoneTypeCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @PostalCode NVARCHAR(5)=NULL,
    @PhoneNumber NVARCHAR(50)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Personals].[Phones] WHERE (([Code] = @Code) AND (   ↙
    [AccountCode]=@AccountCode) AND ([Deleted]=@Deleted)))
            BEGIN
                INSERT INTO [Personals].[Phones]
                (
                    [AccountCode],
                    [PhoneTypeCode],
                    [ScreenCode],
                    [PostalCode],
                    [PhoneNumber],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    @AccountCode,
                    @PhoneTypeCode,
                    @ScreenCode,
                    @PostalCode,
                    @PhoneNumber,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [Personals].[Phones]
                SET
                    [AccountCode]=@AccountCode,
                    [PhoneTypeCode]=@PhoneTypeCode,
                    [ScreenCode]=@ScreenCode,
                    [PostalCode]=@PostalCode,
                    [PhoneNumber]=@PhoneNumber,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([AccountCode]=@AccountCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPPhonesInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPPhonesDeletePermanently]    Script Date: ↙
    08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
```

```
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Personal.Phones Table

CREATE PROC [Personals].[SPPhonesDeletePermanently]
(
    @Code INT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Phones] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Personals].[Phones]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPPhonesDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPPhonesDelete]    Script Date: 08/16/2011 ↙
    01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Personal. ↙
    Phones Table

CREATE PROC [Personals].[SPPhonesDelete]
(
    @Code INT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Phones] WHERE (([Code] = @Code) AND (Deleted↙
    =@Deleted)))
            BEGIN
            UPDATE [Personals].[Phones]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPPhonesDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPPermissionsSelect]    Script Date: 08/16/2011 ↙
    01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Academics.Semesters   ↙
    Table

CREATE PROC [SetUp].[SPPermissionsSelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Deleted BIT=NULL

)
```

```
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Permissions] WHERE (([Code] = @Code) AND (       ↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted)))
            BEGIN
            SELECT * FROM [SetUp].[Permissions]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [SetUp].[Permissions]
            WHERE
                (
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPPermissionsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPPermissionsInsertUpdate]    Script Date: 08/16↙
    /2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Godwin Mathias
--Reviewer          :   Godwin Mathias
--Date Created      :   08/06/2011
--Last Updated      :   08/06/2011
--Last Updated By   :   Godwin Mathias
--Description       :   Stored procedure for saving values into Academics.Semesters Table

CREATE PROC [SetUp].[SPPermissionsInsertUpdate]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @ModuleCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50)=NULL,
    @RoleCode UNIQUEIDENTIFIER=NULL,
    @ActionCode NVARCHAR(50)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[Permissions] WHERE ([Code] = @Code) AND (    ↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
                INSERT INTO [SetUp].[Permissions]
                (
                    [UniversityCode],
                    [ModuleCode],
                    [ScreenCode],
                    [RoleCode],
                    [ActionCode],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]

                )
                VALUES
                (
                    @UniversityCode,
                    @ModuleCode,
                    @ScreenCode,
                    @RoleCode,
                    @ActionCode,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted

                )
            END
        ELSE
            BEGIN
                UPDATE [SetUp].[Permissions]
                SET
                    [UniversityCode]=@UniversityCode,
                    [ModuleCode]=@ModuleCode,
                    [ScreenCode]=@ScreenCode,
                    [RoleCode]=@RoleCode,
                    [ActionCode]=@ActionCode,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted

                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN
```

```
DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPPermissionsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPPermissionsDeletePermanently]    Script Date: ↙
    08/16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Academics.Semesters    ↙
    Table

CREATE PROC [SetUp].[SPPermissionsDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Permissions] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [SetUp].[Permissions]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPPermissionsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPPermissionsDelete]    Script Date: 08/16/2011 ↙
    01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Academics. ↙
    Semesters Table

CREATE PROC [SetUp].[SPPermissionsDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Permissions] WHERE ([Code] = @Code) AND (Deleted↙
    =@Deleted))
            BEGIN
            UPDATE [SetUp].[Permissions]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPPermissionsDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPMedicalConditionsSelect]    Script Date:  ↙
    08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Personal.             ↙
    MedicalConditions Table

CREATE PROC [Personals].[SPMedicalConditionsSelect]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[MedicalConditions] WHERE (([Code] = @Code)  ↙
    AND ([AccountCode]=@AccountCode) AND ([ScreenCode]=@ScreenCode) AND ([Deleted]=@      ↙
    Deleted)))
            BEGIN
            SELECT * FROM [Personals].[MedicalConditions]
            WHERE
                (
                ([Code] = @Code) AND
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Personals].[MedicalConditions]
            WHERE
                (
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPMedicalConditionsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPMedicalConditionsInsertUpdate]    Script ↙
    Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Personals.                ↙
    MedicalConditions Table

CREATE PROC [Personals].[SPMedicalConditionsInsertUpdate]
(
    @Code INT=NULL,
    @DiseaseCode NVARCHAR(50)=NULL,
    @Conditions NVARCHAR(500)=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Personals].[MedicalConditions] WHERE (([Code] = @   ↙
    Code) AND ([AccountCode]=@AccountCode) AND ([Deleted]=@Deleted)))
            BEGIN
                INSERT INTO [Personals].[MedicalConditions]
                (
                    [DiseaseCode],
                    [Conditions],
                    [AccountCode],
                    [ScreenCode],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    @DiseaseCode,
                    @Conditions,
                    @AccountCode,
                    @ScreenCode,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [Personals].[MedicalConditions]
                SET
                    [DiseaseCode]=@DiseaseCode,
                    [Conditions]=@Conditions,
                    [AccountCode]=@AccountCode,
                    [ScreenCode]=@ScreenCode,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([AccountCode]=@AccountCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPMedicalConditionsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPMedicalConditionsDeletePermanently]    ↙
    Script Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Personal.        ↙
    MedicalConditions Table

CREATE PROC [Personals].[SPMedicalConditionsDeletePermanently]
(
    @Code INT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;
```

```
        IF EXISTS (SELECT * FROM [Personals].[MedicalConditions] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Personals].[MedicalConditions]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPMedicalConditionsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPMedicalConditionsDelete]    Script Date:  ↙
    08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Personal.  ↙
    MedicalConditions Table

CREATE PROC [Personals].[SPMedicalConditionsDelete]
(
    @Code INT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[MedicalConditions] WHERE (([Code] = @Code)  ↙
    AND (Deleted=@Deleted)))
            BEGIN
            UPDATE [Personals].[MedicalConditions]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPMedicalConditionsDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPLockingsSelect]    Script Date: 08/16/2011 01:↙
    24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for retrieving values from SetUp.Lockings Table

CREATE PROC [SetUp].[SPLockingsSelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Lockings] WHERE ([Code] = @Code) AND (          ↙
    [UniversityCode] = @UniversityCode))
            BEGIN
            SELECT * FROM [SetUp].[Lockings]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode)
                )
            END
```

```sql
        ELSE
            BEGIN
            SELECT * FROM [SetUp].[Lockings]
            WHERE
                (
                ([UniversityCode] = @UniversityCode)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPLockingsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPLockingsInsertUpdate]    Script Date: 08/16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for saving values into SetUp.Lockings Table

CREATE PROC [SetUp].[SPLockingsInsertUpdate]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @FacultyCode BIGINT=NULL,
    @DepartmentCode BIGINT=NULL,
    @CourseCode BIGINT=NULL,
    @ProgramCode BIGINT=NULL,
    @SessionCode NVARCHAR(50)=NULL,
    @SemesterCode NVARCHAR(50)=NULL,
    @LevelCode INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @EntityCode NVARCHAR(50)=NULL,
    @Locked BIT=NULL,
    @LockedOn DATETIME=NULL,
    @LockedBy NVARCHAR(50)=NULL,
    @UnlockedOn DATETIME=NULL,
    @UnlockedBy NVARCHAR(50)=NULL,
    @LastLockedOn DATETIME=NULL,
    @LastLockedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[Lockings]  WHERE ([Code] = @Code) AND (    ↙
    [UniversityCode] = @UniversityCode))
            BEGIN
                INSERT INTO [SetUp].[Lockings]
                (
                    [UniversityCode],
                    [FacultyCode],
                    [DepartmentCode],
                    [CourseCode],
                    [ProgramCode],
                    [SessionCode],
                    [SemesterCode],
                    [LevelCode],
                    [AccountCode],
                    [EntityCode],
                    [Locked],
                    [LockedOn],
                    [LockedBy],
                    [UnlockedOn],
                    [UnlockedBy],
                    [LastLockedOn],
                    [LastLockedBy]

                )
                VALUES
                (
                    @UniversityCode,
                    @FacultyCode,
                    @DepartmentCode,
                    @CourseCode,
                    @ProgramCode,
                    @SessionCode,
                    @SemesterCode,
                    @LevelCode,
                    @AccountCode,
                    @EntityCode,
                    @Locked,
                    @LockedOn,
                    @LockedBy,
                    @UnlockedOn,
                    @UnlockedBy,
                    @LastLockedOn,
                    @LastLockedBy
                )
            END
        ELSE
            BEGIN
                UPDATE [SetUp].[Lockings]
                SET
                    [UniversityCode]=@UniversityCode,
                    [FacultyCode]=@FacultyCode,
                    [DepartmentCode]=@DepartmentCode,
                    [CourseCode]=@CourseCode,
                    [ProgramCode]=@ProgramCode,
                    [SessionCode]=@SessionCode,
                    [SemesterCode]=@SemesterCode,
                    [LevelCode]=@LevelCode,
                    [AccountCode]=@AccountCode,
                    [EntityCode]=@EntityCode,
                    [Locked]=@Locked,
                    [LockedOn]=@LockedOn,
                    [LockedBy]=@LockedBy,
                    [UnlockedOn]=@UnlockedOn,
                    [UnlockedBy]=@UnlockedBy,
                    [LastLockedOn]=@LastLockedOn,
                    [LastLockedBy]=@LastLockedBy

                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPLockingsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPLockingsDeletePermanently]    Script Date: 08/↙
    16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/12/2011
--Last Updated      :   07/12/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from SetUp.Lockings Table

CREATE PROC [SetUp].[SPLockingsDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Lockings] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [SetUp].[Lockings]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY
```

```sql
BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPLockingsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPParametersSelect]    Script Date: 08/16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/14/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from SetUp.Parameters Table

CREATE PROC [SetUp].[SPParametersSelect]
(
    @Code NVARCHAR(50)=NULL

    )
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Parameters] WHERE ([Code] = @Code))
            BEGIN
            SELECT * FROM [SetUp].[Parameters]
            WHERE
                (
                ([Code] = @Code)
                    )
            END
        ELSE
            BEGIN
            SELECT * FROM [SetUp].[Parameters]

            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPParametersSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPParametersInsertUpdate]    Script Date: 08/16/↙
    2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into SetUp.Parameters Table

CREATE PROC [SetUp].[SPParametersInsertUpdate]
(
    @Code NVARCHAR(50)=NULL,
    @Description NVARCHAR(256)=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[Parameters]  WHERE ([Code] = @Code))
            BEGIN
                INSERT INTO [SetUp].[Parameters]
                (
                    [Code],
                    [Description]


                )
                VALUES
                (
                    @Code,
                    @Description

                )
            END
        ELSE
            BEGIN
                UPDATE [SetUp].[Parameters]
                SET
                    [Description]=@Description

                WHERE
                    ([Code] = @Code)
            END
COMMIT TRAN
```

```
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPParametersInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPParametersDeletePermanently]    Script Date:  ↙
    08/16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/12/2011
--Last Updated      :   07/12/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from SetUp.Parameters Table

CREATE PROC [SetUp].[SPParametersDeletePermanently]
(
    @Code NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Parameters] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [SetUp].[Parameters]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);
```

```
SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPParametersDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPOthersSelect]    Script Date: 08/16/2011   ↙
    01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for retrieving values from Personal.Others Table

CREATE PROC [Personals].[SPOthersSelect]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Others] WHERE (([Code] = @Code) AND (       ↙
    [AccountCode]=@AccountCode) AND ([ScreenCode]=@ScreenCode) AND ([Deleted]=@Deleted)))
            BEGIN
            SELECT * FROM [Personals].[Others]
            WHERE
                (
                ([Code] = @Code) AND
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Personals].[Others]
            WHERE
                (
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPOthersSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPOthersInsertUpdate]    Script Date: 08/16/↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Personal.Others Table

CREATE PROC [Personals].[SPOthersInsertUpdate]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @Notes NVARCHAR(500)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @OthersCode NVARCHAR(50)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Personals].[Others] WHERE (([Code] = @Code) AND (    ↙
    [AccountCode]=@AccountCode) AND ([Deleted]=@Deleted)))
            BEGIN
                INSERT INTO [Personals].[Others]
                (
                    [AccountCode],
                    [Notes],
                    [ScreenCode],
                    [OthersCode],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    @AccountCode,
                    @Notes,
                    @ScreenCode,
                    @OthersCode,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [Personals].[Others]
                SET
                    [AccountCode]=@AccountCode,
                    [Notes]=@Notes,
                    [ScreenCode]=@ScreenCode,
                    [OthersCode]=@OthersCode,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([AccountCode]=@AccountCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPOthersInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPOthersDeletePermanently]    Script Date:  ↙
    08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for deleting values from Personal.Others Table

CREATE PROC [Personals].[SPOthersDeletePermanently]
(
    @Code INT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
```

```
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Others] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Personals].[Others]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPOthersDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPOthersDelete]    Script Date: 08/16/2011  ↙
    01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for deleting values temporarily from Personal.  ↙
    Others Table

CREATE PROC [Personals].[SPOthersDelete]
(
    @Code INT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Others] WHERE (([Code] = @Code) AND (Deleted↙
    =@Deleted)))
            BEGIN
            UPDATE [Personals].[Others]
            SET
```

```
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPOthersDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPLanguageSkillsSelect]    Script Date: 08/ ↙
    16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Personal.          ↙
    LanguageSkills Table

CREATE PROC [Personals].[SPLanguageSkillsSelect]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[LanguageSkills] WHERE (([Code] = @Code) AND ↙
    ([AccountCode]=@AccountCode) AND ([ScreenCode]=@ScreenCode) AND ([Deleted]=@Deleted)))
            BEGIN
            SELECT * FROM [Personals].[LanguageSkills]
            WHERE
                (
                ([Code] = @Code) AND
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Personals].[LanguageSkills]
            WHERE
                (
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPLanguageSkillsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPLanguageSkillsInsertUpdate]    Script Date↙
    : 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author             :   Ademola Adebo
--Reviewer           :   Godwin Mathias
--Date Created       :   07/08/2011
--Last Updated       :   07/08/2011
--Last Updated By    :   Ademola Adebo
--Description        :   Stored procedure for saving values into Personals.LanguageSkills  ↙
    Table

CREATE PROC [Personals].[SPLanguageSkillsInsertUpdate]
(
    @Code INT=NULL,
    @LanguageCode NVARCHAR(50)=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @DateX DATETIME=NULL,
    @ScreenCode NVARCHAR(50)=NULL,
    @ReadingDegree NVARCHAR(50)=NULL,
    @SpeakingDegree NVARCHAR(50)=NULL,
    @WritingDegree NVARCHAR(50)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Personals].[LanguageSkills] WHERE (([Code] = @Code) ↙
    AND ([AccountCode]=@AccountCode) AND ([Deleted]=@Deleted)))
            BEGIN
                INSERT INTO [Personals].[LanguageSkills]
                (
                    [LanguageCode],
                    [AccountCode],
                    [DateX],
                    [ScreenCode],
                    [ReadingDegree],
                    [SpeakingDegree],
                    [WritingDegree],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    @LanguageCode,
                    @AccountCode,
                    @DateX,
                    @ScreenCode,
                    @ReadingDegree,
                    @SpeakingDegree,
                    @WritingDegree,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [Personals].[LanguageSkills]
                SET
                    [LanguageCode]=@LanguageCode,
                    [AccountCode]=@AccountCode,
                    [DateX]=@DateX,
                    [ScreenCode]=@ScreenCode,
                    [ReadingDegree]=@ReadingDegree,
                    [SpeakingDegree]=@SpeakingDegree,
                    [WritingDegree]=@WritingDegree,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([AccountCode]=@AccountCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPLanguageSkillsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPLanguageSkillsDeletePermanently]    Script↙
     Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Personal.LanguageSkills ↙
    Table

CREATE PROC [Personals].[SPLanguageSkillsDeletePermanently]
(
    @Code INT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[LanguageSkills] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Personals].[LanguageSkills]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()
```

```
RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPLanguageSkillsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPLanguageSkillsDelete]    Script Date: 08/ ↙
    16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Personal.   ↙
    LanguageSkills Table

CREATE PROC [Personals].[SPLanguageSkillsDelete]
(
    @Code INT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[LanguageSkills] WHERE (([Code] = @Code) AND ↙
    (Deleted=@Deleted)))
            BEGIN
            UPDATE [Personals].[LanguageSkills]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);
```

```
END CATCH

GRANT EXECUTE ON [Personals].[SPLanguageSkillsDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPJAMBsSelect]    Script Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Personal.JAMBs Table

CREATE PROC [Personals].[SPJAMBsSelect]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[JAMBs] WHERE (([Code] = @Code) AND ([AccountCode]=@AccountCode) AND ([ScreenCode]=@ScreenCode) AND ([Deleted]=@Deleted)))
            BEGIN
            SELECT * FROM [Personals].[JAMBs]
            WHERE
                (
                ([Code] = @Code) AND
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Personals].[JAMBs]
            WHERE
                (
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPJAMBsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPJAMBsInsertUpdate]    Script Date: 08/16/ ↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Personals.JAMBs Table

CREATE PROC [Personals].[SPJAMBsInsertUpdate]
(
    @Code INT=NULL,
    @RegNo NVARCHAR(50)=NULL,
    @Score INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50)=NULL,
    @Year INT=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Personals].[JAMBs] WHERE (([Code] = @Code) AND (     ↙
    [AccountCode]=@AccountCode) AND ([Deleted]=@Deleted)))
            BEGIN
                INSERT INTO [Personals].[JAMBs]
                (
                    [RegNo],
                    [Score],
                    [AccountCode],
                    [ScreenCode],
                    [Year],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    @RegNo,
                    @Score,
                    @AccountCode,
                    @ScreenCode,
                    @Year,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [Personals].[JAMBs]
                SET
                    [RegNo]=@RegNo,
                    [Score]=@Score,
                    [AccountCode]=@AccountCode,
                    [ScreenCode]=@ScreenCode,
                    [Year]=@Year,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([AccountCode]=@AccountCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPJAMBsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPJAMBsDeletePermanently]    Script Date: 08↙
    /16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Personal.JAMBs Table

CREATE PROC [Personals].[SPJAMBsDeletePermanently]
(
    @Code INT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[JAMBs] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Personals].[JAMBs]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPJAMBsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPJAMBsDelete]    Script Date: 08/16/2011 01↙
    :24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for deleting values temporarily from Personal.   ↙
    JAMBs Table

CREATE PROC [Personals].[SPJAMBsDelete]
(
    @Code INT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[JAMBs] WHERE (([Code] = @Code) AND (Deleted=↙
    @Deleted)))
            BEGIN
            UPDATE [Personals].[JAMBs]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPJAMBsDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPHobbiesSelect]    Script Date: 08/16/2011 ↙
    01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author             :   Ademola Adebo
--Reviewer           :   Godwin Mathias
--Date Created       :   07/08/2011
--Last Updated       :   07/08/2011
--Last Updated By    :   Ademola Adebo
--Description        :   Stored procedure for retrieving values from Personal.Hobbies Table

CREATE PROC [Personals].[SPHobbiesSelect]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Hobbies] WHERE (([Code] = @Code) AND (      ↙
    [AccountCode]=@AccountCode) AND ([ScreenCode]=@ScreenCode) AND ([Deleted]=@Deleted)))
            BEGIN
            SELECT * FROM [Personals].[Hobbies]
            WHERE
                (
                ([Code] = @Code) AND
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Personals].[Hobbies]
            WHERE
                (
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPHobbiesSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPHobbiesInsertUpdate]    Script Date: 08/16↙
    /2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Personals.Hobbies Table

CREATE PROC [Personals].[SPHobbiesInsertUpdate]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @HobbyCode NVARCHAR(50)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;
```

```
        IF NOT EXISTS (SELECT * FROM [Personals].[Hobbies] WHERE (([Code] = @Code) AND (  ↙
    [AccountCode]=@AccountCode) AND ([Deleted]=@Deleted)))
            BEGIN
                INSERT INTO [Personals].[Hobbies]
                (
                    [AccountCode],
                    [ScreenCode],
                    [HobbyCode],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    @AccountCode,
                    @ScreenCode,
                    @HobbyCode,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [Personals].[Hobbies]
                SET
                    [AccountCode]=@AccountCode,
                    [ScreenCode]=@ScreenCode,
                    [HobbyCode]=@HobbyCode,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([AccountCode]=@AccountCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPHobbiesInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPHobbiesDeletePermanently]    Script Date: ↙
    08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
```

```
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Personal.Hobbies Table

CREATE PROC [Personals].[SPHobbiesDeletePermanently]
(
    @Code INT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Hobbies] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Personals].[Hobbies]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPHobbiesDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPHobbiesDelete]    Script Date: 08/16/2011 ↙
    01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Personal.   ↙
    Hobbies Table

CREATE PROC [Personals].[SPHobbiesDelete]
(
    @Code INT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Hobbies] WHERE (([Code] = @Code) AND        ↙
    (Deleted=@Deleted)))
            BEGIN
            UPDATE [Personals].[Hobbies]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPHobbiesDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPGuardiansSelect]    Script Date: 08/16/   ↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Personal.Guardians   ↙
    Table

CREATE PROC [Personals].[SPGuardiansSelect]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Guardians] WHERE (([Code] = @Code) AND (    ↙
    [AccountCode]=@AccountCode) AND ([ScreenCode]=@ScreenCode) AND ([Deleted]=@Deleted)))
            BEGIN
            SELECT * FROM [Personals].[Guardians]
            WHERE
                (
                ([Code] = @Code) AND
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Personals].[Guardians]
            WHERE
                (
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPGuardiansSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPGuardiansInsertUpdate]    Script Date: 08/↙
    16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for saving values into Personals.Guardians Table
```

```
CREATE PROC [Personals].[SPGuardiansInsertUpdate]
(
    @Code INT=NULL,
    @GuardianCode NVARCHAR(50)=NULL,
    @TitleCode NVARCHAR(50)=NULL,
    @GuardianName NVARCHAR(50)=NULL,
    @OccupationCode NVARCHAR(50)=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Personals].[Guardians] WHERE (([Code] = @Code) AND (↙
    [AccountCode]=@AccountCode) AND ([Deleted]=@Deleted)))
            BEGIN
                INSERT INTO [Personals].[Guardians]
                (
                    [GuardianCode],
                    [TitleCode],
                    [GuardianName],
                    [OccupationCode],
                    [AccountCode],
                    [ScreenCode],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    @GuardianCode,
                    @TitleCode,
                    @GuardianName,
                    @OccupationCode,
                    @AccountCode,
                    @ScreenCode,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [Personals].[Guardians]
                SET
                    [GuardianCode]=@GuardianCode,
                    [TitleCode]=@TitleCode,
                    [GuardianName]=@GuardianName,
                    [OccupationCode]=@OccupationCode,
                    [AccountCode]=@AccountCode,
                    [ScreenCode]=@ScreenCode,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([AccountCode]=@AccountCode))
            END
```

```
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPGuardiansInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPGuardiansDeletePermanently]    Script Date↙
    : 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Personal.Guardians Table

CREATE PROC [Personals].[SPGuardiansDeletePermanently]
(
    @Code INT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Guardians] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Personals].[Guardians]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);
```

```
SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPGuardiansDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPGuardiansDelete]    Script Date: 08/16/   ↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Personal.   ↙
    Guardians Table

CREATE PROC [Personals].[SPGuardiansDelete]
(
    @Code INT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Guardians] WHERE (([Code] = @Code) AND      ↙
    (Deleted=@Deleted)))
            BEGIN
            UPDATE [Personals].[Guardians]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);
```

```
SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPGuardiansDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPGuarantorsSelect]    Script Date: 08/16/  ↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for retrieving values from Personal.Guarantors  ↙
    Table

CREATE PROC [Personals].[SPGuarantorsSelect]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Guarantors] WHERE (([Code] = @Code) AND (  ↙
    [AccountCode]=@AccountCode) AND ([ScreenCode]=@ScreenCode) AND ([Deleted]=@Deleted)))
            BEGIN
            SELECT * FROM [Personals].[Guarantors]
            WHERE
                (
                ([Code] = @Code) AND
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Personals].[Guarantors]
            WHERE
                (
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN
```

```
DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPGuarantorsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPGuarantorsInsertUpdate]    Script Date: 08↙
    /16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Personals.Guarantors Table

CREATE PROC [Personals].[SPGuarantorsInsertUpdate]
(
    @Code INT=NULL,
    @TitleCode NVARCHAR(150)=NULL,
    @Name NVARCHAR(150)=NULL,
    @OccupationCode NVARCHAR(50)=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50)=NULL,
    @Undertaking NVARCHAR(500)=NULL,
    @Notes NVARCHAR(500)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Personals].[Guarantors] WHERE (([Code] = @Code) AND ↙
    ([AccountCode]=@AccountCode) AND ([Deleted]=@Deleted)))
            BEGIN
                INSERT INTO [Personals].[Guarantors]
                (
                    [TitleCode],
                    [Name],
                    [OccupationCode],
                    [AccountCode],
                    [ScreenCode],
                    [Undertaking],
                    [Notes],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    @TitleCode,
                    @Name,
                    @OccupationCode,
                    @AccountCode,
                    @ScreenCode,
                    @Undertaking,
                    @Notes,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [Personals].[Guarantors]
                SET
                    [TitleCode]=@TitleCode,
                    [Name]=@Name,
                    [OccupationCode]=@OccupationCode,
                    [AccountCode]=@AccountCode,
                    [ScreenCode]=@ScreenCode,
                    [Undertaking]=@Undertaking,
                    [Notes]=@Notes,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([AccountCode]=@AccountCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPGuarantorsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPGuarantorsDeletePermanently]    Script    ↙
    Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
```

```
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Personal.Guarantors    ↙
    Table

CREATE PROC [Personals].[SPGuarantorsDeletePermanently]
(
    @Code INT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Guarantors] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Personals].[Guarantors]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPGuarantorsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPGuarantorsDelete]    Script Date: 08/16/  ↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Personal.  ↙
    Guarantors Table

CREATE PROC [Personals].[SPGuarantorsDelete]
(
    @Code INT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Guarantors] WHERE (([Code] = @Code) AND    ↙
    (Deleted=@Deleted)))
            BEGIN
            UPDATE [Personals].[Guarantors]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPGuarantorsDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPOLevelsSelect]    Script Date: 08/16/2011 ↙
    01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Personal.OLevels Table

CREATE PROC [Personals].[SPOLevelsSelect]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[OLevels] WHERE (([Code] = @Code) AND (      ↙
    [AccountCode]=@AccountCode) AND ([ScreenCode]=@ScreenCode) AND ([Deleted]=@Deleted)))
            BEGIN
            SELECT * FROM [Personals].[OLevels]
            WHERE
                (
                ([Code] = @Code) AND
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Personals].[OLevels]
            WHERE
                (
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPOLevelsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPOLevelsInsertUpdate]    Script Date: 08/16↙
    /2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Personals.OLevels Table
```

```
CREATE PROC [Personals].[SPOLevelsInsertUpdate]
(
    @Code INT=NULL,
    @SubjectCode NVARCHAR(50)=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50)=NULL,
    @GradeCode INT=NULL,
    @GradeDescCode NVARCHAR(50)=NULL,
    @ExamTypeCode NVARCHAR(50)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Personals].[OLevels] WHERE (([Code] = @Code) AND (   ↙
    [AccountCode]=@AccountCode) AND ([Deleted]=@Deleted)))
            BEGIN
                INSERT INTO [Personals].[OLevels]
                (
                    [SubjectCode],
                    [AccountCode],
                    [ScreenCode],
                    [GradeCode],
                    [GradeDescCode],
                    [ExamTypeCode],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    @SubjectCode,
                    @AccountCode,
                    @ScreenCode,
                    @GradeCode,
                    @GradeDescCode,
                    @ExamTypeCode,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [Personals].[OLevels]
                SET
                    [SubjectCode]=@SubjectCode,
                    [AccountCode]=@AccountCode,
                    [ScreenCode]=@ScreenCode,
                    [GradeCode]=@GradeCode,
                    [GradeDescCode]=@GradeDescCode,
                    [ExamTypeCode]=@ExamTypeCode,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([AccountCode]=@AccountCode))
            END
COMMIT TRAN
```

```
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPOLevelsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPOLevelsDeletePermanently]    Script Date: ↙
    08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Personal.OLevels Table

CREATE PROC [Personals].[SPOLevelsDeletePermanently]
(
    @Code INT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[OLevels] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Personals].[OLevels]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);
```

```
SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPOLevelsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPOLevelsDelete]    Script Date: 08/16/2011 ↙
    01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Personal.  ↙
    OLevels Table

CREATE PROC [Personals].[SPOLevelsDelete]
(
    @Code INT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[OLevels] WHERE (([Code] = @Code) AND       ↙
    (Deleted=@Deleted)))
            BEGIN
            UPDATE [Personals].[OLevels]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
```

```
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPOLevelsDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPNextOfKinsSelect]    Script Date: 08/16/ ↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Personal.NextOfKins ↙
    Table

CREATE PROC [Personals].[SPNextOfKinsSelect]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[NextOfKins] WHERE (([Code] = @Code) AND ( ↙
    [AccountCode]=@AccountCode) AND ([ScreenCode]=@ScreenCode) AND ([Deleted]=@Deleted)))
            BEGIN
            SELECT * FROM [Personals].[NextOfKins]
            WHERE
                (
                ([Code] = @Code) AND
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Personals].[NextOfKins]
            WHERE
                (
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPNextOfKinsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPNextOfKinsInsertUpdate]    Script Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Personals.NextOfKins Table

CREATE PROC [Personals].[SPNextOfKinsInsertUpdate]
(
    @Code INT=NULL,
    @RelTypeCode NVARCHAR(50)=NULL,
    @TitleCode NVARCHAR(50)=NULL,
    @Name NVARCHAR(50)=NULL,
    @OccupationCode NVARCHAR(50)=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Personals].[NextOfKins] WHERE (([Code] = @Code) AND ([AccountCode]=@AccountCode) AND ([Deleted]=@Deleted)))
            BEGIN
                INSERT INTO [Personals].[NextOfKins]
                (
                    [RelTypeCode],
                    [TitleCode],
                    [Name],
                    [OccupationCode],
                    [AccountCode],
                    [ScreenCode],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    @RelTypeCode,
                    @TitleCode,
                    @Name,
                    @OccupationCode,
                    @AccountCode,
                    @ScreenCode,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [Personals].[NextOfKins]
                SET
                    [RelTypeCode]=@RelTypeCode,
                    [TitleCode]=@TitleCode,
                    [Name]=@Name,
                    [OccupationCode]=@OccupationCode,
                    [AccountCode]=@AccountCode,
                    [ScreenCode]=@ScreenCode,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([AccountCode]=@AccountCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPNextOfKinsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPNextOfKinsDeletePermanently]    Script    ↙
    Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for deleting values from Personal.NextOfKins    ↙
```

```
    Table

CREATE PROC [Personals].[SPNextOfKinsDeletePermanently]
(
    @Code INT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[NextOfKins] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Personals].[NextOfKins]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPNextOfKinsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPNextOfKinsDelete]    Script Date: 08/16/ ↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Personal. ↙
    NextOfKins Table

CREATE PROC [Personals].[SPNextOfKinsDelete]
(
    @Code INT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
```

```
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[NextOfKins] WHERE (([Code] = @Code) AND     ↙
    (Deleted=@Deleted)))
            BEGIN
            UPDATE [Personals].[NextOfKins]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPNextOfKinsDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPEntryRequirementsSelect]    Script Date: 08/16↙
    /2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from SetUp.               ↙
    EntryRequirements Table

CREATE PROC [SetUp].[SPEntryRequirementsSelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
```

```
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[EntryRequirements] WHERE ([Code] = @Code) AND ( ↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
            SELECT * FROM [SetUp].[EntryRequirements]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [SetUp].[EntryRequirements]
            WHERE
                (
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPEntryRequirementsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPEntryRequirementsInsertUpdate]    Script Date:↙
     08/16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into SetUp.EntryRequirements   ↙
    Table

CREATE PROC [SetUp].[SPEntryRequirementsInsertUpdate]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @FacultyCode BIGINT=NULL,
    @DepartmentCode BIGINT=NULL,
    @CourseCode BIGINT=NULL,
    @ProgramCode BIGINT=NULL,
    @EntryMode NVARCHAR(50)=NULL,
    @ModeOfStudy NVARCHAR(50)=NULL,
    @NoOfCredits INT=NULL,
    @NoOfSittings INT=NULL,
    @RequirementType NVARCHAR(5)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[EntryRequirements]  WHERE ([Code] = @Code)  ↙
    AND ([UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
                INSERT INTO [SetUp].[EntryRequirements]
                (
                    [UniversityCode],
                    [FacultyCode],
                    [DepartmentCode],
                    [CourseCode],
                    [ProgramCode],
                    [EntryMode],
                    [ModeOfStudy],
                    [NoOfCredits],
                    [NoOfSittings],
                    [RequirementType],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]

                )
                VALUES
                (
                    @UniversityCode,
                    @FacultyCode,
                    @DepartmentCode,
                    @CourseCode,
                    @ProgramCode,
                    @EntryMode,
                    @ModeOfStudy,
                    @NoOfCredits,
                    @NoOfSittings,
                    @RequirementType,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [SetUp].[EntryRequirements]
                SET
                    [UniversityCode]=@UniversityCode,
                    [FacultyCode]=@FacultyCode,
                    [DepartmentCode]=@DepartmentCode,
                    [CourseCode]=@CourseCode,
                    [ProgramCode]=@ProgramCode,
                    [EntryMode]=@EntryMode,
                    [ModeOfStudy]=@ModeOfStudy,
                    [NoOfCredits]=@NoOfCredits,
                    [NoOfSittings]=@NoOfSittings,
                    [RequirementType]=@RequirementType,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPEntryRequirementsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPEntryRequirementsDeletePermanently]    Script ↙
    Date: 08/16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/12/2011
--Last Updated      :   07/12/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from SetUp.EntryRequirements ↙
    Table

CREATE PROC [SetUp].[SPEntryRequirementsDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[EntryRequirements] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [SetUp].[EntryRequirements]
            WHERE
                (
                    ([Code] = @Code)
                )
            END
```

```
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPEntryRequirementsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPEntryRequirementsDelete]    Script Date: 08/16↙
    /2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/27/2011
--Last Updated      :   07/27/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from SetUp.    ↙
    EntryRequirements Table

CREATE PROC [SetUp].[SPEntryRequirementsDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[EntryRequirements] WHERE (([Code] = @Code)  ↙
    AND (Deleted=@Deleted)))
            BEGIN
            UPDATE [SetUp].[EntryRequirements]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END
```

```
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPEntryRequirementsDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPEmploymentHistoriesSelect]    Script Date:↙
     08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Personal.             ↙
    EmploymentHistories Table

CREATE PROC [Personals].[SPEmploymentHistoriesSelect]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[EmploymentHistories] WHERE (([Code] = @Code)↙
     AND ([AccountCode]=@AccountCode) AND ([ScreenCode]=@ScreenCode) AND ([Deleted]=@     ↙
    Deleted)))
            BEGIN
            SELECT * FROM [Personals].[EmploymentHistories]
            WHERE
                (
                ([Code] = @Code) AND
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Personals].[EmploymentHistories]
            WHERE
                (
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPEmploymentHistoriesSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPEmploymentHistoriesInsertUpdate]    Script↙
     Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Personals.              ↙
    EmploymentHistories Table

CREATE PROC [Personals].[SPEmploymentHistoriesInsertUpdate]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50)=NULL,
    @TitleCode NVARCHAR(50)=NULL,
    @JobCategoryCode NVARCHAR(50)=NULL,
    @JobDescCode NVARCHAR(50)=NULL,
    @LevelOfXpertiseCode NVARCHAR(50)=NULL,
    @WorkPercentageCode NVARCHAR(50)=NULL,
    @FromYearCode NVARCHAR(50)=NULL,
    @ToYearCode NVARCHAR(50)=NULL,
    @FromMonthCode NVARCHAR(50)=NULL,
    @ToMonthCode NVARCHAR(50)=NULL,
    @FromSAS NVARCHAR(50)=NULL,
    @FromSASCurrencyCode NVARCHAR(50)=NULL,
    @FromSASTotals DECIMAL(18,0)=NULL,
    @FromEAS NVARCHAR(50)=NULL,
    @FromEASCurrencyCode NVARCHAR(50)=NULL,
    @FromEASTotals DECIMAL(18,0)=NULL,
    @Supervisor NVARCHAR(150)=NULL,
    @SupervisorTitle NVARCHAR(50)=NULL,
    @Employer NVARCHAR(256)=NULL,
    @EmployerAddress NVARCHAR(256)=NULL,
    @EmployerBusinessNatureCode NVARCHAR(50)=NULL,
    @EmployerURL NVARCHAR(256)=NULL,
    @DutiesDescription NVARCHAR(500)=NULL,
    @KeyAchievements NVARCHAR(500)=NULL,
    @ContactRefrenceCode NVARCHAR(50)=NULL,
    @NoOfPeopleSupervised INT=NULL,
    @ReasonForLeaving NVARCHAR(256)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Personals].[EmploymentHistories] WHERE (([Code] = @ ↙
    Code) AND ([AccountCode]=@AccountCode) AND ([Deleted]=@Deleted)))
            BEGIN
                INSERT INTO [Personals].[EmploymentHistories]
                (
                    [AccountCode],
                    [ScreenCode],
                    [TitleCode],
                    [JobCategoryCode],
                    [JobDescCode],
                    [LevelOfXpertiseCode],
                    [WorkPercentageCode],
                    [FromYearCode],
                    [ToYearCode],
                    [FromMonthCode],
                    [ToMonthCode],
                    [FromSAS],
                    [FromSASCurrencyCode],
                    [FromSASTotals],
                    [FromEAS],
                    [FromEASCurrencyCode],
                    [FromEASTotals],
                    [Supervisor],
                    [SupervisorTitle],
                    [Employer],
                    [EmployerAddress],
                    [EmployerBusinessNatureCode],
                    [EmployerURL],
                    [DutiesDescription],
                    [KeyAchievements],
                    [ContactRefrenceCode],
                    [NoOfPeopleSupervised],
                    [ReasonForLeaving],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    @AccountCode,
                    @ScreenCode,
                    @TitleCode,
                    @JobCategoryCode,
                    @JobDescCode,
                    @LevelOfXpertiseCode,
                    @WorkPercentageCode,
                    @FromYearCode,
                    @ToYearCode,
                    @FromMonthCode,
                    @ToMonthCode,
                    @FromSAS,
                    @FromSASCurrencyCode,
                    @FromSASTotals,
                    @FromEAS,
                    @FromEASCurrencyCode,
                    @FromEASTotals,
                    @Supervisor,
                    @SupervisorTitle,
                    @Employer,
                    @EmployerAddress,
                    @EmployerBusinessNatureCode,
                    @EmployerURL,
                    @DutiesDescription,
                    @KeyAchievements,
                    @ContactRefrenceCode,
                    @NoOfPeopleSupervised,
                    @ReasonForLeaving,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [Personals].[EmploymentHistories]
                SET
                    [AccountCode]=@AccountCode,
                    [ScreenCode]=@ScreenCode,
                    [TitleCode]=@TitleCode,
                    [JobCategoryCode]=@JobCategoryCode,
                    [JobDescCode]=@JobDescCode,
                    [LevelOfXpertiseCode]=@LevelOfXpertiseCode,
                    [WorkPercentageCode]=@WorkPercentageCode,
                    [FromYearCode]=@FromYearCode,
                    [ToYearCode]=@ToYearCode,
                    [FromMonthCode]=@FromMonthCode,
                    [ToMonthCode]=@ToMonthCode,
                    [FromSAS]=@FromSAS,
                    [FromSASCurrencyCode]=@FromSASCurrencyCode,
                    [FromSASTotals]=@FromSASTotals,
                    [FromEAS]=@FromEAS,
                    [FromEASCurrencyCode]=@FromEASCurrencyCode,
                    [FromEASTotals]=@FromEASTotals,
                    [Supervisor]=@Supervisor,
                    [SupervisorTitle]=@SupervisorTitle,
                    [Employer]=@Employer,
                    [EmployerAddress]=@EmployerAddress,
                    [EmployerBusinessNatureCode]=@EmployerBusinessNatureCode,
                    [EmployerURL]=@EmployerURL,
                    [DutiesDescription]=@DutiesDescription,
                    [KeyAchievements]=@KeyAchievements,
                    [ContactRefrenceCode]=@ContactRefrenceCode,
                    [NoOfPeopleSupervised]=@NoOfPeopleSupervised,
                    [ReasonForLeaving]=@ReasonForLeaving,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([AccountCode]=@AccountCode))
            END
COMMIT TRAN
```

```
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPEmploymentHistoriesInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPEmploymentHistoriesDeletePermanently]     ↙
    Script Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Personal.            ↙
    EmploymentHistories Table

CREATE PROC [Personals].[SPEmploymentHistoriesDeletePermanently]
(
    @Code INT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[EmploymentHistories] WHERE (([Code] = @     ↙
    Code)))
            BEGIN
            DELETE FROM [Personals].[EmploymentHistories]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
```

```sql
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPEmploymentHistoriesDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPEmploymentHistoriesDelete]    Script Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author             :   Ademola Adebo
--Reviewer           :   Godwin Mathias
--Date Created       :   07/08/2011
--Last Updated       :   07/08/2011
--Last Updated By    :   Ademola Adebo
--Description        :   Stored procedure for deleting values temporarily from Personal.EmploymentHistories Table

CREATE PROC [Personals].[SPEmploymentHistoriesDelete]
(
    @Code INT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[EmploymentHistories] WHERE (([Code] = @Code) AND (Deleted=@Deleted)))
            BEGIN
            UPDATE [Personals].[EmploymentHistories]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);
```

```
SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPEmploymentHistoriesDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPEmergenciesSelect]    Script Date: 08/16/ ↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Personal.Emergencies ↙
    Table

CREATE PROC [Personals].[SPEmergenciesSelect]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Emergencies] WHERE (([Code] = @Code) AND ( ↙
    [AccountCode]=@AccountCode) AND ([ScreenCode]=@ScreenCode) AND ([Deleted]=@Deleted)))
            BEGIN
            SELECT * FROM [Personals].[Emergencies]
            WHERE
                (
                ([Code] = @Code) AND
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Personals].[Emergencies]
            WHERE
                (
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN
```

```
DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPEmergenciesSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPEmergenciesInsertUpdate]    Script Date:  ↙
    08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Personals.Emergencies     ↙
    Table

CREATE PROC [Personals].[SPEmergenciesInsertUpdate]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50)=NULL,
    @TitleCode NVARCHAR(50)=NULL,
    @Name NVARCHAR(150)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Personals].[Emergencies] WHERE (([Code] = @Code) AND↙
     ([AccountCode]=@AccountCode) AND ([Deleted]=@Deleted)))
            BEGIN
                INSERT INTO [Personals].[Emergencies]
                (
                    [AccountCode],
                    [ScreenCode],
                    [TitleCode],
                    [Name],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
```

```sql
                VALUES
                (
                    @AccountCode,
                    @ScreenCode,
                    @TitleCode,
                    @Name,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [Personals].[Emergencies]
                SET
                    [AccountCode]=@AccountCode,
                    [ScreenCode]=@ScreenCode,
                    [TitleCode]=@TitleCode,
                    [Name]=@Name,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([AccountCode]=@AccountCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPEmergenciesInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPEmergenciesDeletePermanently]    Script  ↙
    Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Personal.Emergencies  ↙
    Table

CREATE PROC [Personals].[SPEmergenciesDeletePermanently]
(
    @Code INT=NULL
```

```
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Emergencies] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Personals].[Emergencies]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPEmergenciesDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPEmergenciesDelete]    Script Date: 08/16/ ↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Personal. ↙
    Emergencies Table

CREATE PROC [Personals].[SPEmergenciesDelete]
(
    @Code INT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Emergencies] WHERE (([Code] = @Code) AND ↙
    (Deleted=@Deleted)))
            BEGIN
            UPDATE [Personals].[Emergencies]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPEmergenciesDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPEmailsSelect]    Script Date: 08/16/2011  ↙
    01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Personal.Emails Table

CREATE PROC [Personals].[SPEmailsSelect]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Emails] WHERE (([Code] = @Code) AND (      ↙
    [AccountCode]=@AccountCode) AND ([ScreenCode]=@ScreenCode) AND ([Deleted]=@Deleted)))
            BEGIN
            SELECT * FROM [Personals].[Emails]
            WHERE
                (
                ([Code] = @Code) AND
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Personals].[Emails]
            WHERE
                (
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPEmailsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPEmailsInsertUpdate]    Script Date: 08/16/↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Personals.Emails Table

CREATE PROC [Personals].[SPEmailsInsertUpdate]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @Email NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Personals].[Emails] WHERE (([Code] = @Code) AND (  ↙
    [AccountCode]=@AccountCode) AND ([Deleted]=@Deleted)))
            BEGIN
                INSERT INTO [Personals].[Emails]
                (
                    [AccountCode],
                    [Email],
                    [ScreenCode],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    @AccountCode,
                    @Email,
                    @ScreenCode,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [Personals].[Emails]
                SET
                    [AccountCode]=@AccountCode,
                    [Email]=@Email,
                    [ScreenCode]=@ScreenCode,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([AccountCode]=@AccountCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH
```

```
GRANT EXECUTE ON [Personals].[SPEmailsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPEmailsDeletePermanently]    Script Date: ↙
    08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Personal.Emails Table

CREATE PROC [Personals].[SPEmailsDeletePermanently]
(
    @Code INT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Emails] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Personals].[Emails]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPEmailsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPEmailsDelete]    Script Date: 08/16/2011 ↙
    01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
```

```
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Personal.   ↙
    Emails Table

CREATE PROC [Personals].[SPEmailsDelete]
(
    @Code INT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Emails] WHERE (([Code] = @Code) AND (Deleted↙
    =@Deleted)))
            BEGIN
            UPDATE [Personals].[Emails]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPEmailsDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPGetUrl]    Script Date: 08/16/2011 01:24:29 **↙
    ****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE PROC [SetUp].[SPGetUrl]
(
    @SearchWord nvarchar(50)
)
```

```
AS
BEGIN
    DECLARE @SearchedWord NVARCHAR(256)=NULL
    SET @SearchedWord = (SELECT Url FROM [SetUp].[Universities] WHERE FREETEXT(Url, @     ↙
    SearchWord))
    select @SearchedWord
END
GO
/****** Object:  StoredProcedure [SetUp].[SPCourseRegulationSelect]    Script Date: 08/16/↙
    2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for retrieving values from SetUp.CourseRegulation↙
     Table

CREATE PROC [SetUp].[SPCourseRegulationSelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[CourseRegulation] WHERE ([Code] = @Code) AND (   ↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
            SELECT * FROM [SetUp].[CourseRegulation]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [SetUp].[CourseRegulation]
            WHERE
                (
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPCourseRegulationSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPCourseRegulationInsertUpdate]    Script Date: ↙
    08/16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into SetUp.CourseRegulation    ↙
    Table

CREATE PROC [SetUp].[SPCourseRegulationInsertUpdate]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @FacultyCode BIGINT=NULL,
    @DepartmentCode BIGINT=NULL,
    @CourseCode BIGINT=NULL,
    @ProgramCode BIGINT=NULL,
    @LevelCode BIGINT=NULL,
    @SemesterCode NVARCHAR(50)=NULL,
    @CreditLowerBound INT=NULL,
    @CreditUpperBound INT=NULL,
    @ModeOfStudyCode NVARCHAR(50)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[CourseRegulation]  WHERE ([Code] = @Code)   ↙
    AND ([UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
                INSERT INTO [SetUp].[CourseRegulation]
                (
                    [UniversityCode],
                    [FacultyCode],
                    [DepartmentCode],
                    [CourseCode],
                    [ProgramCode],
                    [LevelCode],
                    [SemesterCode],
                    [CreditLowerBound],
                    [CreditUpperBound],
                    [ModeOfStudyCode],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]

                )
                VALUES
                (
                    @UniversityCode,
                    @FacultyCode,
                    @DepartmentCode,
                    @CourseCode,
                    @ProgramCode,
                    @LevelCode,
                    @SemesterCode,
                    @CreditLowerBound,
                    @CreditUpperBound,
                    @ModeOfStudyCode,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [SetUp].[CourseRegulation]
                SET
                    [UniversityCode]=@UniversityCode,
                    [FacultyCode]=@FacultyCode,
                    [DepartmentCode]=@DepartmentCode,
                    [CourseCode]=@CourseCode,
                    [ProgramCode]=@ProgramCode,
                    [LevelCode]=@LevelCode,
                    [SemesterCode]=@SemesterCode,
                    [CreditLowerBound]=@CreditLowerBound,
                    [CreditUpperBound]=@CreditUpperBound,
                    [ModeOfStudyCode]=@ModeOfStudyCode,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPCourseRegulationInsertUpdate] TO PUBLIC
GO
```

```
/****** Object:  StoredProcedure [SetUp].[SPCourseRegulationDeletePermanently]    Script  ↙
    Date: 08/16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for deleting values from SetUp.CourseRegulation  ↙
    Table

CREATE PROC [SetUp].[SPCourseRegulationDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[CourseRegulation] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [SetUp].[CourseRegulation]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPCourseRegulationDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPCourseRegulationDelete]    Script Date: 08/16/↙
    2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/27/2011
--Last Updated        :   07/27/2011
```

```
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from SetUp.    ↙
    CourseRegulation Table

CREATE PROC [SetUp].[SPCourseRegulationDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[CourseRegulation] WHERE (([Code] = @Code)   ↙
    AND (Deleted=@Deleted)))
            BEGIN
            UPDATE [SetUp].[CourseRegulation]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPCourseRegulationDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Assessment].[SPDefinitionSelect]    Script Date: 08/16/ ↙
    2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
```

```
--Description       :   Stored procedure for retrieving values from Assessment.Definition ↙
    Table

CREATE PROC [Assessment].[SPDefinitionSelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Deleted BIT=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Assessment].[Definition] WHERE (([Code] = @Code) AND (   ↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted)))
            BEGIN
            SELECT * FROM [Assessment].[Definition]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Assessment].[Definition]
            WHERE
                (
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Assessment].[SPDefinitionSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Assessment].[SPDefinitionInsertUpdate]    Script Date:   ↙
    08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
```

```
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Assessment.Definition    ↙
    Table

CREATE PROC [Assessment].[SPDefinitionInsertUpdate]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @FacultyCode BIGINT=NULL,
    @DepartmentCode BIGINT=NULL,
    @CourseCode BIGINT=NULL,
    @ProgramCode BIGINT=NULL,
    @SessionCode BIGINT=NULL,
    @SemesterCode BIGINT=NULL,
    @AssessmentTypeCode NVARCHAR(50)=NULL,
    @Value DECIMAL(18,2)=NULL,
    @NoOfAssessment INT=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Assessment].[Definition] WHERE ([Code] = @Code) AND ↙
    ([UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
                INSERT INTO [Assessment].[Definition]
                (
                    [UniversityCode],
                    [FacultyCode],
                    [DepartmentCode],
                    [CourseCode],
                    [ProgramCode],
                    [SessionCode],
                    [SemesterCode],
                    [AssessmentTypeCode],
                    [Value],
                    [NoOfAssessment],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]

                )
                VALUES
                (
                    @UniversityCode,
                    @FacultyCode,
                    @DepartmentCode,
                    @CourseCode,
                    @ProgramCode,
                    @SessionCode,
                    @SemesterCode,
                    @AssessmentTypeCode,
                    @Value,
                    @NoOfAssessment,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted

                )
            END
        ELSE
            BEGIN
                UPDATE [Assessment].[Definition]
                SET
                    [UniversityCode]=@UniversityCode,
                    [FacultyCode]=@FacultyCode,
                    [DepartmentCode]=@DepartmentCode,
                    [CourseCode]=@CourseCode,
                    [ProgramCode]=@ProgramCode,
                    [SessionCode]=@SessionCode,
                    [SemesterCode]=@SemesterCode,
                    [AssessmentTypeCode]=@AssessmentTypeCode,
                    [Value]=@Value,
                    [NoOfAssessment]=@NoOfAssessment,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted

                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Assessment].[SPDefinitionInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Assessment].[SPDefinitionDeletePermanently]    Script   ↙
    Date: 08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Assessment.Definition   ↙
    Table

CREATE PROC [Assessment].[SPDefinitionDeletePermanently]
(
    @Code BIGINT=NULL
```

```
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Assessment].[Definition] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Assessment].[Definition]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Assessment].[SPDefinitionDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Assessment].[SPDefinitionDelete]    Script Date: 08/16/ ↙
    2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/18/2011
--Last Updated        :   07/18/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for deleting values temporarily from Assessment. ↙
    Definition Table

CREATE PROC [Assessment].[SPDefinitionDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Assessment].[Definition] WHERE ([Code] = @Code) AND     ↙
    (Deleted=@Deleted))
            BEGIN
            UPDATE [Assessment].[Definition]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Assessment].[SPDefinitionDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPDeclarationsSelect]    Script Date: 08/16/↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for retrieving values from Personal.Declarations ↙
    Table

CREATE PROC [Personals].[SPDeclarationsSelect]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Declarations] WHERE (([Code] = @Code) AND ( ↙
    [AccountCode]=@AccountCode) AND ([ScreenCode]=@ScreenCode) AND ([Deleted]=@Deleted)))
```

```
            BEGIN
            SELECT * FROM [Personals].[Declarations]
            WHERE
                (
                ([Code] = @Code) AND
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Personals].[Declarations]
            WHERE
                (
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPDeclarationsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPDeclarationsInsertUpdate]    Script Date: ↙
    08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Personals.Declarations    ↙
    Table

CREATE PROC [Personals].[SPDeclarationsInsertUpdate]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50)=NULL,
    @Notes NVARCHAR(1000)=NULL,
    @GuardianCode INT=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Personals].[Declarations] WHERE (([Code] = @Code)   ↙
    AND ([AccountCode]=@AccountCode) AND ([Deleted]=@Deleted)))
            BEGIN
                INSERT INTO [Personals].[Declarations]
                (
                    [AccountCode],
                    [ScreenCode],
                    [Notes],
                    [GuardianCode],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    @AccountCode,
                    @ScreenCode,
                    @Notes,
                    @GuardianCode,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [Personals].[Declarations]
                SET
                    [AccountCode]=@AccountCode,
                    [ScreenCode]=@ScreenCode,
                    [Notes]=@Notes,
                    [GuardianCode]=@GuardianCode,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([AccountCode]=@AccountCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPDeclarationsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPDeclarationsDeletePermanently]    Script  ↙
    Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for deleting values from Personal.Declarations   ↙
    Table

CREATE PROC [Personals].[SPDeclarationsDeletePermanently]
(
    @Code INT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Declarations] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Personals].[Declarations]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPDeclarationsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPDeclarationsDelete]    Script Date: 08/16/↙
    2011 01:24:28 ******/
```

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for deleting values temporarily from Personal.   ↙
    Declarations Table

CREATE PROC [Personals].[SPDeclarationsDelete]
(
    @Code INT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Declarations] WHERE (([Code] = @Code) AND   ↙
    (Deleted=@Deleted)))
            BEGIN
            UPDATE [Personals].[Declarations]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPDeclarationsDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPAreasOfSpecializationsSelect]    Script Date: ↙
    08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
```

```
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from SetUp.          ↙
    AreasOfSpecializations Table

CREATE PROC [SetUp].[SPAreasOfSpecializationsSelect]
(
    @Code INT=NULL,
    @DescriptionsCode NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[AreasOfSpecializations] WHERE ([Code] = @Code))
            BEGIN
            SELECT * FROM [SetUp].[AreasOfSpecializations]
            WHERE
                (
                ([Code] = @Code) AND
                ([DescriptionsCode] = @DescriptionsCode)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [SetUp].[AreasOfSpecializations]
            WHERE
                (
                [DescriptionsCode] = @DescriptionsCode
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPAreasOfSpecializationsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPAreasOfSpecializationsInsertUpdate]    Script ↙
    Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
```

```
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for saving values into SetUp.             ↙
    AreasOfSpecializations Table

CREATE PROC [SetUp].[SPAreasOfSpecializationsInsertUpdate]
(
    @Code INT=NULL,
    @DescriptionsCode NVARCHAR(50)=NULL,
    @Specialization NVARCHAR(256)=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[AreasOfSpecializations]  WHERE ([Code] = @  ↙
    Code) )
            BEGIN
                INSERT INTO [SetUp].[AreasOfSpecializations]
                (
                    [Code],
                    [DescriptionsCode],
                    [Specialization]
                )
                VALUES
                (
                    @Code,
                    @DescriptionsCode,
                    @Specialization
                )
            END
        ELSE
            BEGIN
                UPDATE [SetUp].[AreasOfSpecializations]
                SET
                    [DescriptionsCode]=@DescriptionsCode,
                    [Specialization]=@Specialization
                WHERE
                    ([Code] = @Code)
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);
```

```
END CATCH

GRANT EXECUTE ON [SetUp].[SPAreasOfSpecializationsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPAreasOfSpecializationsDeletePermanently]     ↙
    Script Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for deleting values from SetUp.                ↙
    AreasOfSpecializations Table

CREATE PROC [SetUp].[SPAreasOfSpecializationsDeletePermanently]
(
    @Code INT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[AreasOfSpecializations] WHERE (([Code] = @      ↙
    Code)))
            BEGIN
            DELETE FROM [SetUp].[AreasOfSpecializations]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPAreasOfSpecializationsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPAreasOfExpertisesSelect]    Script Date:   ↙
    08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
```

```sql
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Personal.
    AreasOfExpertises Table

CREATE PROC [Personals].[SPAreasOfExpertisesSelect]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[AreasOfExpertises] WHERE (([Code] = @Code)
    AND ([AccountCode]=@AccountCode) AND ([ScreenCode]=@ScreenCode) AND ([Deleted]=@
    Deleted)))
            BEGIN
            SELECT * FROM [Personals].[AreasOfExpertises]
            WHERE
                (
                ([Code] = @Code) AND
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Personals].[AreasOfExpertises]
            WHERE
                (
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH
```

```
GRANT EXECUTE ON [Personals].[SPAreasOfExpertisesSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPAreasOfExpertisesInsertUpdate]    Script ↙
    Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Personals.AreaOfExpertises↙
     Table

CREATE PROC [Personals].[SPAreasOfExpertisesInsertUpdate]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @JobCategoryCode NVARCHAR(50)=NULL,
    @AOSCode NVARCHAR(50)=NULL,
    @YearOfExperience NVARCHAR(50)=NULL,
    @LeveOfExperience NVARCHAR(50)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Personals].[AreasOfExpertises] WHERE (([Code] = @   ↙
    Code) AND ([AccountCode]=@AccountCode) AND ([Deleted]=@Deleted)))
            BEGIN
                INSERT INTO [Personals].[AreasOfExpertises]
                (
                    [AccountCode],
                    [ScreenCode],
                    [JobCategoryCode],
                    [AOSCode],
                    [YearOfExperience],
                    [LeveOfExperience],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    @AccountCode,
                    @ScreenCode,
                    @JobCategoryCode,
                    @AOSCode,
                    @YearOfExperience,
                    @LeveOfExperience,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [Personals].[AreasOfExpertises]
                SET
                    [AccountCode]=@AccountCode,
                    [ScreenCode]=@ScreenCode,
                    [JobCategoryCode]=@JobCategoryCode,
                    [AOSCode]=@AOSCode,
                    [YearOfExperience]=@YearOfExperience,
                    [LeveOfExperience]=@LeveOfExperience,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([AccountCode]=@AccountCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPAreasOfExpertisesInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPAreasOfExpertisesDeletePermanently]       ↙
    Script Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Personal.               ↙
    AreasOfExpertises Table

CREATE PROC [Personals].[SPAreasOfExpertisesDeletePermanently]
(
    @Code INT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[AreasOfExpertises] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Personals].[AreasOfExpertises]
```

```sql
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPAreasOfExpertisesDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPAreasOfExpertisesDelete]    Script Date:  ↙
    08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for deleting values temporarily from Personal.  ↙
    AreasOfExpertises Table

CREATE PROC [Personals].[SPAreasOfExpertisesDelete]
(
    @Code INT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[AreasOfExpertises] WHERE (([Code] = @Code)  ↙
    AND (Deleted=@Deleted)))
            BEGIN
            UPDATE [Personals].[AreasOfExpertises]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
```

```
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPAreasOfExpertisesDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPApprovalsSelect]    Script Date: 08/16/2011 01↙
    :24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from SetUp.Approvals Table

CREATE PROC [SetUp].[SPApprovalsSelect]
(
    @Code UNIQUEIDENTIFIER=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Approvals] WHERE ([Code] = @Code) AND (         ↙
    [UniversityCode] = @UniversityCode) AND ([ScreenCode]=@ScreenCode))
            BEGIN
            SELECT * FROM [SetUp].[Approvals]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([ScreenCode] = @ScreenCode)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [SetUp].[Approvals]
            WHERE
                (
                ([UniversityCode] = @UniversityCode) AND
                ([ScreenCode] = @ScreenCode)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPApprovalsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPApprovalsInsertUpdate]    Script Date: 08/16/ ↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into SetUp.Approvals Table

CREATE PROC [SetUp].[SPApprovalsInsertUpdate]
(
    @Code UNIQUEIDENTIFIER=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @FacultyCode BIGINT=NULL,
    @DepartmentCode BIGINT=NULL,
    @CourseCode BIGINT=NULL,
    @ProgramCode INT=NULL,
    @SessionCode NVARCHAR(50)=NULL,
    @ModuleCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50)=NULL,
    @SemesterCode NVARCHAR(50)=NULL,
    @LevelCode INT=NULL,
    @RoleCode UNIQUEIDENTIFIER=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @EntityCode NVARCHAR(50)=NULL,
    @ApprovalType NVARCHAR(50)=NULL,
    @Order INT=NULL,
    @Approved BIT=NULL,
    @ApprovedOn DATETIME=NULL,
    @ApprovedBy NVARCHAR(50)=NULL,
    @ApprovedNotes NVARCHAR(1000)=NULL,
    @Requested BIT=NULL,
    @RequestedOn DATETIME=NULL,
    @RequestedBy NVARCHAR(50)=NULL,
    @RequestedNotes NVARCHAR(1000)=NULL,
    @Rejected BIT=NULL,
    @RejectedOn DATETIME=NULL,
    @RejectedBy NVARCHAR(50)=NULL,
    @RejectedNotes NVARCHAR(1000)=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[Approvals]  WHERE ([Code] = @Code) AND (     ↙
    [UniversityCode] = @UniversityCode))
            BEGIN
                INSERT INTO [SetUp].[Approvals]
                (
                    [Code],
                    [UniversityCode],
                    [FacultyCode],
                    [DepartmentCode],
                    [CourseCode],
                    [ProgramCode],
                    [SessionCode],
                    [ModuleCode],
                    [ScreenCode],
                    [SemesterCode],
                    [LevelCode],
                    [RoleCode],
                    [AccountCode],
                    [EntityCode],
                    [ApprovalType],
                    [Order],
                    [Approved],
                    [ApprovedOn],
                    [ApprovedBy],
                    [ApprovedNotes],
                    [Requested],
                    [RequestedOn],
                    [RequestedBy],
                    [RequestedNotes],
                    [Rejected],
                    [RejectedOn],
                    [RejectedBy],
                    [RejectedNotes]
                )
                VALUES
                (
                    @Code,
                    @UniversityCode,
                    @FacultyCode,
                    @DepartmentCode,
                    @CourseCode,
                    @ProgramCode,
                    @SessionCode,
                    @ModuleCode,
                    @ScreenCode,
                    @SemesterCode,
                    @LevelCode,
                    @RoleCode,
                    @AccountCode,
                    @EntityCode,
                    @ApprovalType,
                    @Order,
                    @Approved,
                    @ApprovedOn,
                    @ApprovedBy,
                    @ApprovedNotes,
                    @Requested,
                    @RequestedOn,
                    @RequestedBy,
                    @RequestedNotes,
                    @Rejected,
                    @RejectedOn,
                    @RejectedBy,
                    @RejectedNotes
                )
            END
        ELSE
            BEGIN
                UPDATE [SetUp].[Approvals]
                SET
                    [UniversityCode] = @UniversityCode,
                    [FacultyCode] = @FacultyCode,
                    [DepartmentCode]=@DepartmentCode,
                    [CourseCode]=@CourseCode,
                    [ProgramCode]=@ProgramCode,
                    [SessionCode]=@SessionCode,
                    [ModuleCode]=@ModuleCode,
                    [ScreenCode]=@ScreenCode,
                    [SemesterCode]=@SemesterCode,
                    [LevelCode]=@LevelCode,
                    [RoleCode]=@RoleCode,
                    [AccountCode]=@AccountCode,
                    [EntityCode]=@EntityCode,
                    [ApprovalType]=@ApprovalType,
                    [Order]=@Order,
                    [Approved]=@Approved,
                    [ApprovedOn]=@ApprovedOn,
                    [ApprovedBy]=@ApprovedBy,
                    [ApprovedNotes]=@ApprovedNotes,
                    [Requested]=@Requested,
                    [RequestedOn]=@RequestedOn,
                    [RequestedBy]=@RequestedBy,
                    [RequestedNotes]=@RequestedNotes,
                    [Rejected]=@Rejected,
                    [RejectedOn]=@RejectedOn,
                    [RejectedBy]=@RejectedBy,
                    [RejectedNotes]=@RejectedNotes
                WHERE
                    ([Code] = @Code)
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()
```

```
RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPApprovalsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPApprovalsDeletePermanently]    Script Date: 08↙
    /16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from SetUp.Approvals Table

CREATE PROC [SetUp].[SPApprovalsDeletePermanently]
(
    @Code UNIQUEIDENTIFIER=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Approvals] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [SetUp].[Approvals]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPApprovalsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPDescriptionsSelect]    Script Date: 08/16/2011↙
     01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
```

```
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from SetUp.Descriptions   ↙
    Table

CREATE PROC [SetUp].[SPDescriptionsSelect]
(
    @Code NVARCHAR(50)=NULL,
    @ParametersCode NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Descriptions] WHERE ([Code] = @Code) AND (      ↙
    [ParametersCode] = @ParametersCode))
            BEGIN
            SELECT * FROM [SetUp].[Descriptions]
            WHERE
                (
                ([Code] = @Code) AND
                ([ParametersCode] = @ParametersCode)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [SetUp].[Descriptions]
            WHERE
                (
                ([ParametersCode] = @ParametersCode)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPDescriptionsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPDescriptionsInsertUpdate]    Script Date: 08/ ↙
    16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
```

```
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into SetUp.Descriptions Table

CREATE PROC [SetUp].[SPDescriptionsInsertUpdate]
(
    @ID INT=NULL,
    @Code Nvarchar(50)=NULL,
    @ParametersCode NVARCHAR(50)=NULL,
    @Name NVARCHAR(500)=NULL,
    @Notes NVARCHAR(1000)=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[Descriptions]  WHERE (([Code] = @Code) AND (↙
    [ParametersCode] = @ParametersCode)))
            BEGIN
                INSERT INTO [SetUp].[Descriptions]
                (
                    [ParametersCode],
                    [Name],
                    [Notes]

                )
                VALUES
                (
                    @ParametersCode,
                    @Name,
                    @Notes

                )
            END
        ELSE
            BEGIN
                UPDATE [SetUp].[Descriptions]
                SET
                    [Name]=@Name,
                    [Notes]=@Notes

                WHERE
                    (([Code] = @Code) AND ([ParametersCode] = @ParametersCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPDescriptionsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPDescriptionsDeletePermanently]    Script Date:↙
     08/16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/12/2011
--Last Updated      :   07/12/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from SetUp.Descriptions Table

CREATE PROC [SetUp].[SPDescriptionsDeletePermanently]
(
    @Code NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Descriptions] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [SetUp].[Descriptions]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPDescriptionsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPFacultiesSelect]    Script Date: 08/16/2011 01↙
    :24:29 ******/
SET ANSI_NULLS ON
```

```
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from SetUp.Faculties Table

CREATE PROC [SetUp].[SPFacultiesSelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Faculties] WHERE ([Code] = @Code) AND (         ↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
            SELECT * FROM [SetUp].[Faculties]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [SetUp].[Faculties]
            WHERE
                (
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPFacultiesSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPFacultiesInsertUpdate]    Script Date: 08/16/ ↙
```

```
    2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into SetUp.Faculties Table

CREATE PROC [SetUp].[SPFacultiesInsertUpdate]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Acronym NVARCHAR(50)=NULL,
    @Description NVARCHAR(500)=NULL,
    @Notes NVARCHAR(1000)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[Faculties]  WHERE ([Code] = @Code) AND (     ↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
                INSERT INTO [SetUp].[Faculties]
                (
                    [UniversityCode],
                    [Acronym],
                    [Description],
                    [Notes],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]

                )
                VALUES
                (
                    @UniversityCode,
                    @Acronym,
                    @Description,
                    @Notes,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [SetUp].[Faculties]
                SET
                    [UniversityCode]=@UniversityCode,
                    [Acronym]=@Acronym,
                    [Description]=@Description,
                    [Notes]=@Notes,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPFacultiesInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPFacultiesDeletePermanently]    Script Date: 08↙
    /16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/12/2011
--Last Updated      :   07/12/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from SetUp.Faculties Table

CREATE PROC [SetUp].[SPFacultiesDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Faculties] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [SetUp].[Faculties]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN
```

```
DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPFacultiesDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPFacultiesDelete]    Script Date: 08/16/2011 01↙
    :24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/27/2011
--Last Updated      :   07/27/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from SetUp.      ↙
    Faculties Table

CREATE PROC [SetUp].[SPFacultiesDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[Faculties] WHERE (([Code] = @Code) AND      ↙
    (Deleted=@Deleted)))
            BEGIN
            UPDATE [SetUp].[Faculties]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPFacultiesDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Registry].[SPExamsSelect]    Script Date: 08/16/2011 01:↙
    24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Registry.Exams Table

CREATE PROC [Registry].[SPExamsSelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Registry].[Exams] WHERE (([Code] = @Code) AND (         ↙
    [UniversityCode]=@UniversityCode) AND ([Deleted]=@Deleted)))
            BEGIN
            SELECT * FROM [Registry].[Exams]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted]=@Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Registry].[Exams]
            WHERE
                (
                ([UniversityCode]=@UniversityCode) AND
                ([Deleted]=@Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN
```

```
DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Registry].[SPExamsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Registry].[SPExamsInsertUpdate]    Script Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Registry.Exams Table

CREATE PROC [Registry].[SPExamsInsertUpdate]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @TypeCode NVARCHAR(50)=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @FacultyCode BIGINT=NULL,
    @DepartmentCode BIGINT=NULL,
    @CourseCode BIGINT=NULL,
    @ProgramCode BIGINT=NULL,
    @AwardCode NVARCHAR(50)=NULL,
    @LevelCode INT=NULL,
    @EntryMode NVARCHAR(50)=NULL,
    @StudyMode NVARCHAR(50)=NULL,
    @SessionCode BIGINT=NULL,
    @DateApplied DATETIME2(7)=NULL,
    @Notes NVARCHAR(256)=NULL,
    @NoOfSitting INT=NULL,
    @NoOfCredits INT=NULL,
    @NoOfPasses INT=NULL,
    @NoOfDistinctions INT=NULL,
    @NoOfFails INT=NULL,
    @Total INT=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
```

```
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Registry].[Exams] WHERE ([Code] = @Code) AND (       ↙
    [UniversityCode]=@UniversityCode) AND ([Deleted]=@Deleted))
            BEGIN
                INSERT INTO [Registry].[Exams]
                (
                    [UniversityCode],
                    [AccountCode],
                    [FacultyCode],
                    [DepartmentCode],
                    [CourseCode],
                    [ProgramCode],
                    [AwardCode],
                    [LevelCode],
                    [EntryMode],
                    [StudyMode],
                    [SessionCode],
                    [DateApplied],
                    [Notes],
                    [NoOfSitting],
                    [NoOfCredits],
                    [NoOfPasses],
                    [NoOfDistinctions],
                    [NoOfFails],
                    [Total],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    @UniversityCode,
                    @AccountCode,
                    @FacultyCode,
                    @DepartmentCode,
                    @CourseCode,
                    @ProgramCode,
                    @AwardCode,
                    @LevelCode,
                    @EntryMode,
                    @StudyMode,
                    @SessionCode,
                    @DateApplied,
                    @Notes,
                    @NoOfSitting,
                    @NoOfCredits,
                    @NoOfPasses,
                    @NoOfDistinctions,
                    @NoOfFails,
                    @Total,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [Registry].[Exams]
                SET
                    [UniversityCode]=@UniversityCode,
                    [AccountCode]=@AccountCode,
                    [FacultyCode]=@FacultyCode,
                    [DepartmentCode]=@DepartmentCode,
                    [CourseCode]=@CourseCode,
                    [ProgramCode]=@ProgramCode,
                    [AwardCode]=@AwardCode,
                    [LevelCode]=@LevelCode,
                    [EntryMode]=@EntryMode,
                    [StudyMode]=@StudyMode,
                    [SessionCode]=@SessionCode,
                    [DateApplied]=@DateApplied,
                    [Notes]=@Notes,
                    [NoOfSitting]=@NoOfSitting,
                    [NoOfCredits]=@NoOfCredits,
                    [NoOfPasses]=@NoOfPasses,
                    [NoOfDistinctions]=@NoOfDistinctions,
                    [NoOfFails]=@NoOfFails,
                    [Total]=@Total,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Registry].[SPExamsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Registry].[SPExamsDeletePermanently]    Script Date: 08/↙
    16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/18/2011
--Last Updated        :   07/18/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for deleting values from Registry.Exams Table

CREATE PROC [Registry].[SPExamsDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Registry].[Exams] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Registry].[Exams]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Registry].[SPExamsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Registry].[SPExamsDelete]    Script Date: 08/16/2011 01:↙
    24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Registry.   ↙
    Exams Table

CREATE PROC [Registry].[SPExamsDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Registry].[Exams] WHERE (([Code] = @Code) AND (Deleted=@↙
    Deleted)))
            BEGIN
            UPDATE [Registry].[Exams]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Registry].[SPExamsDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPModulesSelect]    Script Date: 08/16/2011 01: ↙
    24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from SetUp.Modules Table

CREATE PROC [SetUp].[SPModulesSelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Modules] WHERE ([Code] = @Code) AND (      ↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
            SELECT * FROM [SetUp].[Modules]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
        ELSE
```

```
            BEGIN
            SELECT * FROM [SetUp].[Modules]
            WHERE
                (
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPModulesSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPModulesInsertUpdate]    Script Date: 08/16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into SetUp.Modules Table

CREATE PROC [SetUp].[SPModulesInsertUpdate]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Description NVARCHAR(256)=NULL,
    @Url NVARCHAR(256)=NULL,
    @Notes NVARCHAR(500)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;
```

```
        IF NOT EXISTS (SELECT * FROM [SetUp].[Modules]  WHERE ([Code] = @Code) AND ( ↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
                INSERT INTO [SetUp].[Modules]
                (
                    [UniversityCode],
                    [Description],
                    [Url],
                    [Notes],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]

                )
                VALUES
                (
                    @UniversityCode,
                    @Description,
                    @Url,
                    @Notes,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [SetUp].[Modules]
                SET
                    [UniversityCode]=@UniversityCode,
                    [Description]=@Description,
                    [Url]=@Url,
                    [Notes]=@Notes,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPModulesInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPModulesDeletePermanently]    Script Date: 08/ ↙
    16/2011 01:24:29 ******/
SET ANSI_NULLS ON
```

```
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/12/2011
--Last Updated        :   07/12/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for deleting values from SetUp.Modules Table

CREATE PROC [SetUp].[SPModulesDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Modules] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [SetUp].[Modules]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPModulesDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPModulesDelete]    Script Date: 08/16/2011 01: ↙
    24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/27/2011
--Last Updated        :   07/27/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for deleting values temporarily from SetUp.     ↙
    Modules Table
```

```
CREATE PROC [SetUp].[SPModulesDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[Modules] WHERE (([Code] = @Code) AND          ↙
    (Deleted=@Deleted)))
            BEGIN
            UPDATE [SetUp].[Modules]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPModulesDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPGradingSystemSelect]    Script Date: 08/16/   ↙
    2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for retrieving values from SetUp.GradingSystem   ↙
    Table

CREATE PROC [SetUp].[SPGradingSystemSelect]
```

```
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[GradingSystem] WHERE ([Code] = @Code) AND (     ↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
            SELECT * FROM [SetUp].[GradingSystem]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [SetUp].[GradingSystem]
            WHERE
                (
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPGradingSystemSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPGradingSystemInsertUpdate]    Script Date: 08/↙
    16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into SetUp.GradingSystem Table
```

```sql
CREATE PROC [SetUp].[SPGradingSystemInsertUpdate]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @ScoreLowerBound DECIMAL(18,2)=NULL,
    @ScoreUpperBound DECIMAL(18,2)=NULL,
    @Description NVARCHAR(2)=NULL,
    @CGPALowerBound DECIMAL(18,2)=NULL,
    @CGPAUpperBound DECIMAL(18,2)=NULL,
    @Notes NVARCHAR(500)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[GradingSystem]  WHERE ([Code] = @Code) AND (↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
                INSERT INTO [SetUp].[GradingSystem]
                (
                    [UniversityCode],
                    [ScoreLowerBound],
                    [ScoreUpperBound],
                    [Description],
                    [CGPALowerBound],
                    [CGPAUpperBound],
                    [Notes],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]

                )
                VALUES
                (
                    @UniversityCode,
                    @ScoreLowerBound,
                    @ScoreUpperBound,
                    @Description,
                    @CGPALowerBound,
                    @CGPAUpperBound,
                    @Notes,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [SetUp].[GradingSystem]
                SET
                    [UniversityCode]=@UniversityCode,
                    [ScoreLowerBound]=@ScoreLowerBound,
                    [ScoreUpperBound]=@ScoreUpperBound,
                    [Description]=@Description,
                    [CGPALowerBound]=@CGPALowerBound,
                    [CGPAUpperBound]=@CGPAUpperBound,
                    [Notes]=@Notes,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPGradingSystemInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPGradingSystemDeletePermanently]    Script Date↙
    : 08/16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/12/2011
--Last Updated        :   07/12/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for deleting values from SetUp.GradingSystem     ↙
    Table

CREATE PROC [SetUp].[SPGradingSystemDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[GradingSystem] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [SetUp].[GradingSystem]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
```

```sql
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPGradingSystemDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPGradingSystemDelete]    Script Date: 08/16/   ↙
    2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/27/2011
--Last Updated      :   07/27/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from SetUp.     ↙
    GradingSystem Table

CREATE PROC [SetUp].[SPGradingSystemDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[GradingSystem] WHERE (([Code] = @Code) AND  ↙
    (Deleted=@Deleted)))
            BEGIN
            UPDATE [SetUp].[GradingSystem]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN
```

```
DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPGradingSystemDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPFeeDefinitionSelect]    Script Date: 08/16/ ↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/18/2011
--Last Updated        :   07/18/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for retrieving values from Finance.FeeDefinition ↙
    Table

CREATE PROC [Finance].[SPFeeDefinitionSelect]
(
    @Code NVARCHAR(50)=NULL,
    @UniversityCode NVARCHAR(50)=NULL
    )
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance].[FeeDefinition] WHERE ([Code] = @Code) AND (   ↙
    [UniversityCode]=@UniversityCode))
            BEGIN
            SELECT * FROM [Finance].[FeeDefinition]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Finance] .[FeeDefinition]
            WHERE
                (
                ([UniversityCode]=@UniversityCode)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN
```

```sql
DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPFeeDefinitionSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPFeeDefinitionInsertUpdate]    Script Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Finance.FeeDefinition Table

CREATE PROC [Finance].[SPFeeDefinitionInsertUpdate]
(
    @Code NVARCHAR(50)=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Description NVARCHAR(256)=NULL,
    @FeeStatusCode NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Finance].[FeeDefinition] WHERE ([Code] = @Code) AND ([UniversityCode]=@UniversityCode))
            BEGIN
                INSERT INTO [Finance].[FeeDefinition]
                (
                    [Code],
                    [UniversityCode],
                    [Description],
                    [FeeStatusCode]

                )
                VALUES
                (
                    @Code,
                    @UniversityCode,
                    @Description,
                    @FeeStatusCode
                )
            END
        ELSE
            BEGIN
```

```
                UPDATE [Finance].[FeeDefinition]
                SET

                    [UniversityCode]=@UniversityCode,
                    [Description]=@Description,
                    [FeeStatusCode] =@FeeStatusCode

                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPFeeDefinitionInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPFeeDefinitionDeletePermanently]    Script   ↙
    Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/18/2011
--Last Updated        :   07/18/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for deleting values from Finance.FeeDefinition   ↙
    Table

CREATE PROC [Finance].[SPFeeDefinitionDeletePermanently]
(
    @Code NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance] .[FeeDefinition]  WHERE ([Code] = @Code))
            BEGIN
            DELETE FROM [Finance] .[FeeDefinition]
            WHERE
                (
                    ([Code] = @Code)
                )
            END
```

```
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPFeeDefinitionDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPSubjectRequirementDetailsSelect]    Script  ↙
    Date: 08/16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from SetUp.           ↙
    SubjectRequirementDetails Table

CREATE PROC [SetUp].[SPSubjectRequirementDetailsSelect]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[SubjectRequirementDetails] WHERE ([Code] = @  ↙
    Code)AND ([Deleted] = @Deleted))
            BEGIN
            SELECT * FROM [SetUp].[SubjectRequirementDetails]
            WHERE
                (
                ([Code] = @Code) AND
                ([Deleted] = @Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [SetUp].[SubjectRequirementDetails]
            WHERE
                (
                ([Deleted] = @Deleted)
                )
            END
```

```
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPSubCoursesSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPSubjectRequirementDetailsInsertUpdate]        ↙
    Script Date: 08/16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into SetUp.                   ↙
    SubjectRequirementDetails Table

CREATE PROC [SetUp].[SPSubjectRequirementDetailsInsertUpdate]
(
    @Code BIGINT=NULL,
    @SubjectRequirementCode BIGINT=NULL,
    @GradeCode NVARCHAR(50)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[SubjectRequirementDetails]  WHERE ([Code] = ↙
    @Code) AND ([Deleted] = @Deleted))
            BEGIN
                INSERT INTO [SetUp].[SubjectRequirementDetails]
                (
                    [SubjectRequirementCode],
                    [GradeCode],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]

                )
                VALUES
                (
                    @SubjectRequirementCode,
                    @GradeCode,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [SetUp].[SubjectRequirementDetails]
                SET
                    [SubjectRequirementCode]=@SubjectRequirementCode,
                    [GradeCode]=@GradeCode,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    ([Code] = @Code)
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPSubjectRequirementDetailsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPSubjectRequirementDetailsDeletePermanently]   ↙
     Script Date: 08/16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/12/2011
--Last Updated      :   07/12/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from SetUp.                   ↙
    SubjectRequirementDetails Table

CREATE PROC [SetUp].[SPSubjectRequirementDetailsDeletePermanently]
(
    @Code BIGINT=NULL
)
```

```
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[SubjectRequirementDetails] WHERE (([Code] = @  ↙
    Code)))
            BEGIN
            DELETE FROM [SetUp].[SubjectRequirementDetails]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPSubjectRequirementDetailsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPSubjectRequirementDetailsDelete]    Script   ↙
    Date: 08/16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/28/2011
--Last Updated      :   07/28/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from SetUp.    ↙
    SubjectRequirementDetails Table

CREATE PROC [SetUp].[SPSubjectRequirementDetailsDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[SubjectRequirementDetails] WHERE (([Code] = ↙
    @Code) AND (Deleted=@Deleted)))
            BEGIN
            UPDATE [SetUp].[SubjectRequirementDetails]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPSubjectRequirementDetailsDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Registry].[SPStudentsSelect]    Script Date: 08/16/2011 ↙
    01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author             :   Ademola Adebo
--Reviewer           :   Godwin Mathias
--Date Created       :   07/18/2011
--Last Updated       :   07/18/2011
--Last Updated By    :   Ademola Adebo
--Description        :   Stored procedure for retrieving values from Registry.Students     ↙
    Table

CREATE PROC [Registry].[SPStudentsSelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @MatricNo NVARCHAR(50)=NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Registry].[Students] WHERE (([Code] = @Code) AND (       ↙
    [UniversityCode]=@UniversityCode) AND ([MatricNo] = @MatricNo) AND ([Deleted]=@       ↙
    Deleted)))
            BEGIN
            SELECT * FROM [Registry].[Students]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([MatricNo] = @MatricNo) AND
                ([Deleted]=@Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Registry].[Students]
            WHERE
                (
                ([UniversityCode]=@UniversityCode) AND
                ([MatricNo] = @MatricNo) AND
                ([Deleted]=@Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Registry].[SPStudentsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Registry].[SPStudentsInsertUpdate]    Script Date: 08/16↙
    /2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Registry.Students Table

CREATE PROC [Registry].[SPStudentsInsertUpdate]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @FacultyCode BIGINT=NULL,
    @DepartmentCode BIGINT=NULL,
    @ProgramCode BIGINT=NULL,
    @CourseCode BIGINT=NULL,
    @LevelCode INT=NULL,
    @MatricNo NVARCHAR(50)=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @EntryMode NVARCHAR(50)=NULL,
    @StudyMode NVARCHAR(50)=NULL,
    @SessionCode BIGINT=NULL,
    @AdmittedOn DATE=NULL,
    @AwardCode NVARCHAR(50)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Registry].[Students] WHERE ([Code] = @Code) AND (   ↙
    [UniversityCode]=@UniversityCode) AND ([MatricNo]=@MatricNo) AND ([Deleted]=@Deleted))
            BEGIN
                INSERT INTO [Registry].[Students]
                (
                    [UniversityCode],
                    [AccountCode],
                    [FacultyCode],
                    [DepartmentCode],
                    [CourseCode],
                    [ProgramCode],
                    [AwardCode],
                    [LevelCode],
                    [EntryMode],
                    [StudyMode],
                    [SessionCode],
                    [AdmittedOn],
                    [MatricNo],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    @UniversityCode,
                    @AccountCode,
                    @FacultyCode,
                    @DepartmentCode,
                    @CourseCode,
                    @ProgramCode,
                    @AwardCode,
                    @LevelCode,
                    @EntryMode,
                    @StudyMode,
                    @SessionCode,
                    @AdmittedOn,
                    @MatricNo,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [Registry].[Students]
                SET
```

```
                    [UniversityCode]=@UniversityCode,
                    [AccountCode]=@AccountCode,
                    [FacultyCode]=@FacultyCode,
                    [DepartmentCode]=@DepartmentCode,
                    [CourseCode]=@CourseCode,
                    [ProgramCode]=@ProgramCode,
                    [AwardCode]=@AwardCode,
                    [LevelCode]=@LevelCode,
                    [EntryMode]=@EntryMode,
                    [StudyMode]=@StudyMode,
                    [SessionCode]=@SessionCode,
                    [AdmittedOn]=@AdmittedOn,
                    [MatricNo]=@MatricNo,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode) AND (      ↙
    [MatricNo] = @MatricNo))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Registry].[SPStudentsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Registry].[SPStudentsDeletePermanently]    Script Date: ↙
    08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Registry.Students Table

CREATE PROC [Registry].[SPStudentsDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;
```

```
        IF EXISTS (SELECT * FROM [Registry].[Students] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Registry].[Students]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Registry].[SPStudentsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Registry].[SPStudentsDelete]    Script Date: 08/16/2011 ↙
    01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/18/2011
--Last Updated        :   07/18/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for deleting values temporarily from Registry.   ↙
    Students Table

CREATE PROC [Registry].[SPStudentsDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Registry].[Students] WHERE (([Code] = @Code) AND        ↙
    (Deleted=@Deleted)))
            BEGIN
            UPDATE [Registry].[Students]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Registry].[SPStudentsDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPStatesSelect]    Script Date: 08/16/2011 01:24↙
    :29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for retrieving values from SetUp.States Table

CREATE PROC [SetUp].[SPStatesSelect]
(
    @Code NVARCHAR(50)=NULL,
    @CountriesCode NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[States] WHERE ([Code] = @Code))
            BEGIN
            SELECT * FROM [SetUp].[States]
            WHERE
                (
                ([Code] = @Code) AND
                ([CountriesCode]  = @CountriesCode)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [SetUp].[States]
            WHERE
                (
                ([CountriesCode]  = @CountriesCode)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPStatesSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPStatesInsertUpdate]    Script Date: 08/16/2011↙
     01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for saving values into SetUp.States Table

CREATE PROC [SetUp].[SPStatesInsertUpdate]
(
    @Code NVARCHAR(50)=NULL,
    @Name NVARCHAR(256)=NULL,
    @CountriesCode NVARCHAR(50)=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[States]  WHERE (([Code] = @Code) AND (       ↙
    [CountriesCode] = @CountriesCode)) )
            BEGIN
                INSERT INTO [SetUp].[States]
                (
                    [Code],
                    [Name],
                    [CountriesCode]

                )
                VALUES
                (
                    @Code,
                    @Name,
                    @CountriesCode

                )
            END
        ELSE
            BEGIN
                UPDATE [SetUp].[States]
                SET
                    [Name]=@Name,
                    [CountriesCode]=@CountriesCode

                WHERE
                    ([Code] = @Code)
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPStatesInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPStatesDeletePermanently]    Script Date: 08/16↙
    /2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author             :   Ademola Adebo
--Reviewer           :   Godwin Mathias
--Date Created       :   07/12/2011
--Last Updated       :   07/12/2011
--Last Updated By    :   Ademola Adebo
--Description        :   Stored procedure for deleting values from SetUp.States Table

CREATE PROC [SetUp].[SPStatesDeletePermanently]
(
    @Code NVARCHAR(50)=NULL,
    @CountriesCode NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[States] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [SetUp].[States]
            WHERE
                (
                    ([Code] = @Code) AND
                    ([CountriesCode] = @CountriesCode)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPStatesDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPStaffSelect]    Script Date: 08/16/2011 01:24:↙
    29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for retrieving values from SetUp.Staff Table

CREATE PROC [SetUp].[SPStaffSelect]
(
    @Code NVARCHAR(50)=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Staff] WHERE ([Code] = @Code) AND (           ↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
            SELECT * FROM [SetUp].[Staff]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [SetUp].[Staff]
            WHERE
                (
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPStaffSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPStaffInsertUpdate]    Script Date: 08/16/2011 ↙
    01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for saving values into SetUp.Staff Table

CREATE PROC [SetUp].[SPStaffInsertUpdate]
(
    @Code NVARCHAR(50)=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @FacultyCode BIGINT=NULL,
    @DepartmentCode BIGINT=NULL,
    @StaffType NVARCHAR(50)=NULL,
    @ContractType NVARCHAR(50)=NULL,
    @ComDate DATETIME2(7)=NULL,
    @CessDate DATETIME2(7)=NULL,
    @StatusCode NVARCHAR(50)=NULL,
    @StatusReason NVARCHAR(500)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[Staff]  WHERE ([Code] = @Code) AND ( ↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
                INSERT INTO [SetUp].[Staff]
                (
                    [Code],
                    [AccountCode],
                    [UniversityCode],
                    [FacultyCode],
                    [DepartmentCode],
                    [StaffType],
                    [ContractType],
                    [ComDate],
                    [CessDate],
                    [StatusCode],
                    [StatusReason],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]

                )
                VALUES
                (
                    @Code,
                    @AccountCode,
                    @UniversityCode,
                    @FacultyCode,
                    @DepartmentCode,
                    @StaffType,
                    @ContractType,
                    @ComDate,
                    @CessDate,
                    @StatusCode,
                    @StatusReason,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [SetUp].[Staff]
                SET
                    [Code]=@Code,
                    [AccountCode]=@AccountCode,
                    [UniversityCode]=@UniversityCode,
                    [FacultyCode]=@FacultyCode,
                    [DepartmentCode]=@DepartmentCode,
                    [StaffType]=@StaffType,
                    [ContractType]=@ContractType,
                    [ComDate]=@ComDate,
                    [CessDate]=@CessDate,
                    [StatusCode]=@StatusCode,
                    [StatusReason]=@StatusReason,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPStaffInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPStaffDeletePermanently]    Script Date: 08/16/↙
    2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/12/2011
--Last Updated      :   07/12/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from SetUp.Staff Table

CREATE PROC [SetUp].[SPStaffDeletePermanently]
(
    @Code NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Staff] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [SetUp].[Staff]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPStaffDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPStaffDelete]    Script Date: 08/16/2011 01:24:↙
    29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/28/2011
--Last Updated      :   07/28/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from SetUp.Staff ↙
    Table

CREATE PROC [SetUp].[SPStaffDelete]
(
    @Code NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[Staff] WHERE (([Code] = @Code) AND (Deleted=↙
    @Deleted)))
            BEGIN
            UPDATE [SetUp].[Staff]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);
```

```
SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPStaffDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPReligionsSelect]    Script Date: 08/16/   ↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Personal.Religions    ↙
    Table

CREATE PROC [Personals].[SPReligionsSelect]
(
    @Code INT=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Religions] WHERE (([Code] = @Code) AND (    ↙
    [AccountCode]=@AccountCode) AND ([ScreenCode]=@ScreenCode) AND ([Deleted]=@Deleted)))
            BEGIN
            SELECT * FROM [Personals].[Religions]
            WHERE
                (
                ([Code] = @Code) AND
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Personals].[Religions]
            WHERE
                (
                ([AccountCode]=@AccountCode) AND
                ([ScreenCode]=@ScreenCode) AND
                ([Deleted]=@Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN
```

```sql
DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPReligionsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPReligionsInsertUpdate]    Script Date: 08/↙
    16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Personals.Religions Table

CREATE PROC [Personals].[SPReligionsInsertUpdate]
(
    @Code INT=NULL,
    @BioDataCode NVARCHAR(50)=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @ReligionCode NVARCHAR(50)=NULL,
    @Clergy NVARCHAR(150)=NULL,
    @PlaceOfWorship NVARCHAR(256)=NULL,
    @PlaceOfWorshipAddress NVARCHAR(256)=NULL,
    @ClergyPhoneNumber NVARCHAR(50)=NULL,
    @ClergyEmail NVARCHAR(50)=NULL,
    @PlaceOfWorshipPhoneNumber NVARCHAR(50)=NULL,
    @PlaceOfWorshipEmail NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50) = NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Personals].[Religions] WHERE (([Code] = @Code) AND (↙
    [AccountCode]=@AccountCode) AND ([Deleted]=@Deleted)))
            BEGIN
                INSERT INTO [Personals].[Religions]
                (
                    [BioDataCode],
                    [AccountCode],
                    [ReligionCode],
                    [Clergy],
                    [PlaceOfWorship],
                    [PlaceOfWorshipAddress],
                    [ClergyPhoneNumber],
                    [ClergyEmail],
                    [PlaceOfWorshipPhoneNumber],
                    [PlaceOfWorshipEmail],
                    [ScreenCode],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    @BioDataCode,
                    @AccountCode,
                    @ReligionCode,
                    @Clergy,
                    @PlaceOfWorship,
                    @PlaceOfWorshipAddress,
                    @ClergyPhoneNumber,
                    @ClergyEmail,
                    @PlaceOfWorshipPhoneNumber,
                    @PlaceOfWorshipEmail,
                    @ScreenCode,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [Personals].[Religions]
                SET
                    [BioDataCode]=@BioDataCode,
                    [AccountCode]=@AccountCode,
                    [ReligionCode]=@ReligionCode,
                    [Clergy]=@Clergy,
                    [PlaceOfWorship]=@PlaceOfWorship,
                    [PlaceOfWorshipAddress]=@PlaceOfWorshipAddress,
                    [ClergyPhoneNumber]=@ClergyPhoneNumber,
                    [ClergyEmail]=@ClergyEmail,
                    [PlaceOfWorshipPhoneNumber]=@PlaceOfWorshipPhoneNumber,
                    [PlaceOfWorshipEmail]=@PlaceOfWorshipEmail,
                    [ScreenCode]=@ScreenCode,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([AccountCode]=@AccountCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPReligionsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPReligionsDeletePermanently]    Script Date↙
    : 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Personal.Religions Table

CREATE PROC [Personals].[SPReligionsDeletePermanently]
(
    @Code INT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Religions] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Personals].[Religions]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPReligionsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Personals].[SPReligionsDelete]    Script Date: 08/16/   ↙
```

```
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Personal.   ↙
    Religions Table

CREATE PROC [Personals].[SPReligionsDelete]
(
    @Code INT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Personals].[Religions] WHERE (([Code] = @Code) AND      ↙
    (Deleted=@Deleted)))
            BEGIN
            UPDATE [Personals].[Religions]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Personals].[SPReligionsDelete] TO PUBLIC
GO
/****** Object:  Table [Academics].[Submissions]    Script Date: 08/16/2011 01:24:32 *****↙
    */
```

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Academics].[Submissions](
    [Code] [bigint] NOT NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [FacultyCode] [bigint] NULL,
    [DepartmentCode] [bigint] NULL,
    [CourseCode] [bigint] NULL,
    [ProgramCode] [bigint] NULL,
    [LevelCode] [int] NULL,
    [StudentCode] [bigint] NULL,
    [MatricNo] [nvarchar](50) NULL,
    [SessionCode] [bigint] NULL,
    [SemesterCode] [bigint] NULL,
    [Submitted] [bit] NULL,
    [ScreenCode] [nvarchar](50) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
    [Accepted] [bit] NULL,
    [AcceptedOn] [datetime2](7) NULL,
    [AcceptedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Submissions] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,          ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
EXEC sys.sp_addextendedproperty @name=N'MS_Description', @value=N'receive submission of  ↙
    forms for both department & faculty' , @level0type=N'SCHEMA',@level0name=N'Academics',↙
     @level1type=N'TABLE',@level1name=N'Submissions', @level2type=N'COLUMN',@level2name=N ↙
    'ScreenCode'
GO
EXEC sys.sp_addextendedproperty @name=N'MS_Description', @value=N'Note:
Use IF Not Exist(Select * from Academics where UniversityCode,FacultyCode,DepartmentCode, ↙
    CourseCode, ProgramCode, SemesterCode, LevelCode, ScreenCode, Submitted)' , @         ↙
    level0type=N'SCHEMA',@level0name=N'Academics', @level1type=N'TABLE',@level1name=N     ↙
    'Submissions'
GO
/****** Object:  View [dbo].[vw_aspnet_WebPartState_Paths]    Script Date: 08/16/2011 01: ↙
    24:34 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE VIEW [dbo].[vw_aspnet_WebPartState_Paths]
  AS SELECT [dbo].[aspnet_Paths].[ApplicationId], [dbo].[aspnet_Paths].[PathId], [dbo].   ↙
    [aspnet_Paths].[Path], [dbo].[aspnet_Paths].[LoweredPath]
  FROM [dbo].[aspnet_Paths]
GO
/****** Object:  View [dbo].[vw_aspnet_Users]    Script Date: 08/16/2011 01:24:34 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE VIEW [dbo].[vw_aspnet_Users]
  AS SELECT [dbo].[aspnet_Users].[ApplicationId], [dbo].[aspnet_Users].[UserId], [dbo].   ↙
    [aspnet_Users].[UserName], [dbo].[aspnet_Users].[LoweredUserName], [dbo].             ↙
    [aspnet_Users].[MobileAlias], [dbo].[aspnet_Users].[IsAnonymous], [dbo].[aspnet_Users]↙
    .[LastActivityDate]
  FROM [dbo].[aspnet_Users]
GO
/****** Object:  View [dbo].[vw_aspnet_Roles]    Script Date: 08/16/2011 01:24:34 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE VIEW [dbo].[vw_aspnet_Roles]
  AS SELECT [dbo].[aspnet_Roles].[ApplicationId], [dbo].[aspnet_Roles].[RoleId], [dbo].   ↙
    [aspnet_Roles].[RoleName], [dbo].[aspnet_Roles].[LoweredRoleName], [dbo].            ↙
    [aspnet_Roles].[Description]
  FROM [dbo].[aspnet_Roles]
GO
/****** Object:  Table [dbo].[aspnet_PersonalizationPerUser]    Script Date: 08/16/2011 01↙
    :24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[aspnet_PersonalizationPerUser](
    [Id] [uniqueidentifier] NOT NULL,
    [PathId] [uniqueidentifier] NULL,
    [UserId] [uniqueidentifier] NULL,
    [PageSettings] [image] NOT NULL,
    [LastUpdatedDate] [datetime] NOT NULL,
PRIMARY KEY NONCLUSTERED
(
    [Id] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,          ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
GO
/****** Object:  Table [dbo].[aspnet_Profile]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[aspnet_Profile](
    [UserId] [uniqueidentifier] NOT NULL,
    [PropertyNames] [ntext] NOT NULL,
    [PropertyValuesString] [ntext] NOT NULL,
    [PropertyValuesBinary] [image] NOT NULL,
    [LastUpdatedDate] [datetime] NOT NULL,
PRIMARY KEY CLUSTERED
(
    [UserId] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,          ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
GO
/****** Object:  Table [Assessment].[Attendance]    Script Date: 08/16/2011 01:24:32 *****↙
    */
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Assessment].[Attendance](
    [Code] [bigint] IDENTITY(1,1) NOT NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [FacultyCode] [bigint] NULL,
    [DepartmentCode] [bigint] NULL,
    [CourseCode] [bigint] NULL,
    [ProgramCode] [bigint] NULL,
    [LevelCode] [int] NULL,
    [StudentCode] [bigint] NULL,
    [MatricNo] [nvarchar](50) NULL,
    [SessionCode] [bigint] NULL,
    [SemesterCode] [bigint] NULL,
    [AttendanceStatus] [nvarchar](50) NULL,
    [StaffCode] [nvarchar](50) NULL,
    [SubCourseCode] [bigint] NULL,
    [AttendanceType] [nvarchar](50) NULL,
    [Remark] [nchar](10) NULL,
    [ScreenCode] [nvarchar](50) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Attendance] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,        ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  StoredProcedure [dbo].[aspnet_Users_CreateUser]    Script Date: 08/16/   ↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_Users_CreateUser]
    @ApplicationId    uniqueidentifier,
    @UserName         nvarchar(256),
    @IsUserAnonymous  bit,
    @LastActivityDate DATETIME,
    @UserId           uniqueidentifier OUTPUT
AS
BEGIN
    IF( @UserId IS NULL )
        SELECT @UserId = NEWID()
    ELSE
    BEGIN
        IF( EXISTS( SELECT UserId FROM dbo.aspnet_Users
                    WHERE @UserId = UserId ) )
            RETURN -1
    END

    INSERT dbo.aspnet_Users (ApplicationId, UserId, UserName, LoweredUserName, IsAnonymous↙
    , LastActivityDate)
    VALUES (@ApplicationId, @UserId, @UserName, LOWER(@UserName), @IsUserAnonymous, @     ↙
    LastActivityDate)

    RETURN 0
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_Roles_RoleExists]    Script Date: 08/16/   ↙
    2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_Roles_RoleExists]
    @ApplicationName  nvarchar(256),
    @RoleName         nvarchar(256)
AS
BEGIN
    DECLARE @ApplicationId uniqueidentifier
    SELECT  @ApplicationId = NULL
    SELECT  @ApplicationId = ApplicationId FROM aspnet_Applications WHERE LOWER(@         ↙
    ApplicationName) = LoweredApplicationName
    IF (@ApplicationId IS NULL)
        RETURN(0)
    IF (EXISTS (SELECT RoleName FROM dbo.aspnet_Roles WHERE LOWER(@RoleName) =           ↙
    LoweredRoleName AND ApplicationId = @ApplicationId ))
        RETURN(1)
    ELSE
        RETURN(0)
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_Roles_GetAllRoles]    Script Date: 08/16/  ↙
    2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_Roles_GetAllRoles] (
    @ApplicationName           nvarchar(256))
AS
BEGIN
    DECLARE @ApplicationId uniqueidentifier
    SELECT  @ApplicationId = NULL
    SELECT  @ApplicationId = ApplicationId FROM aspnet_Applications WHERE LOWER(@         ↙
    ApplicationName) = LoweredApplicationName
    IF (@ApplicationId IS NULL)
        RETURN
    SELECT RoleName
    FROM   dbo.aspnet_Roles WHERE ApplicationId = @ApplicationId
    ORDER BY RoleName
END
GO
/****** Object:  Table [dbo].[aspnet_UsersInRoles]    Script Date: 08/16/2011 01:24:32 ***↙
    ***/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[aspnet_UsersInRoles](
    [UserId] [uniqueidentifier] NOT NULL,
    [RoleId] [uniqueidentifier] NOT NULL,
    [SchoolCode] [nvarchar](50) NULL,
    [BranchCode] [nvarchar](50) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK__aspnet_U__AF2760AD2F10CBD2] PRIMARY KEY CLUSTERED
(
    [UserId] ASC,
    [RoleId] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,          ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[aspnet_Membership]    Script Date: 08/16/2011 01:24:32 *****↙
    */
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[aspnet_Membership](
    [ApplicationId] [uniqueidentifier] NOT NULL,
    [UserId] [uniqueidentifier] NOT NULL,
    [Password] [nvarchar](128) NOT NULL,
    [PasswordFormat] [int] NOT NULL,
    [PasswordSalt] [nvarchar](128) NOT NULL,
    [MobilePIN] [nvarchar](16) NULL,
    [Email] [nvarchar](256) NULL,
    [LoweredEmail] [nvarchar](256) NULL,
    [PasswordQuestion] [nvarchar](256) NULL,
    [PasswordAnswer] [nvarchar](128) NULL,
    [IsApproved] [bit] NOT NULL,
    [IsLockedOut] [bit] NOT NULL,
    [CreateDate] [datetime] NOT NULL,
    [LastLoginDate] [datetime] NOT NULL,
    [LastPasswordChangedDate] [datetime] NOT NULL,
    [LastLockoutDate] [datetime] NOT NULL,
    [FailedPasswordAttemptCount] [int] NOT NULL,
    [FailedPasswordAttemptWindowStart] [datetime] NOT NULL,
    [FailedPasswordAnswerAttemptCount] [int] NOT NULL,
    [FailedPasswordAnswerAttemptWindowStart] [datetime] NOT NULL,
    [Comment] [ntext] NULL,
PRIMARY KEY NONCLUSTERED
(
    [UserId] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,       ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
GO
INSERT [dbo].[aspnet_Membership] ([ApplicationId], [UserId], [Password], [PasswordFormat],↙
     [PasswordSalt], [MobilePIN], [Email], [LoweredEmail], [PasswordQuestion],          ↙
    [PasswordAnswer], [IsApproved], [IsLockedOut], [CreateDate], [LastLoginDate],       ↙
    [LastPasswordChangedDate], [LastLockoutDate], [FailedPasswordAttemptCount],         ↙
    [FailedPasswordAttemptWindowStart], [FailedPasswordAnswerAttemptCount],             ↙
    [FailedPasswordAnswerAttemptWindowStart], [Comment]) VALUES (N'3436a128-b3cb-4d6a-a7f2↙
    -782b22a6d04d', N'b6a16061-1f52-40b4-a680-be68f8df7d26', N'zaUNKSgIrGJXpiwFKuaxckQnsOc↙
    =', 1, N'CLfvvk4RIsZopstApTnGOA==', NULL, N'chiriksmat@hotmail.com', N'chiriksmat@    ↙
    hotmail.com', N'name', N'TgQM6Mr5tdMhuOoJ18+CMNtTZps=', 0, 0, CAST(0x00009EEC016CD544 ↙
    AS DateTime), CAST(0x00009EEC016CD544 AS DateTime), CAST(0x00009EEC016CD544 AS      ↙
    DateTime), CAST(0xFFFF2FB300000000 AS DateTime), 0, CAST(0xFFFF2FB300000000 AS      ↙
    DateTime), 0, CAST(0xFFFF2FB300000000 AS DateTime), NULL)
/****** Object:  StoredProcedure [dbo].[aspnet_Paths_CreatePath]    Script Date: 08/16/   ↙
    2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_Paths_CreatePath]
    @ApplicationId UNIQUEIDENTIFIER,
    @Path           NVARCHAR(256),
    @PathId         UNIQUEIDENTIFIER OUTPUT
AS
BEGIN
    BEGIN TRANSACTION
    IF (NOT EXISTS(SELECT * FROM dbo.aspnet_Paths WHERE LoweredPath = LOWER(@Path) AND   ↙
    ApplicationId = @ApplicationId))
    BEGIN
        INSERT dbo.aspnet_Paths (ApplicationId, Path, LoweredPath) VALUES (@ApplicationId,↙
     @Path, LOWER(@Path))
    END
    COMMIT TRANSACTION
    SELECT @PathId = PathId FROM dbo.aspnet_Paths WHERE LOWER(@Path) = LoweredPath AND    ↙
    ApplicationId = @ApplicationId
END
GO
/****** Object:  Table [dbo].[aspnet_PersonalizationAllUsers]    Script Date: 08/16/2011  ↙
    01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[aspnet_PersonalizationAllUsers](
    [PathId] [uniqueidentifier] NOT NULL,
    [PageSettings] [image] NOT NULL,
    [LastUpdatedDate] [datetime] NOT NULL,
PRIMARY KEY CLUSTERED 
(
    [PathId] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,      ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
GO
/****** Object:  Table [SetUp].[LGAs]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [SetUp].[LGAs](
    [Code] [nvarchar](50) NOT NULL,
    [StatesCode] [nvarchar](50) NOT NULL,
    [CountriesCode] [nvarchar](50) NOT NULL,
    [LgName] [nvarchar](256) NOT NULL,
 CONSTRAINT [PK_LGAs] PRIMARY KEY CLUSTERED 
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,      ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
INSERT [SetUp].[LGAs] ([Code], [StatesCode], [CountriesCode], [LgName]) VALUES (N'AFJ', N ↙
    'OY', N'NGN', N'Afijio')
INSERT [SetUp].[LGAs] ([Code], [StatesCode], [CountriesCode], [LgName]) VALUES (N'APP', N ↙
    'LG', N'NGN', N'Apapa')
INSERT [SetUp].[LGAs] ([Code], [StatesCode], [CountriesCode], [LgName]) VALUES (N'ATB', N ↙
    'OY', N'NGN', N'Atiba')
INSERT [SetUp].[LGAs] ([Code], [StatesCode], [CountriesCode], [LgName]) VALUES (N'IKM', N ↙
    'CR', N'NGN', N'Ikom')
INSERT [SetUp].[LGAs] ([Code], [StatesCode], [CountriesCode], [LgName]) VALUES (N'MB', N  ↙
    'AD', N'NGN', N'Mubi')
INSERT [SetUp].[LGAs] ([Code], [StatesCode], [CountriesCode], [LgName]) VALUES (N'MCH', N ↙
    'AD', N'NGN', N'Michika')
INSERT [SetUp].[LGAs] ([Code], [StatesCode], [CountriesCode], [LgName]) VALUES (N'OYW', N ↙
    'OY', N'NGN', N'Oyo-West')
INSERT [SetUp].[LGAs] ([Code], [StatesCode], [CountriesCode], [LgName]) VALUES (N'WWW', N ↙
    'OD', N'NGN', N'Owo')
INSERT [SetUp].[LGAs] ([Code], [StatesCode], [CountriesCode], [LgName]) VALUES (N'YL ', N ↙
    'AD', N'NGN', N'Yola')
/****** Object:  UserDefinedFunction [dbo].[GetProgramName]    Script Date: 08/16/2011 01:↙
    24:33 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE FUNCTION [dbo].[GetProgramName](@Code NVARCHAR(50)) RETURNS NVARCHAR(256)
AS
BEGIN
    DECLARE @ProgramName NVARCHAR(256)
    SET @ProgramName = (SELECT [Name] FROM [SetUp].[Descriptions] WHERE ([Code]=@Code))
    RETURN @ProgramName
END
GO
/****** Object:  Table [Academics].[FinalSignatories]    Script Date: 08/16/2011 01:24:32 ↙
    ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Academics].[FinalSignatories](
    [Code] [bigint] NOT NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [FacultyCode] [bigint] NULL,
    [DepartmentCode] [bigint] NULL,
    [CourseCode] [bigint] NULL,
    [ProgramCode] [bigint] NULL,
    [LevelCode] [int] NULL,
    [StudentCode] [bigint] NULL,
    [MatricNo] [nvarchar](50) NULL,
    [SessionCode] [bigint] NULL,
    [SemesterCode] [bigint] NULL,
    [Signed] [bit] NULL,
    [SignedOn] [datetime2](7) NULL,
    [SignedBy] [nvarchar](50) NULL,
    [DesignationCode] [nvarchar](50) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_FinalSignatories] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF, ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
EXEC sys.sp_addextendedproperty @name=N'MS_Description', @value=N'store staffcode here' , ↙
    @level0type=N'SCHEMA',@level0name=N'Academics', @level1type=N'TABLE',@level1name=N ↙
    'FinalSignatories', @level2type=N'COLUMN',@level2name=N'SignedBy'
GO
/****** Object:  StoredProcedure [dbo].[aspnet_Roles_CreateRole]    Script Date: 08/16/ ↙
    2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_Roles_CreateRole]
    @ApplicationName  nvarchar(256),
    @RoleName         nvarchar(256)
AS
BEGIN
    DECLARE @ApplicationId uniqueidentifier
    SELECT  @ApplicationId = NULL

    DECLARE @ErrorCode     int
    SET @ErrorCode = 0

    DECLARE @TranStarted   bit
    SET @TranStarted = 0

    IF( @@TRANCOUNT = 0 )
    BEGIN
        BEGIN TRANSACTION
        SET @TranStarted = 1
    END
    ELSE
        SET @TranStarted = 0

    EXEC dbo.aspnet_Applications_CreateApplication @ApplicationName, @ApplicationId OUTPUT

    IF( @@ERROR <> 0 )
    BEGIN
        SET @ErrorCode = -1
        GOTO Cleanup
    END
    IF (EXISTS(SELECT RoleId FROM dbo.aspnet_Roles WHERE LoweredRoleName = LOWER(@        ↙
    RoleName) AND ApplicationId = @ApplicationId))
    BEGIN
        SET @ErrorCode = 1
        GOTO Cleanup
    END

    INSERT INTO dbo.aspnet_Roles
                (ApplicationId, RoleName, LoweredRoleName)
         VALUES (@ApplicationId, @RoleName, LOWER(@RoleName))

    IF( @@ERROR <> 0 )
    BEGIN
        SET @ErrorCode = -1
        GOTO Cleanup
    END

    IF( @TranStarted = 1 )
    BEGIN
        SET @TranStarted = 0
        COMMIT TRANSACTION
    END

    RETURN(0)

Cleanup:

    IF( @TranStarted = 1 )
    BEGIN
        SET @TranStarted = 0
        ROLLBACK TRANSACTION
    END

    RETURN @ErrorCode

END
GO
/****** Object:  Table [SetUp].[Designations]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [SetUp].[Designations](
    [Code] [bigint] IDENTITY(1,1) NOT NULL,
    [StaffCode] [nvarchar](50) NULL,
    [ComDate] [datetime2](7) NULL,
    [CessDate] [datetime2](7) NULL,
    [DesignationCode] [nvarchar](50) NULL,
    [TypeCode] [nvarchar](50) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Designations] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,         ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[aspnet_UsersInBranches]    Script Date: 08/16/2011 01:24:32 ↙
    ******/
SET ANSI_NULLS ON
```

```
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[aspnet_UsersInBranches](
    [Code] [uniqueidentifier] NOT NULL,
    [UserId] [uniqueidentifier] NULL,
    [SchoolCode] [nvarchar](50) NULL,
    [BranchCode] [nvarchar](50) NULL,
    [DefaultBranch] [bit] NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_aspnet_UsersInBranches_1] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,        ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Assessment].[Records]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Assessment].[Records](
    [Code] [bigint] IDENTITY(1,1) NOT NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [FacultyCode] [bigint] NULL,
    [DepartmentCode] [bigint] NULL,
    [CourseCode] [bigint] NULL,
    [ProgramCode] [bigint] NULL,
    [SessionCode] [bigint] NULL,
    [SemesterCode] [bigint] NULL,
    [LevelCode] [int] NULL,
    [AssessmentTypeCode] [nvarchar](50) NULL,
    [SubCourseCode] [nvarchar](50) NULL,
    [StaffCode] [nvarchar](50) NULL,
    [Score] [decimal](18, 2) NULL,
    [GradeDescription] [nvarchar](50) NULL,
    [GP] [decimal](18, 0) NULL,
    [Position] [bigint] NULL,
    [StudentCode] [bigint] NULL,
    [MatricNo] [nvarchar](50) NULL,
    [Remark] [nvarchar](50) NULL,
    [ScreenCode] [nvarchar](50) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Records] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,        ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  UserDefinedFunction [Academics].[GetSessionStatusDescription]    Script  ↙
    Date: 08/16/2011 01:24:33 ******/
SET ANSI_NULLS ON
GO
```

```
SET QUOTED_IDENTIFIER ON
GO
CREATE FUNCTION [Academics].[GetSessionStatusDescription](@Code NVARCHAR(50)) RETURNS     ↙
    NVARCHAR(50)
AS
BEGIN
        DECLARE @Name NVARCHAR(50)=NULL
        SET @Name=(SELECT [Name] FROM [SetUp].[Descriptions] WHERE ([Code]=@Code))
        RETURN ISNULL(@Name,'N/A')
END
GO
/****** Object:  StoredProcedure [Finance].[SPAccountTypesSelect]    Script Date: 08/16/   ↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author             :   Ademola Adebo
--Reviewer           :   Godwin Mathias
--Date Created       :   07/18/2011
--Last Updated       :   07/18/2011
--Last Updated By    :   Ademola Adebo
--Description        :   Stored procedure for retrieving values from Finance.AccountTypes   ↙
    Table

CREATE PROC [Finance].[SPAccountTypesSelect]
(
    @Code DECIMAL(18,0)=NULL,
    @UniversityCode NVARCHAR(50)=NULL
    )
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance].[AccountTypes] WHERE ([Code] = @Code) AND (     ↙
    [UniversityCode]=@UniversityCode))
            BEGIN
            SELECT * FROM [Finance].[AccountTypes]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Finance] .[AccountTypes]
            WHERE
                (
                ([UniversityCode]=@UniversityCode)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPAccountTypesSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPAccountTypesInsertUpdate]    Script Date: 08↙
    /16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Finance.AccountTypes Table

CREATE PROC [Finance].[SPAccountTypesInsertUpdate]
(
    @Code DECIMAL(18,0)=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Description NVARCHAR(256)=NULL,
    @DescriptionCode NVARCHAR(50)=NULL,
    @AccountSubCode NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Finance].[AccountTypes] WHERE ([Code] = @Code) AND (↙
    [UniversityCode]=@UniversityCode))
            BEGIN
                INSERT INTO [Finance].[AccountTypes]
                (
                    [UniversityCode],
                    [Description],
                    [DescriptionCode] ,
                    [AccountSubCode]

                )
                VALUES
                (
                    @UniversityCode,
                    @Description,
                    @DescriptionCode ,
                    @AccountSubCode
                )
            END
        ELSE
            BEGIN
                UPDATE [Finance].[AccountTypes]
                SET
                    [UniversityCode]=@UniversityCode,
                    [Description]=@Description,
                    [DescriptionCode] =@DescriptionCode ,
                    [AccountSubCode] =@AccountSubCode
                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
```

```
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPAccountTypesInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPAccountTypesDeletePermanently]    Script         ↙
    Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Finance.AccountTypes    ↙
    Table

CREATE PROC [Finance].[SPAccountTypesDeletePermanently]
(
    @Code DECIMAL(18,0)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance] .[AccountTypes]  WHERE ([Code] = @Code))
            BEGIN
            DELETE FROM [Finance] .[AccountTypes]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
```

```sql
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPAccountTypesDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPAccountSubSelect]    Script Date: 08/16/2011↙
     01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Finance.AccountSub    ↙
    Table

CREATE PROC [Finance].[SPAccountSubSelect]
(
    @Code INT=NULL,
    @UniversityCode NVARCHAR(50)=NULL
    )
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance].[AccountSub] WHERE ([Code] = @Code) AND (      ↙
    [UniversityCode]=@UniversityCode))
            BEGIN
            SELECT * FROM [Finance].[AccountSub]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Finance] .[AccountSub]
            WHERE
                (
                ([UniversityCode]=@UniversityCode)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPAccountSubSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPAccountSubInsertUpdate]    Script Date: 08/ ↙
    16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Finance.AccountSub Table

CREATE PROC [Finance].[SPAccountSubInsertUpdate]
(
    @Code INT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Description NVARCHAR(256)=NULL,
    @DescriptionCode NVARCHAR(50)=NULL,
    @ParameterCode NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Finance].[AccountSub] WHERE ([Code] = @Code) AND (  ↙
    [UniversityCode]=@UniversityCode))
            BEGIN
                INSERT INTO [Finance].[AccountSub]
                (
                    [UniversityCode],
                    [Description],
                    [DescriptionCode] ,
                    [ParameterCode]

                )
                VALUES
                (
                    @UniversityCode,
                    @Description,
                    @DescriptionCode ,
                    @ParameterCode
                )
            END
        ELSE
            BEGIN
                UPDATE [Finance].[AccountSub]
                SET
                    [UniversityCode]=@UniversityCode,
                    [Description]=@Description,
                    [DescriptionCode] =@DescriptionCode ,
                    [ParameterCode]=@ParameterCode
                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPAccountSubInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPAccountSubDeletePermanently]    Script Date:↙
     08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Finance.AccountSub Table

CREATE PROC [Finance].[SPAccountSubDeletePermanently]
(
    @Code INT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance] .[AccountSub]  WHERE ([Code] = @Code))
            BEGIN
            DELETE FROM [Finance] .[AccountSub]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPAccountSubDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPAccountsSelect]    Script Date: 08/16/2011  ↙
    01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/18/2011
--Last Updated        :   07/18/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for retrieving values from Finance.Accounts Table

CREATE PROC [Finance].[SPAccountsSelect]
(
    @Code DECIMAL(18,0)=NULL,
    @UniversityCode NVARCHAR(50)=NULL
    )
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance].[Accounts] WHERE ([Code] = @Code) AND (        ↙
    [UniversityCode]=@UniversityCode))
            BEGIN
            SELECT * FROM [Finance].[Accounts]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Finance] .[Accounts]
            WHERE
                (
                ([UniversityCode]=@UniversityCode)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPAccountsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPAccountsInsertUpdate]    Script Date: 08/16/↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Finance.Accounts Table

CREATE PROC [Finance].[SPAccountsInsertUpdate]
(
    @Code DECIMAL(18,0)=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Description NVARCHAR(256)=NULL,
    @AccountTypeCode DECIMAL(18,0)=NULL,
    @AccountSubCode INT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Finance].[Accounts] WHERE ([Code] = @Code) AND (     ↙
    [UniversityCode]=@UniversityCode))
            BEGIN
                INSERT INTO [Finance].[Accounts]
                (
                    [UniversityCode],
                    [Description],
                    [AccountTypeCode],
                    [AccountSubCode]

                )
                VALUES
                (
                    @UniversityCode,
                    @Description,
                    @AccountTypeCode,
                    @AccountSubCode
                )
            END
        ELSE
            BEGIN
                UPDATE [Finance].[Accounts]
                SET
                    [UniversityCode]=@UniversityCode,
                    [Description]=@Description,
                    [AccountTypeCode]=@AccountTypeCode,
                    [AccountSubCode]=@AccountSubCode
                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPAccountsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPAccountsDeletePermanently]    Script Date:  ↙
    08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Finance.Accounts Table

CREATE PROC [Finance].[SPAccountsDeletePermanently]
(
    @Code DECIMAL(18,0)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance] .[Accounts]  WHERE ([Code] = @Code))
            BEGIN
            DELETE FROM [Finance] .[Accounts]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN
```

```
DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPAccountsDeletePermanently] TO PUBLIC
GO
/****** Object:  Table [Academics].[Registrations]    Script Date: 08/16/2011 01:24:32 ***↙
    ***/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Academics].[Registrations](
    [Code] [bigint] IDENTITY(1,1) NOT NULL,
    [StudentCode] [bigint] NULL,
    [MatricNo] [nvarchar](50) NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [FacultyCode] [bigint] NULL,
    [DepartmentCode] [bigint] NULL,
    [CourseCode] [bigint] NULL,
    [ProgramCode] [bigint] NULL,
    [LevelCode] [int] NULL,
    [TypeCode] [nvarchar](50) NULL,
    [SessionCode] [bigint] NULL,
    [SemesterCode] [bigint] NULL,
    [SubCourseCode] [bigint] NULL,
    [Status] [nvarchar](50) NULL,
    [Signed] [bit] NULL,
    [SignedOn] [datetime2](7) NULL,
    [SignedBy] [nvarchar](50) NULL,
    [SelfSigned] [bit] NULL,
    [SelfSignedOn] [datetime2](7) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
    [TicketCode] [numeric](18, 0) NULL,
    [PinCode] [numeric](18, 0) NULL,
 CONSTRAINT [PK_Registrations] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,         ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
EXEC sys.sp_addextendedproperty @name=N'MS_Description', @value=N'Course registration type↙
     e.g. Fresh Registration| Registration for repeat | Elective | Required | Major' , @  ↙
    level0type=N'SCHEMA',@level0name=N'Academics', @level1type=N'TABLE',@level1name=N    ↙
    'Registrations', @level2type=N'COLUMN',@level2name=N'TypeCode'
GO
```

```
EXEC sys.sp_addextendedproperty @name=N'MS_Description', @value=N'Specify the lecturer ↙
    staffCode' , @level0type=N'SCHEMA',@level0name=N'Academics', @level1type=N'TABLE',@ ↙
    level1name=N'Registrations', @level2type=N'COLUMN',@level2name=N'SignedBy'
GO
/****** Object:  Table [SetUp].[Screens]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [SetUp].[Screens](
    [Code] [nvarchar](50) NOT NULL,
    [ModuleCode] [bigint] NULL,
    [UniversityCode] [nvarchar](50) NOT NULL,
    [ScreenDescription] [nvarchar](256) NULL,
    [Url] [nvarchar](256) NULL,
    [Notes] [nvarchar](500) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Screens] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,       ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'ACAD001', 1, N'ACU', N'Final Signatories', N'http:↙
    //127.0.0.1/UniversityPortalWeb/Academics/FinalSignatories', NULL, CAST               ↙
    (0x07BCEECDAC9B8C340B AS DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'ACAD002', 1, N'ACU', N'Registrations', N'http://  ↙
    127.0.0.1/UniversityPortalWeb/Academics/Registrations', NULL, CAST                    ↙
    (0x07950148FC2C8D340B AS DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'ACAD003', 1, N'ACU', N'Semesters', N'http://127.0.↙
    0.1/UniversityPortalWeb/Academics/Semesters', NULL, CAST(0x07C38804312D8D340B AS      ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'ACAD004', 1, N'ACU', N'Sessions', N'http://127.0.0↙
    .1/UniversityPortalWeb/Academics/Sessions', NULL, CAST(0x078BCA28582D8D340B AS        ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'ACAD005', 1, N'ACU', N'Signing Hierarcy', N'http:/↙
    /127.0.0.1/UniversityPortalWeb/Academics/SigningHierarchy', NULL, CAST                ↙
    (0x072D4ACF7B2D8D340B AS DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'ACAD006', 1, N'ACU', N'Submissions', N'http://127.↙
    0.0.1/UniversityPortalWeb/Academics/Submissions', NULL, CAST(0x0798C909A52D8D340B AS  ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'ASST001', 4, N'ACU', N'Attendance', N'http://127.0↙
    .0.1/UniversityPortalWeb/Assessment/Attendance', NULL, CAST(0x071A9B21E02D8D340B AS   ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'ASST002', 4, N'ACU', N'Definition', N'http://127.0↙
    .0.1/UniversityPortalWeb/Assessment/Definition', NULL, CAST(0x07DC836B112E8D340B AS   ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'ASST003', 4, N'ACU', N'Records', N'http://127.0.0.↙
    1/UniversityPortalWeb/Assessment/Records', NULL, CAST(0x07FE6BEE262E8D340B AS         ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'FINC001', 6, N'ACU', N'Accounts', N'http://127.0.0↙
    .1/UniversityPortalWeb/Finance/Accounts', NULL, CAST(0x07EC457A4B2E8D340B AS          ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'FINC002', 6, N'ACU', N'Account Sub.', N'http://127↙
    .0.0.1/UniversityPortalWeb/Finance/AccountSub', NULL, CAST(0x07062470722E8D340B AS    ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'FINC003', 6, N'ACU', N'Account Types', N'http://  ↙
    127.0.0.1/UniversityPortalWeb/Finance/AccountTypes', NULL, CAST(0x070C1BA38A2E8D340B  ↙
    AS DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'FINC004', 6, N'ACU', N'Fee Allotment', N'http://  ↙
    127.0.0.1/UniversityPortalWeb/Finance/FeeAllotment', NULL, CAST(0x070DB8EBAF2E8D340B  ↙
    AS DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'FINC005', 6, N'ACU', N'Fee Definition', N'http:// ↙
    127.0.0.1/UniversityPortalWeb/Finance/FeeDefinition', NULL, CAST(0x07071C60CA2E8D340B ↙
    AS DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'FINC006', 6, N'ACU', N'Fees', N'http://127.0.0.1/ ↙
    UniversityPortalWeb/Finance/Fees', NULL, CAST(0x0706A790DD2E8D340B AS DateTime2), N   ↙
    'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'FINC007', 6, N'ACU', N'Fees Detail', N'http://127.↙
    0.0.1/UniversityPortalWeb/Finance/FeesDetail', NULL, CAST(0x07BEB8FFF22E8D340B AS     ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'FINC008', 6, N'ACU', N'Payment Plan Detail', N    ↙
    'http://127.0.0.1/UniversityPortalWeb/Finance/PaymentPlanDetails', NULL, CAST         ↙
    (0x079747952E2F8D340B AS DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'FINC009', 6, N'ACU', N'Payment Plan', N'http://127↙
    .0.0.1/UniversityPortalWeb/Finance/PaymentPlan', NULL, CAST(0x075991C8512F8D340B AS   ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'FINC010', 6, N'ACU', N'Refunds', N'http://127.0.0.↙
    1/UniversityPortalWeb/Finance/Refund', NULL, CAST(0x072E31336C2F8D340B AS DateTime2), ↙
    N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'FINC011', 6, N'ACU', N'Transactions', N'http://127↙
    .0.0.1/UniversityPortalWeb/Finance/Transactions', NULL, CAST(0x076098FA862F8D340B AS  ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'FINC012', 6, N'ACU', N'Transactions Detail', N    ↙
    'http://127.0.0.1/UniversityPortalWeb/Finance/TransactionsDetail', NULL, CAST         ↙
    (0x07413BB1A42F8D340B AS DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],     ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],       ↙
    [DeletedOn], [DeletedBy]) VALUES (N'FINC013', 6, N'ACU', N'Vouchers', N'http://127.0.0↙
    .1/UniversityPortalWeb/Finance/Vouchers', NULL, CAST(0x07D91892D42F8D340B AS           ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],     ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],       ↙
    [DeletedOn], [DeletedBy]) VALUES (N'PERS001', 7, N'ACU', N'Addresses', N'http://127.0.↙
    0.1/UniversityPortalWeb/Personals/Addresses', NULL, CAST(0x075B31DFFA2F8D340B AS       ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],     ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],       ↙
    [DeletedOn], [DeletedBy]) VALUES (N'PERS002', 7, N'ACU', N'Area of Expertise', N'http:↙
    //127.0.0.1/UniversityPortalWeb/Personals/AreasOfExpertises', NULL, CAST               ↙
    (0x07517E9639308D340B AS DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],     ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],       ↙
    [DeletedOn], [DeletedBy]) VALUES (N'PERS003', 7, N'ACU', N'BioDatas', N'http://127.0.0↙
    .1/UniversityPortalWeb/Personals/BioDatas', NULL, CAST(0x076439E860308D340B AS         ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],     ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],       ↙
    [DeletedOn], [DeletedBy]) VALUES (N'PERS004', 7, N'ACU', N'Certificates', N'http://127↙
    .0.0.1/UniversityPortalWeb/Personals/Certificates', NULL, CAST(0x07BFEA7FAC308D340B AS↙
     DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],     ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],       ↙
    [DeletedOn], [DeletedBy]) VALUES (N'PERS005', 7, N'ACU', N'Choices', N'http://127.0.0.↙
    1/UniversityPortalWeb/Personals/Choices', NULL, CAST(0x07F1DC46BF308D340B AS           ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],     ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],       ↙
    [DeletedOn], [DeletedBy]) VALUES (N'PERS006', 7, N'ACU', N'Companies', N'http://127.0.↙
    0.1/UniversityPortalWeb/Personals/Companies', NULL, CAST(0x077D21DDD1308D340B AS       ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],     ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],       ↙
    [DeletedOn], [DeletedBy]) VALUES (N'PERS007', 7, N'ACU', N'Computer Skill', N'http:// ↙
    127.0.0.1/UniversityPortalWeb/Personals/ComputerSkills', NULL, CAST                    ↙
    (0x07F6F86EEC308D340B AS DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],     ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],       ↙
    [DeletedOn], [DeletedBy]) VALUES (N'PERS008', 7, N'ACU', N'Declarations', N'http://127↙
    .0.0.1/UniversityPortalWeb/Personals/Declarations', NULL, CAST(0x0757B4F608318D340B AS↙
     DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],     ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],       ↙
    [DeletedOn], [DeletedBy]) VALUES (N'PERS009', 7, N'ACU', N'E-mails', N'http://127.0.0.↙
    1/UniversityPortalWeb/Personals/Emails', NULL, CAST(0x073F1C9126318D340B AS DateTime2)↙
    , N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],     ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],       ↙
    [DeletedOn], [DeletedBy]) VALUES (N'PERS010', 7, N'ACU', N'Emergencies', N'http://127.↙
    0.0.1/UniversityPortalWeb/Personals/Emergencies', NULL, CAST(0x07578F8D40318D340B AS   ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],     ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],       ↙
    [DeletedOn], [DeletedBy]) VALUES (N'PERS011', 7, N'ACU', N'Employment History', N'http↙
    ://127.0.0.1/UniversityPortalWeb/Personals/EmploymentHistories', NULL, CAST            ↙
    (0x0756B17A59318D340B AS DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],     ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],       ↙
    [DeletedOn], [DeletedBy]) VALUES (N'PERS012', 7, N'ACU', N'Guarantor', N'http://127.0.↙
    0.1/UniversityPortalWeb/Personals/Guarantor', NULL, CAST(0x074E9D5977318D340B AS       ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],     ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],       ↙
    [DeletedOn], [DeletedBy]) VALUES (N'PERS013', 7, N'ACU', N'Guardian', N'http://127.0.0↙
    .1/UniversityPortalWeb/Personals/Guardians', NULL, CAST(0x07F31464A6318D340B AS        ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],     ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],       ↙
    [DeletedOn], [DeletedBy]) VALUES (N'PERS014', 7, N'ACU', N'Hobby', N'http://127.0.0.1/↙
    UniversityPortalWeb/Personals/Hobbies', NULL, CAST(0x0725ADACC8318D340B AS DateTime2),↙
     N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],     ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],       ↙
    [DeletedOn], [DeletedBy]) VALUES (N'PERS015', 7, N'ACU', N'JAMB', N'http://127.0.0.1/ ↙
    UniversityPortalWeb/Personals/JAMBs', NULL, CAST(0x0778ECD6E5318D340B AS DateTime2), N↙
    'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],     ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],       ↙
    [DeletedOn], [DeletedBy]) VALUES (N'PERS016', 7, N'ACU', N'Language Skill', N'http:// ↙
    127.0.0.1/UniversityPortalWeb/Personals/LanguageSkills', NULL, CAST                    ↙
    (0x07657236FE318D340B AS DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],     ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],       ↙
    [DeletedOn], [DeletedBy]) VALUES (N'PERS017', 7, N'ACU', N'Medical Condition', N'http:↙
    //127.0.0.1/UniversityPortalWeb/Personals/MedicalConditions', NULL, CAST               ↙
    (0x0725A42021328D340B AS DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],     ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],       ↙
    [DeletedOn], [DeletedBy]) VALUES (N'PERS018', 7, N'ACU', N'Next of Kin', N'http://127.↙
    0.0.1/UniversityPortalWeb/Personals/NextOfKins', NULL, CAST(0x07C61F4349328D340B AS    ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],     ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],       ↙
    [DeletedOn], [DeletedBy]) VALUES (N'PERS019', 7, N'ACU', N'O''Level', N'http://127.0.0↙
    .1/UniversityPortalWeb/Personals/OLevels', NULL, CAST(0x07A9C08E60328D340B AS          ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],     ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],       ↙
    [DeletedOn], [DeletedBy]) VALUES (N'PERS020', 7, N'ACU', N'Others', N'http://127.0.0.1↙
    /UniversityPortalWeb/Personals/Others', NULL, CAST(0x072844AB7D328D340B AS DateTime2),↙
     N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],     ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],       ↙
    [DeletedOn], [DeletedBy]) VALUES (N'PERS021', 7, N'ACU', N'Phones', N'http://127.0.0.1↙
    /UniversityPortalWeb/Personals/Phones', NULL, CAST(0x07689A6599328D340B AS DateTime2),↙
     N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],     ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],       ↙
    [DeletedOn], [DeletedBy]) VALUES (N'PERS022', 7, N'ACU', N'Photos', N'http://127.0.0.1↙
    /UniversityPortalWeb/Personals/Photos', NULL, CAST(0x076480C1B7328D340B AS DateTime2),↙
     N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],     ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],       ↙
    [DeletedOn], [DeletedBy]) VALUES (N'PERS023', 7, N'ACU', N'Qualifications', N'http:// ↙
    127.0.0.1/UniversityPortalWeb/Personals/Qualifications', NULL, CAST                    ↙
    (0x07130149D8328D340B AS DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],     ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],       ↙
    [DeletedOn], [DeletedBy]) VALUES (N'PERS024', 7, N'ACU', N'Referees', N'http://127.0.0↙
    .1/UniversityPortalWeb/Personals/Referees', NULL, CAST(0x079A208EF8328D340B AS         ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],     ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],       ↙
    [DeletedOn], [DeletedBy]) VALUES (N'PERS025', 7, N'ACU', N'Religions', N'http://127.0.↙
    0.1/UniversityPortalWeb/Personals/Religions', NULL, CAST(0x074C5A8E1C338D340B AS      ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],     ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],       ↙
    [DeletedOn], [DeletedBy]) VALUES (N'PERS026', 7, N'ACU', N'Sponsors', N'http://127.0.0↙
    .1/UniversityPortalWeb/Personals/Sponsors', NULL, CAST(0x077EEB8C33338D340B AS      ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],  ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],    ↙
    [DeletedOn], [DeletedBy]) VALUES (N'PERS027', 7, N'ACU', N'Sports', N'http://127.0.0.1↙
    /UniversityPortalWeb/Personals/Sports', NULL, CAST(0x07E11ECB42338D340B AS DateTime2),↙
     N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],  ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],    ↙
    [DeletedOn], [DeletedBy]) VALUES (N'REGT001', 5, N'ACU', N'Admissions', N'http://127.0↙
    .0.1/UniversityPortalWeb/Registry/Admissions', NULL, CAST(0x07D14D2069338D340B AS   ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],  ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],    ↙
    [DeletedOn], [DeletedBy]) VALUES (N'REGT002', 5, N'ACU', N'Exams', N'http://127.0.0.1/↙
    UniversityPortalWeb/Registry/Exams', NULL, CAST(0x07C3ADF696338D340B AS DateTime2), N ↙
    'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],  ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],    ↙
    [DeletedOn], [DeletedBy]) VALUES (N'REGT003', 5, N'ACU', N'Students', N'http://127.0.0↙
    .1/UniversityPortalWeb/Registry/Students', NULL, CAST(0x07348C1BAE338D340B AS       ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],  ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],    ↙
    [DeletedOn], [DeletedBy]) VALUES (N'REGT004', 5, N'ACU', N'Tickets', N'http://127.0.0.↙
    1/UniversityPortalWeb/Registry/Tickets', NULL, CAST(0x07284E57CD338D340B AS DateTime2)↙
    , N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],  ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],    ↙
    [DeletedOn], [DeletedBy]) VALUES (N'STUP001', 3, N'ACU', N'Approvals', N'http://127.0.↙
    0.1/UniversityPortalWeb/SetUp/Approvals', NULL, CAST(0x071F0FFF16348D340B AS        ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],  ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],    ↙
    [DeletedOn], [DeletedBy]) VALUES (N'STUP002', 3, N'ACU', N'Area of Specialization', N ↙
    'http://127.0.0.1/UniversityPortalWeb/SetUp/AreasOfSpecializations', NULL, CAST     ↙
    (0x075169FF55488D340B AS DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],  ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],    ↙
    [DeletedOn], [DeletedBy]) VALUES (N'STUP003', 3, N'ACU', N'Country', N'http://127.0.0.↙
    1/UniversityPortalWeb/SetUp/Countries', NULL, CAST(0x07DB71E377488D340B AS DateTime2),↙
     N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],  ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],    ↙
    [DeletedOn], [DeletedBy]) VALUES (N'STUP004', 3, N'ACU', N'Course Numbering', N'http:/↙
    /127.0.0.1/UniversityPortalWeb/SetUp/CourseNumbering', NULL, CAST(0x078DF8E692488D340B↙
     AS DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],  ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],    ↙
    [DeletedOn], [DeletedBy]) VALUES (N'STUP005', 3, N'ACU', N'Course Regulation', N'http:↙
    //127.0.0.1/UniversityPortalWeb/SetUp/CourseRegulation', NULL, CAST                 ↙
    (0x073A7187B2488D340B AS DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],  ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],    ↙
    [DeletedOn], [DeletedBy]) VALUES (N'STUP006', 3, N'ACU', N'Courses', N'http://127.0.0.↙
    1/UniversityPortalWeb/SetUp/Courses', NULL, CAST(0x0757B1E2CA488D340B AS DateTime2), N↙
    'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],  ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],    ↙
    [DeletedOn], [DeletedBy]) VALUES (N'STUP007', 3, N'ACU', N'Departments', N'http://127.↙
    0.0.1/UniversityPortalWeb/SetUp/Departments', NULL, CAST(0x076F8D01E9488D340B AS    ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],  ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],    ↙
    [DeletedOn], [DeletedBy]) VALUES (N'STUP008', 3, N'ACU', N'Descriptions', N'http://127↙
    .0.0.1/UniversityPortalWeb/SetUp/Descriptions', NULL, CAST(0x07BED91907498D340B AS  ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
```

```
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'STUP009', 3, N'ACU', N'Designations', N'http://127↙
    .0.0.1/UniversityPortalWeb/SetUp/Designations', NULL, CAST(0x07CC04CF1D498D340B AS    ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'STUP010', 3, N'ACU', N'Durations', N'http://127.0.↙
    0.1/UniversityPortalWeb/SetUp/Durations', NULL, CAST(0x07FD610245498D340B AS          ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'STUP011', 3, N'ACU', N'Entry Requirement', N'http:↙
    //127.0.0.1/UniversityPortalWeb/SetUp/EntryRequirements', NULL, CAST                  ↙
    (0x078A7AD55E498D340B AS DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'STUP012', 3, N'ACU', N'Faculties', N'http://127.0.↙
    0.1/UniversityPortalWeb/SetUp/Faculties', NULL, CAST(0x07B659397B498D340B AS          ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'STUP013', 3, N'ACU', N'Grading System', N'http:// ↙
    127.0.0.1/UniversityPortalWeb/SetUp/GradingSystem', NULL, CAST(0x07811E0392498D340B AS↙
     DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'STUP014', 3, N'ACU', N'Local Government Arae', N  ↙
    'http://127.0.0.1/UniversityPortalWeb/SetUp/LGAs', NULL, CAST(0x07327464AF498D340B AS ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'STUP015', 3, N'ACU', N'Lockings', N'http://127.0.0↙
    .1/UniversityPortalWeb/SetUp/Lockings', NULL, CAST(0x07EC503AC4498D340B AS DateTime2),↙
     N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'STUP016', 3, N'ACU', N'Modules', N'http://127.0.0.↙
    1/UniversityPortalWeb/SetUp/Modules', NULL, CAST(0x0788012F7C4A8D340B AS DateTime2), N↙
    'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'STUP017', 3, N'ACU', N'Parameters', N'http://127.0↙
    .0.1/UniversityPortalWeb/SetUp/Parameters', NULL, CAST(0x0737F2858E4A8D340B AS        ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'STUP018', 3, N'ACU', N'Program Requirement', N    ↙
    'http://127.0.0.1/UniversityPortalWeb/SetUp/ProgramRequirements', NULL, CAST          ↙
    (0x0793B43DC54A8D340B AS DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'STUP019', 3, N'ACU', N'Programs', N'http://127.0.0↙
    .1/UniversityPortalWeb/SetUp/Programs', NULL, CAST(0x07BEE083E24A8D340B AS DateTime2),↙
     N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'STUP020', 3, N'ACU', N'Schedules', N'http://127.0.↙
    0.1/UniversityPortalWeb/SetUp/Schedules', NULL, CAST(0x0789059AF54A8D340B AS          ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'STUP021', 3, N'ACU', N'Screens', N'http://127.0.0.↙
    1/UniversityPortalWeb/SetUp/Screens', NULL, CAST(0x07CE69B70C4B8D340B AS DateTime2), N↙
    'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],    ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],      ↙
    [DeletedOn], [DeletedBy]) VALUES (N'STUP022', 3, N'ACU', N'Staff', N'http://127.0.0.1/↙
    UniversityPortalWeb/SetUp/Staff', NULL, CAST(0x07C67A6A494B8D340B AS DateTime2), N     ↙
    'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],     ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],       ↙
    [DeletedOn], [DeletedBy]) VALUES (N'STUP023', 3, N'ACU', N'States', N'http://127.0.0.1↙
    /UniversityPortalWeb/SetUp/States', NULL, CAST(0x0756B97D5C4B8D340B AS DateTime2), N   ↙
    'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],     ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],       ↙
    [DeletedOn], [DeletedBy]) VALUES (N'STUP024', 3, N'ACU', N'Sub. Courses', N'http://127↙
    .0.0.1/UniversityPortalWeb/SetUp/SubCourses', NULL, CAST(0x07EF6CDC094D8D340B AS       ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],     ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],       ↙
    [DeletedOn], [DeletedBy]) VALUES (N'STUP025', 3, N'ACU', N'Subject Requirement Details↙
    ', N'http://127.0.0.1/UniversityPortalWeb/SetUp/SubjectRquirementDetails', NULL, CAST ↙
    (0x076767D73C4D8D340B AS DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],     ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],       ↙
    [DeletedOn], [DeletedBy]) VALUES (N'STUP026', 3, N'ACU', N'Subject Requirements', N    ↙
    'http://127.0.0.1/UniversityPortalWeb/SetUp/SubjectRquirements', NULL, CAST           ↙
    (0x07FEFA1F544D8D340B AS DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
INSERT [SetUp].[Screens] ([Code], [ModuleCode], [UniversityCode], [ScreenDescription],     ↙
    [Url], [Notes], [CreatedOn], [CreatedBy], [ModifiedOn], [ModifiedBy], [Deleted],       ↙
    [DeletedOn], [DeletedBy]) VALUES (N'STUP027', 3, N'ACU', N'Universities', N'http://127↙
    .0.0.1/UniversityPortalWeb/SetUp/Universities', NULL, CAST(0x07D759F2734D8D340B AS     ↙
    DateTime2), N'Ademola', NULL, NULL, 0, NULL, NULL)
/****** Object:  Table [Academics].[Sessions]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Academics].[Sessions](
    [Code] [bigint] IDENTITY(1,1) NOT NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [CurrentYear] [int] NULL,
    [NextYear] [int] NULL,
    [SessionStatus] [nvarchar](50) NULL,
    [StartDate] [datetime2](7) NULL,
    [EndDate] [datetime2](7) NULL,
    [Applicable] [bit] NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
    [SessionDescription]  AS ([Academics].[GetSessionDescription]([CurrentYear],           ↙
    [NextYear])),
    [SessionStatusDescription]  AS ([Academics].[GetSessionStatusDescription](             ↙
    [SessionStatus])),
 CONSTRAINT [PK_Sessions] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,              ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
SET IDENTITY_INSERT [Academics].[Sessions] ON
INSERT [Academics].[Sessions] ([Code], [UniversityCode], [CurrentYear], [NextYear],        ↙
    [SessionStatus], [StartDate], [EndDate], [Applicable], [CreatedOn], [CreatedBy],       ↙
    [ModifiedOn], [ModifiedBy], [Deleted], [DeletedOn], [DeletedBy]) VALUES (1, N'ACU',    ↙
    2011, 2012, N'SSS-75', CAST(0x07000000000092340B AS DateTime2), CAST                   ↙
    (0x07000000000092350B AS DateTime2), 1, CAST(0x07000000000092340B AS DateTime2), N     ↙
    'Godwin Mathias', NULL, NULL, 0, NULL, NULL)
```

```
SET IDENTITY_INSERT [Academics].[Sessions] OFF
/****** Object:  StoredProcedure [Assessment].[SPAttendanceSelect]    Script Date: 08/16/ ↙
    2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Assessment.Attendance ↙
    Table

CREATE PROC [Assessment].[SPAttendanceSelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Deleted BIT=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Assessment].[Attendance] WHERE (([Code] = @Code) AND (  ↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted)))
            BEGIN
            SELECT * FROM [Assessment].[Attendance]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Assessment].[Attendance]
            WHERE
                (
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);
```

```sql
END CATCH

GRANT EXECUTE ON [Assessment].[SPAttendanceSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Assessment].[SPAttendanceInsertUpdate]    Script Date: 08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/18/2011
--Last Updated        :   07/18/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for saving values into Assessment.Attendance Table

CREATE PROC [Assessment].[SPAttendanceInsertUpdate]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @FacultyCode BIGINT=NULL,
    @DepartmentCode BIGINT=NULL,
    @CourseCode BIGINT=NULL,
    @ProgramCode BIGINT=NULL,
    @LevelCode INT=NULL,
    @StudentCode BIGINT=NULL,
    @MatricNo NVARCHAR(50)=NULL,
    @SessionCode BIGINT=NULL,
    @SemesterCode BIGINT=NULL,
    @AttendanceStatus NVARCHAR(50)=NULL,
    @StaffCode NVARCHAR(50)=NULL,
    @SubCourseCode BIGINT=NULL,
    @AttendanceType NVARCHAR(50)=NULL,
    @Remark NCHAR(10)=NULL,
    @ScreenCode NVARCHAR(50)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Assessment].[Attendance] WHERE ([Code] = @Code) AND ([UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
                INSERT INTO [Assessment].[Attendance]
                (
                    [UniversityCode],
                    [FacultyCode],
                    [DepartmentCode],
                    [CourseCode],
                    [ProgramCode],
                    [LevelCode],
                    [StudentCode],
                    [MatricNo],
                    [SessionCode],
                    [SemesterCode],
                    [AttendanceStatus],
                    [StaffCode],
                    [SubCourseCode],
                    [AttendanceType],
                    [Remark],
                    [ScreenCode],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]

                )
                VALUES
                (
                    @UniversityCode,
                    @FacultyCode,
                    @DepartmentCode,
                    @CourseCode,
                    @ProgramCode,
                    @LevelCode,
                    @StudentCode,
                    @MatricNo,
                    @SessionCode,
                    @SemesterCode,
                    @AttendanceStatus,
                    @StaffCode,
                    @SubCourseCode,
                    @AttendanceType,
                    @Remark,
                    @ScreenCode,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted

                )
            END
        ELSE
            BEGIN
                UPDATE [Assessment].[Attendance]
                SET
                    [UniversityCode]=@UniversityCode,
                    [FacultyCode]=@FacultyCode,
                    [DepartmentCode]=@DepartmentCode,
                    [CourseCode]=@CourseCode,
                    [ProgramCode]=@ProgramCode,
                    [LevelCode]=@LevelCode,
                    [StudentCode]=@StudentCode,
                    [MatricNo]=@MatricNo,
                    [SessionCode]=@SessionCode,
                    [SemesterCode]=@SemesterCode,
                    [AttendanceStatus]=@AttendanceStatus,
                    [StaffCode]=@StaffCode,
                    [SubCourseCode]=@SubCourseCode,
                    [AttendanceType]=@AttendanceType,
                    [Remark]=@Remark,
                    [ScreenCode]=@ScreenCode,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted

                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN
```

```
DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Assessment].[SPAttendanceInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Assessment].[SPAttendanceDeletePermanently]    Script   ↙
    Date: 08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Assessment.Attendance   ↙
    Table

CREATE PROC [Assessment].[SPAttendanceDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Assessment].[Attendance] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Assessment].[Attendance]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Assessment].[SPAttendanceDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Assessment].[SPAttendanceDelete]    Script Date: 08/16/ ↙
    2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Assessment. ↙
    Attendance Table

CREATE PROC [Assessment].[SPAttendanceDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Assessment].[Attendance] WHERE ([Code] = @Code) AND     ↙
    (Deleted=@Deleted))
            BEGIN
            UPDATE [Assessment].[Attendance]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Assessment].[SPAttendanceDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [dbo].[aspnet_Users_DeleteUser]    Script Date: 08/16/   ↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_Users_DeleteUser]
    @ApplicationName  nvarchar(256),
    @UserName         nvarchar(256),
    @TablesToDeleteFrom int,
    @NumTablesDeletedFrom int OUTPUT
AS
BEGIN
    DECLARE @UserId               uniqueidentifier
    SELECT  @UserId               = NULL
    SELECT  @NumTablesDeletedFrom = 0

    DECLARE @TranStarted   bit
    SET @TranStarted = 0

    IF( @@TRANCOUNT = 0 )
    BEGIN
        BEGIN TRANSACTION
        SET @TranStarted = 1
    END
    ELSE
    SET @TranStarted = 0

    DECLARE @ErrorCode   int
    DECLARE @RowCount    int

    SET @ErrorCode = 0
    SET @RowCount  = 0

    SELECT  @UserId = u.UserId
    FROM    dbo.aspnet_Users u, dbo.aspnet_Applications a
    WHERE   u.LoweredUserName       = LOWER(@UserName)
        AND u.ApplicationId         = a.ApplicationId
        AND LOWER(@ApplicationName) = a.LoweredApplicationName

    IF (@UserId IS NULL)
    BEGIN
        GOTO Cleanup
    END

    -- Delete from Membership table if (@TablesToDeleteFrom & 1) is set
    IF ((@TablesToDeleteFrom & 1) <> 0 AND
        (EXISTS (SELECT name FROM sysobjects WHERE (name = N'vw_aspnet_MembershipUsers')  ↙
    AND (type = 'V'))))
    BEGIN
        DELETE FROM dbo.aspnet_Membership WHERE @UserId = UserId

        SELECT @ErrorCode = @@ERROR,
               @RowCount = @@ROWCOUNT

        IF( @ErrorCode <> 0 )
            GOTO Cleanup

        IF (@RowCount <> 0)
```

```
            SELECT  @NumTablesDeletedFrom = @NumTablesDeletedFrom + 1
    END

    -- Delete from aspnet_UsersInRoles table if (@TablesToDeleteFrom & 2) is set
    IF ((@TablesToDeleteFrom & 2) <> 0  AND
        (EXISTS (SELECT name FROM sysobjects WHERE (name = N'vw_aspnet_UsersInRoles') AND ↙
    (type = 'V'))) )
    BEGIN
        DELETE FROM dbo.aspnet_UsersInRoles WHERE @UserId = UserId

        SELECT @ErrorCode = @@ERROR,
                @RowCount = @@ROWCOUNT

        IF( @ErrorCode <> 0 )
            GOTO Cleanup

        IF (@RowCount <> 0)
            SELECT  @NumTablesDeletedFrom = @NumTablesDeletedFrom + 1
    END

    -- Delete from aspnet_Profile table if (@TablesToDeleteFrom & 4) is set
    IF ((@TablesToDeleteFrom & 4) <> 0  AND
        (EXISTS (SELECT name FROM sysobjects WHERE (name = N'vw_aspnet_Profiles') AND     ↙
    (type = 'V'))) )
    BEGIN
        DELETE FROM dbo.aspnet_Profile WHERE @UserId = UserId

        SELECT @ErrorCode = @@ERROR,
                @RowCount = @@ROWCOUNT

        IF( @ErrorCode <> 0 )
            GOTO Cleanup

        IF (@RowCount <> 0)
            SELECT  @NumTablesDeletedFrom = @NumTablesDeletedFrom + 1
    END

    -- Delete from aspnet_PersonalizationPerUser table if (@TablesToDeleteFrom & 8) is set
    IF ((@TablesToDeleteFrom & 8) <> 0  AND
        (EXISTS (SELECT name FROM sysobjects WHERE (name = N'vw_aspnet_WebPartState_User')↙
     AND (type = 'V'))) )
    BEGIN
        DELETE FROM dbo.aspnet_PersonalizationPerUser WHERE @UserId = UserId

        SELECT @ErrorCode = @@ERROR,
                @RowCount = @@ROWCOUNT

        IF( @ErrorCode <> 0 )
            GOTO Cleanup

        IF (@RowCount <> 0)
            SELECT  @NumTablesDeletedFrom = @NumTablesDeletedFrom + 1
    END

    -- Delete from aspnet_Users table if (@TablesToDeleteFrom & 1,2,4 & 8) are all set
    IF ((@TablesToDeleteFrom & 1) <> 0 AND
        (@TablesToDeleteFrom & 2) <> 0 AND
        (@TablesToDeleteFrom & 4) <> 0 AND
        (@TablesToDeleteFrom & 8) <> 0 AND
        (EXISTS (SELECT UserId FROM dbo.aspnet_Users WHERE @UserId = UserId)))
    BEGIN
        DELETE FROM dbo.aspnet_Users WHERE @UserId = UserId

        SELECT @ErrorCode = @@ERROR,
                @RowCount = @@ROWCOUNT

        IF( @ErrorCode <> 0 )
```

```
            GOTO Cleanup

        IF (@RowCount <> 0)
            SELECT  @NumTablesDeletedFrom = @NumTablesDeletedFrom + 1
    END

    IF( @TranStarted = 1 )
    BEGIN
        SET @TranStarted = 0
        COMMIT TRANSACTION
    END

    RETURN 0

Cleanup:
    SET @NumTablesDeletedFrom = 0

    IF( @TranStarted = 1 )
    BEGIN
        SET @TranStarted = 0
        ROLLBACK TRANSACTION
    END

    RETURN @ErrorCode

END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_UsersInRolesSelect]    Script Date: 08/16/ ↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Godwin Mathias
--Reviewer          :   Godwin Mathias
--Date Created      :   05/16/2011
--Last Updated      :   05/16/2011
--Last Updated By   :   Godwin Mathias
--Description       :   Stored procedure for retrieving values in the dbo.               ↙
    aspnet_UsersInRoles  Table

CREATE PROC [dbo].[aspnet_UsersInRolesSelect]
(
    @UserId uniqueidentifier = null,
    @RoleId uniqueidentifier = null,
    @SchoolCode nvarchar(50) = null,
    @BranchCode NVARCHAR(50) = NULL,
    @Deleted BIT = NULL
)

AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM  [dbo].[aspnet_UsersInRoles] WHERE ((UserId = @UserId)   ↙
    AND (RoleId=@RoleId) AND (SchoolCode=@SchoolCode) AND (BranchCode=@BranchCode) AND   ↙
    (Deleted=@Deleted)))

            BEGIN
                SELECT * FROM  [dbo].[aspnet_UsersInRoles]
                WHERE ((UserId = @UserId) AND (RoleId=@RoleId) AND (SchoolCode=@          ↙
    SchoolCode) AND (BranchCode=@BranchCode) AND (Deleted=@Deleted))
            END
        ELSE
            BEGIN
                SELECT * FROM  [dbo].[aspnet_UsersInRoles]
                WHERE ((SchoolCode=@SchoolCode) AND (BranchCode=@BranchCode) AND (Deleted=↙
    @Deleted))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [dbo].[aspnet_UsersInRolesSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [dbo].[aspnet_UsersInRolesInsertUpdate]    Script Date:  ↙
    08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Godwin Mathias
--Reviewer          :   Godwin Mathias
--Date Created      :   05/18/2011
--Last Updated      :   05/18/2011
--Last Updated By   :   Godwin Mathias
--Description       :   Stored procedure for inserting/updating values in  [dbo].       ↙
    [aspnet_UsersInBranches] Table

CREATE PROC [dbo].[aspnet_UsersInRolesInsertUpdate]
(
    @UserId UNIQUEIDENTIFIER = Null,
    @RoleId UNIQUEIDENTIFIER = null,
    @SchoolCode NVARCHAR(50) = null,
    @BranchCode NVARCHAR(50) = null,
    @CreatedOn DATETIME2(7) = null,
    @CreatedBy NVARCHAR(50) = null,
    @ModifiedOn DATETIME2(7) = null,
    @ModifiedBy NVARCHAR(50) = null,
    @Deleted BIT = null
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [dbo].[aspnet_UsersInRoles] WHERE ((UserId = @UserId)↙
     AND (RoleId=@RoleId) AND (SchoolCode=@SchoolCode) AND (BranchCode=@BranchCode)))
            BEGIN
                INSERT INTO [dbo].[aspnet_UsersInRoles]
                (
                    [UserId],
                    [RoleId],
                    [SchoolCode],
                    [BranchCode],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    @UserId,
                    @RoleId,
                    @SchoolCode,
                    @BranchCode,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [dbo].[aspnet_UsersInRoles]
                SET
                    [UserId]=@UserId,
                    [RoleId]=@RoleId,
                    [SchoolCode]=@SchoolCode,
                    [BranchCode]=@BranchCode,
                    [ModifiedOn] = @ModifiedOn,
                    [ModifiedBy] = @ModifiedBy,
                    [Deleted]=@Deleted

                WHERE ((UserId = @UserId) AND (RoleId=@RoleId) AND (SchoolCode=@         ↙
    SchoolCode) AND (BranchCode=@BranchCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON  [dbo].[aspnet_UsersInRolesInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [dbo].[aspnet_UsersInRolesDeletePermanently]    Script   ↙
    Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author                :   Godwin Mathias
--Reviewer              :   Godwin Mathias
--Date Created          :   05/16/2011
--Last Updated          :   05/16/2011
--Last Updated By       :   Godwin Mathias
```

```
--Description        :   Stored procedure for deleting permanently values in the dbo.  ↙
    aspnet_UsersInRoles  Table

CREATE PROC [dbo].[aspnet_UsersInRolesDeletePermanently]
(
    @UserId uniqueidentifier = null,
    @RoleId uniqueidentifier = null,
    @SchoolCode nvarchar(50) = null,
    @BranchCode NVARCHAR(50) = NULL
)

AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM  [dbo].[aspnet_UsersInRoles] WHERE ((UserId = @UserId)  ↙
    AND (RoleId=@RoleId) AND (SchoolCode=@SchoolCode) AND (BranchCode=@BranchCode)))

            BEGIN
                DELETE FROM [dbo].[aspnet_UsersInRoles]
                WHERE ((UserId = @UserId) AND (RoleId=@RoleId) AND (SchoolCode=@  ↙
    SchoolCode) AND (BranchCode=@BranchCode))
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [dbo].[aspnet_UsersInRolesDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [dbo].[aspnet_UsersInRolesDelete]    Script Date: 08/16/ ↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author             :   Godwin Mathias
--Reviewer           :   Godwin Mathias
--Date Created       :   05/16/2011
--Last Updated       :   05/18/2011
--Last Updated By    :   Godwin Mathias
--Description        :   Stored procedure for retrieving values in the dbo.  ↙
    aspnet_UsersInRoles  Table

CREATE PROC [dbo].[aspnet_UsersInRolesDelete]
(
    @RoleId uniqueidentifier = null,
    @UserId uniqueidentifier = null,
    @SchoolCode nvarchar(50) = null,
    @BranchCode NVARCHAR(50) = NULL,
    @Deleted BIT = NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)

AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM  [dbo].[aspnet_UsersInRoles] WHERE ((RoleId = @RoleId)  ↙
    AND (UserId=@UserId) AND (SchoolCode=@SchoolCode) AND (BranchCode=@BranchCode)))

            BEGIN
                UPDATE  [dbo].[aspnet_UsersInRoles]
                SET
                    Deleted=@Deleted,
                    DeletedOn=@DeletedOn,
                    DeletedBy=@DeletedBy
                WHERE (([RoleId] = @RoleId) AND ([UserId]=@UserId) AND ([SchoolCode]=@  ↙
    SchoolCode) AND ([BranchCode]=@BranchCode))
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [dbo].[aspnet_UsersInRolesDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [dbo].[aspnet_UsersInRoles_RemoveUsersFromRoles]  ↙
    Script Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_UsersInRoles_RemoveUsersFromRoles]
    @ApplicationName  nvarchar(256),
    @UserNames        nvarchar(4000),
    @RoleNames        nvarchar(4000)
AS
BEGIN
    DECLARE @AppId uniqueidentifier
    SELECT  @AppId = NULL
    SELECT  @AppId = ApplicationId FROM aspnet_Applications WHERE LOWER(@ApplicationName) ↙
    = LoweredApplicationName
    IF (@AppId IS NULL)
        RETURN(2)


    DECLARE @TranStarted   bit
    SET @TranStarted = 0

    IF( @@TRANCOUNT = 0 )
    BEGIN
        BEGIN TRANSACTION
        SET @TranStarted = 1
    END

    DECLARE @tbNames  table(Name nvarchar(256) NOT NULL PRIMARY KEY)
    DECLARE @tbRoles  table(RoleId uniqueidentifier NOT NULL PRIMARY KEY)
    DECLARE @tbUsers  table(UserId uniqueidentifier NOT NULL PRIMARY KEY)
    DECLARE @Num      int
    DECLARE @Pos      int
    DECLARE @NextPos  int
    DECLARE @Name     nvarchar(256)
    DECLARE @CountAll int
    DECLARE @CountU   int
    DECLARE @CountR   int


    SET @Num = 0
    SET @Pos = 1
    WHILE(@Pos <= LEN(@RoleNames))
    BEGIN
        SELECT @NextPos = CHARINDEX(N',', @RoleNames,  @Pos)
        IF (@NextPos = 0 OR @NextPos IS NULL)
            SELECT @NextPos = LEN(@RoleNames) + 1
        SELECT @Name = RTRIM(LTRIM(SUBSTRING(@RoleNames, @Pos, @NextPos - @Pos)))
        SELECT @Pos = @NextPos+1

        INSERT INTO @tbNames VALUES (@Name)
        SET @Num = @Num + 1
    END

    INSERT INTO @tbRoles
      SELECT RoleId
      FROM   dbo.aspnet_Roles ar, @tbNames t
      WHERE  LOWER(t.Name) = ar.LoweredRoleName AND ar.ApplicationId = @AppId
    SELECT @CountR = @@ROWCOUNT

    IF (@CountR <> @Num)
    BEGIN
        SELECT TOP 1 N'', Name
        FROM   @tbNames
        WHERE  LOWER(Name) NOT IN (SELECT ar.LoweredRoleName FROM dbo.aspnet_Roles ar,  @ ↙
    tbRoles r WHERE r.RoleId = ar.RoleId)
        IF( @TranStarted = 1 )
            ROLLBACK TRANSACTION
        RETURN(2)
    END


    DELETE FROM @tbNames WHERE 1=1
    SET @Num = 0
    SET @Pos = 1


    WHILE(@Pos <= LEN(@UserNames))
    BEGIN
        SELECT @NextPos = CHARINDEX(N',', @UserNames,  @Pos)
        IF (@NextPos = 0 OR @NextPos IS NULL)
            SELECT @NextPos = LEN(@UserNames) + 1
        SELECT @Name = RTRIM(LTRIM(SUBSTRING(@UserNames, @Pos, @NextPos - @Pos)))
        SELECT @Pos = @NextPos+1

        INSERT INTO @tbNames VALUES (@Name)
        SET @Num = @Num + 1
    END

    INSERT INTO @tbUsers
      SELECT UserId
      FROM   dbo.aspnet_Users ar, @tbNames t
      WHERE  LOWER(t.Name) = ar.LoweredUserName AND ar.ApplicationId = @AppId

    SELECT @CountU = @@ROWCOUNT
    IF (@CountU <> @Num)
    BEGIN
        SELECT TOP 1 Name, N''
        FROM   @tbNames
        WHERE  LOWER(Name) NOT IN (SELECT au.LoweredUserName FROM dbo.aspnet_Users au,  @ ↙
    tbUsers u WHERE u.UserId = au.UserId)

        IF( @TranStarted = 1 )
            ROLLBACK TRANSACTION
        RETURN(1)
    END

    SELECT  @CountAll = COUNT(*)
    FROM    dbo.aspnet_UsersInRoles ur, @tbUsers u, @tbRoles r
    WHERE   ur.UserId = u.UserId AND ur.RoleId = r.RoleId

    IF (@CountAll <> @CountU * @CountR)
    BEGIN
        SELECT TOP 1 UserName, RoleName
        FROM         @tbUsers tu, @tbRoles tr, dbo.aspnet_Users u, dbo.aspnet_Roles r
        WHERE        u.UserId = tu.UserId AND r.RoleId = tr.RoleId AND
                     tu.UserId NOT IN (SELECT ur.UserId FROM dbo.aspnet_UsersInRoles ur   ↙
    WHERE ur.RoleId = tr.RoleId) AND
                     tr.RoleId NOT IN (SELECT ur.RoleId FROM dbo.aspnet_UsersInRoles ur   ↙
    WHERE ur.UserId = tu.UserId)
        IF( @TranStarted = 1 )
            ROLLBACK TRANSACTION
        RETURN(3)
    END

    DELETE FROM dbo.aspnet_UsersInRoles
    WHERE UserId IN (SELECT UserId FROM @tbUsers)
      AND RoleId IN (SELECT RoleId FROM @tbRoles)
    IF( @TranStarted = 1 )
        COMMIT TRANSACTION
    RETURN(0)
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_UsersInRoles_IsUserInRole]    Script Date: ↙
    08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_UsersInRoles_IsUserInRole]
    @ApplicationName  nvarchar(256),
    @UserName         nvarchar(256),
    @RoleName         nvarchar(256)
AS
BEGIN
    DECLARE @ApplicationId uniqueidentifier
    SELECT  @ApplicationId = NULL
    SELECT  @ApplicationId = ApplicationId FROM aspnet_Applications WHERE LOWER(@         ↙
    ApplicationName) = LoweredApplicationName
    IF (@ApplicationId IS NULL)
        RETURN(2)
    DECLARE @UserId uniqueidentifier
    SELECT  @UserId = NULL
    DECLARE @RoleId uniqueidentifier
    SELECT  @RoleId = NULL

    SELECT  @UserId = UserId
    FROM    dbo.aspnet_Users
    WHERE   LoweredUserName = LOWER(@UserName) AND ApplicationId = @ApplicationId

    IF (@UserId IS NULL)
        RETURN(2)

    SELECT  @RoleId = RoleId
    FROM    dbo.aspnet_Roles
    WHERE   LoweredRoleName = LOWER(@RoleName) AND ApplicationId = @ApplicationId

    IF (@RoleId IS NULL)
        RETURN(3)

    IF (EXISTS( SELECT * FROM dbo.aspnet_UsersInRoles WHERE  UserId = @UserId AND RoleId =↙
     @RoleId))
        RETURN(1)
    ELSE
        RETURN(0)
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_UsersInRoles_GetUsersInRoles]    Script    ↙
    Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_UsersInRoles_GetUsersInRoles]
    @ApplicationName  nvarchar(256),
    @RoleName         nvarchar(256)
AS
BEGIN
    DECLARE @ApplicationId uniqueidentifier
    SELECT  @ApplicationId = NULL
    SELECT  @ApplicationId = ApplicationId FROM aspnet_Applications WHERE LOWER(@         ↙
    ApplicationName) = LoweredApplicationName
    IF (@ApplicationId IS NULL)
        RETURN(1)
     DECLARE @RoleId uniqueidentifier
     SELECT  @RoleId = NULL

     SELECT  @RoleId = RoleId
     FROM    dbo.aspnet_Roles
     WHERE   LOWER(@RoleName) = LoweredRoleName AND ApplicationId = @ApplicationId

     IF (@RoleId IS NULL)
         RETURN(1)

    SELECT u.UserName
    FROM   dbo.aspnet_Users u, dbo.aspnet_UsersInRoles ur
    WHERE  u.UserId = ur.UserId AND @RoleId = ur.RoleId AND u.ApplicationId = @           ↙
    ApplicationId
    ORDER BY u.UserName
    RETURN(0)
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_UsersInRoles_GetRolesForUser]    Script    ↙
    Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
```

```
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_UsersInRoles_GetRolesForUser]
    @ApplicationName  nvarchar(256),
    @UserName         nvarchar(256)
AS
BEGIN
    DECLARE @ApplicationId uniqueidentifier
    SELECT  @ApplicationId = NULL
    SELECT  @ApplicationId = ApplicationId FROM aspnet_Applications WHERE LOWER(@ ↙
    ApplicationName) = LoweredApplicationName
    IF (@ApplicationId IS NULL)
        RETURN(1)
    DECLARE @UserId uniqueidentifier
    SELECT  @UserId = NULL

    SELECT  @UserId = UserId
    FROM    dbo.aspnet_Users
    WHERE   LoweredUserName = LOWER(@UserName) AND ApplicationId = @ApplicationId

    IF (@UserId IS NULL)
        RETURN(1)

    SELECT r.RoleName
    FROM   dbo.aspnet_Roles r, dbo.aspnet_UsersInRoles ur
    WHERE  r.RoleId = ur.RoleId AND r.ApplicationId = @ApplicationId AND ur.UserId = @ ↙
    UserId
    ORDER BY r.RoleName
    RETURN (0)
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_UsersInRoles_FindUsersInRole]    Script ↙
    Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_UsersInRoles_FindUsersInRole]
    @ApplicationName  nvarchar(256),
    @RoleName         nvarchar(256),
    @UserNameToMatch  nvarchar(256)
AS
BEGIN
    DECLARE @ApplicationId uniqueidentifier
    SELECT  @ApplicationId = NULL
    SELECT  @ApplicationId = ApplicationId FROM aspnet_Applications WHERE LOWER(@ ↙
    ApplicationName) = LoweredApplicationName
    IF (@ApplicationId IS NULL)
        RETURN(1)
     DECLARE @RoleId uniqueidentifier
     SELECT  @RoleId = NULL

     SELECT  @RoleId = RoleId
     FROM    dbo.aspnet_Roles
     WHERE   LOWER(@RoleName) = LoweredRoleName AND ApplicationId = @ApplicationId

     IF (@RoleId IS NULL)
         RETURN(1)

    SELECT u.UserName
    FROM   dbo.aspnet_Users u, dbo.aspnet_UsersInRoles ur
    WHERE  u.UserId = ur.UserId AND @RoleId = ur.RoleId AND u.ApplicationId = @ ↙
    ApplicationId AND LoweredUserName LIKE LOWER(@UserNameToMatch)
    ORDER BY u.UserName
    RETURN(0)
END
GO
```

```
/****** Object:  StoredProcedure [dbo].[aspnet_UsersInRoles_AddUsersToRoles]    Script   ↙
    Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_UsersInRoles_AddUsersToRoles]
    @ApplicationName  nvarchar(256),
    @UserNames        nvarchar(4000),
    @RoleNames        nvarchar(4000),
    @CurrentTimeUtc   datetime
AS
BEGIN
    DECLARE @AppId uniqueidentifier
    SELECT  @AppId = NULL
    SELECT  @AppId = ApplicationId FROM aspnet_Applications WHERE LOWER(@ApplicationName) ↙
    = LoweredApplicationName
    IF (@AppId IS NULL)
        RETURN(2)
    DECLARE @TranStarted   bit
    SET @TranStarted = 0

    IF( @@TRANCOUNT = 0 )
    BEGIN
        BEGIN TRANSACTION
        SET @TranStarted = 1
    END

    DECLARE @tbNames    table(Name nvarchar(256) NOT NULL PRIMARY KEY)
    DECLARE @tbRoles    table(RoleId uniqueidentifier NOT NULL PRIMARY KEY)
    DECLARE @tbUsers    table(UserId uniqueidentifier NOT NULL PRIMARY KEY)
    DECLARE @Num        int
    DECLARE @Pos        int
    DECLARE @NextPos    int
    DECLARE @Name       nvarchar(256)

    SET @Num = 0
    SET @Pos = 1
    WHILE(@Pos <= LEN(@RoleNames))
    BEGIN
        SELECT @NextPos = CHARINDEX(N',', @RoleNames,  @Pos)
        IF (@NextPos = 0 OR @NextPos IS NULL)
            SELECT @NextPos = LEN(@RoleNames) + 1
        SELECT @Name = RTRIM(LTRIM(SUBSTRING(@RoleNames, @Pos, @NextPos - @Pos)))
        SELECT @Pos = @NextPos+1

        INSERT INTO @tbNames VALUES (@Name)
        SET @Num = @Num + 1
    END

    INSERT INTO @tbRoles
      SELECT RoleId
      FROM   dbo.aspnet_Roles ar, @tbNames t
      WHERE  LOWER(t.Name) = ar.LoweredRoleName AND ar.ApplicationId = @AppId

    IF (@@ROWCOUNT <> @Num)
    BEGIN
        SELECT TOP 1 Name
        FROM   @tbNames
        WHERE  LOWER(Name) NOT IN (SELECT ar.LoweredRoleName FROM dbo.aspnet_Roles ar,  @ ↙
    tbRoles r WHERE r.RoleId = ar.RoleId)
        IF( @TranStarted = 1 )
            ROLLBACK TRANSACTION
        RETURN(2)
    END

    DELETE FROM @tbNames WHERE 1=1
    SET @Num = 0
    SET @Pos = 1

    WHILE(@Pos <= LEN(@UserNames))
    BEGIN
        SELECT @NextPos = CHARINDEX(N',', @UserNames,  @Pos)
        IF (@NextPos = 0 OR @NextPos IS NULL)
            SELECT @NextPos = LEN(@UserNames) + 1
        SELECT @Name = RTRIM(LTRIM(SUBSTRING(@UserNames, @Pos, @NextPos - @Pos)))
        SELECT @Pos = @NextPos+1

        INSERT INTO @tbNames VALUES (@Name)
        SET @Num = @Num + 1
    END

    INSERT INTO @tbUsers
      SELECT UserId
      FROM   dbo.aspnet_Users ar, @tbNames t
      WHERE  LOWER(t.Name) = ar.LoweredUserName AND ar.ApplicationId = @AppId

    IF (@@ROWCOUNT <> @Num)
    BEGIN
        DELETE FROM @tbNames
        WHERE LOWER(Name) IN (SELECT LoweredUserName FROM dbo.aspnet_Users au,  @tbUsers u↙
     WHERE au.UserId = u.UserId)

        INSERT dbo.aspnet_Users (ApplicationId, UserId, UserName, LoweredUserName,        ↙
    IsAnonymous, LastActivityDate)
          SELECT @AppId, NEWID(), Name, LOWER(Name), 0, @CurrentTimeUtc
          FROM   @tbNames

        INSERT INTO @tbUsers
          SELECT  UserId
          FROM  dbo.aspnet_Users au, @tbNames t
          WHERE   LOWER(t.Name) = au.LoweredUserName AND au.ApplicationId = @AppId
    END

    IF (EXISTS (SELECT * FROM dbo.aspnet_UsersInRoles ur, @tbUsers tu, @tbRoles tr WHERE  ↙
    tu.UserId = ur.UserId AND tr.RoleId = ur.RoleId))
    BEGIN
        SELECT TOP 1 UserName, RoleName
        FROM         dbo.aspnet_UsersInRoles ur, @tbUsers tu, @tbRoles tr, aspnet_Users u,↙
     aspnet_Roles r
        WHERE       u.UserId = tu.UserId AND r.RoleId = tr.RoleId AND tu.UserId = ur.     ↙
    UserId AND tr.RoleId = ur.RoleId

        IF( @TranStarted = 1 )
            ROLLBACK TRANSACTION
        RETURN(3)
    END

    INSERT INTO dbo.aspnet_UsersInRoles (UserId, RoleId)
    SELECT UserId, RoleId
    FROM @tbUsers, @tbRoles

    IF( @TranStarted = 1 )
        COMMIT TRANSACTION
    RETURN(0)
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_Profile_DeleteInactiveProfiles]    Script  ↙
    Date: 08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_Profile_DeleteInactiveProfiles]
```

```
    @ApplicationName        nvarchar(256),
    @ProfileAuthOptions     int,
    @InactiveSinceDate      datetime
AS
BEGIN
    DECLARE @ApplicationId uniqueidentifier
    SELECT  @ApplicationId = NULL
    SELECT  @ApplicationId = ApplicationId FROM aspnet_Applications WHERE LOWER(@       ↙
    ApplicationName) = LoweredApplicationName
    IF (@ApplicationId IS NULL)
    BEGIN
        SELECT  0
        RETURN
    END

    DELETE
    FROM    dbo.aspnet_Profile
    WHERE   UserId IN
            (   SELECT  UserId
                FROM    dbo.aspnet_Users u
                WHERE   ApplicationId = @ApplicationId
                        AND (LastActivityDate <= @InactiveSinceDate)
                        AND (
                                (@ProfileAuthOptions = 2)
                             OR (@ProfileAuthOptions = 0 AND IsAnonymous = 1)
                             OR (@ProfileAuthOptions = 1 AND IsAnonymous = 0)
                            )
            )

    SELECT  @@ROWCOUNT
END
GO
/****** Object:  StoredProcedure [dbo].                                                ↙
    [aspnet_PersonalizationAdministration_ResetUserState]    Script Date: 08/16/2011 01:24↙
    :27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_PersonalizationAdministration_ResetUserState] (
    @Count                  int                     OUT,
    @ApplicationName        NVARCHAR(256),
    @InactiveSinceDate      DATETIME                = NULL,
    @UserName               NVARCHAR(256)           = NULL,
    @Path                   NVARCHAR(256)           = NULL)
AS
BEGIN
    DECLARE @ApplicationId UNIQUEIDENTIFIER
    EXEC dbo.aspnet_Personalization_GetApplicationId @ApplicationName, @ApplicationId     ↙
    OUTPUT
    IF (@ApplicationId IS NULL)
        SELECT @Count = 0
    ELSE
    BEGIN
        DELETE FROM dbo.aspnet_PersonalizationPerUser
        WHERE Id IN (SELECT PerUser.Id
                     FROM dbo.aspnet_PersonalizationPerUser PerUser, dbo.aspnet_Users     ↙
    Users, dbo.aspnet_Paths Paths
                     WHERE Paths.ApplicationId = @ApplicationId
                           AND PerUser.UserId = Users.UserId
                           AND PerUser.PathId = Paths.PathId
                           AND (@InactiveSinceDate IS NULL OR Users.LastActivityDate <= @ ↙
    InactiveSinceDate)
                           AND (@UserName IS NULL OR Users.LoweredUserName = LOWER(@      ↙
    UserName))
                           AND (@Path IS NULL OR Paths.LoweredPath = LOWER(@Path)))
        SELECT @Count = @@ROWCOUNT
    END
END
GO
/****** Object:  StoredProcedure [dbo].                                                    ↙
    [aspnet_PersonalizationAdministration_ResetSharedState]    Script Date: 08/16/2011 01:↙
    24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_PersonalizationAdministration_ResetSharedState] (
    @Count int OUT,
    @ApplicationName NVARCHAR(256),
    @Path NVARCHAR(256))
AS
BEGIN
    DECLARE @ApplicationId UNIQUEIDENTIFIER
    EXEC dbo.aspnet_Personalization_GetApplicationId @ApplicationName, @ApplicationId     ↙
    OUTPUT
    IF (@ApplicationId IS NULL)
        SELECT @Count = 0
    ELSE
    BEGIN
        DELETE FROM dbo.aspnet_PersonalizationAllUsers
        WHERE PathId IN
            (SELECT AllUsers.PathId
             FROM dbo.aspnet_PersonalizationAllUsers AllUsers, dbo.aspnet_Paths Paths
             WHERE Paths.ApplicationId = @ApplicationId
                   AND AllUsers.PathId = Paths.PathId
                   AND Paths.LoweredPath = LOWER(@Path))

        SELECT @Count = @@ROWCOUNT
    END
END
GO
/****** Object:  StoredProcedure [dbo].                                                    ↙
    [aspnet_PersonalizationAdministration_GetCountOfState]    Script Date: 08/16/2011 01: ↙
    24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_PersonalizationAdministration_GetCountOfState] (
    @Count int OUT,
    @AllUsersScope bit,
    @ApplicationName NVARCHAR(256),
    @Path NVARCHAR(256) = NULL,
    @UserName NVARCHAR(256) = NULL,
    @InactiveSinceDate DATETIME = NULL)
AS
BEGIN

    DECLARE @ApplicationId UNIQUEIDENTIFIER
    EXEC dbo.aspnet_Personalization_GetApplicationId @ApplicationName, @ApplicationId     ↙
    OUTPUT
    IF (@ApplicationId IS NULL)
        SELECT @Count = 0
    ELSE
        IF (@AllUsersScope = 1)
            SELECT @Count = COUNT(*)
            FROM dbo.aspnet_PersonalizationAllUsers AllUsers, dbo.aspnet_Paths Paths
            WHERE Paths.ApplicationId = @ApplicationId
                  AND AllUsers.PathId = Paths.PathId
                  AND (@Path IS NULL OR Paths.LoweredPath LIKE LOWER(@Path))
        ELSE
            SELECT @Count = COUNT(*)
            FROM dbo.aspnet_PersonalizationPerUser PerUser, dbo.aspnet_Users Users, dbo. ↙
    aspnet_Paths Paths
            WHERE Paths.ApplicationId = @ApplicationId
                  AND PerUser.UserId = Users.UserId
                  AND PerUser.PathId = Paths.PathId
                  AND (@Path IS NULL OR Paths.LoweredPath LIKE LOWER(@Path))
                  AND (@UserName IS NULL OR Users.LoweredUserName LIKE LOWER(@UserName))
                  AND (@InactiveSinceDate IS NULL OR Users.LastActivityDate <= @ ↙
    InactiveSinceDate)
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_PersonalizationAdministration_FindState] ↙
     Script Date: 08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_PersonalizationAdministration_FindState] (
    @AllUsersScope bit,
    @ApplicationName NVARCHAR(256),
    @PageIndex              INT,
    @PageSize               INT,
    @Path NVARCHAR(256) = NULL,
    @UserName NVARCHAR(256) = NULL,
    @InactiveSinceDate DATETIME = NULL)
AS
BEGIN
    DECLARE @ApplicationId UNIQUEIDENTIFIER
    EXEC dbo.aspnet_Personalization_GetApplicationId @ApplicationName, @ApplicationId ↙
    OUTPUT
    IF (@ApplicationId IS NULL)
        RETURN

    -- Set the page bounds
    DECLARE @PageLowerBound INT
    DECLARE @PageUpperBound INT
    DECLARE @TotalRecords   INT
    SET @PageLowerBound = @PageSize * @PageIndex
    SET @PageUpperBound = @PageSize - 1 + @PageLowerBound

    -- Create a temp table to store the selected results
    CREATE TABLE #PageIndex (
        IndexId int IDENTITY (0, 1) NOT NULL,
        ItemId UNIQUEIDENTIFIER
    )

    IF (@AllUsersScope = 1)
    BEGIN
        -- Insert into our temp table
        INSERT INTO #PageIndex (ItemId)
        SELECT Paths.PathId
        FROM dbo.aspnet_Paths Paths,
             ((SELECT Paths.PathId
               FROM dbo.aspnet_PersonalizationAllUsers AllUsers, dbo.aspnet_Paths Paths
               WHERE Paths.ApplicationId = @ApplicationId
                      AND AllUsers.PathId = Paths.PathId
                      AND (@Path IS NULL OR Paths.LoweredPath LIKE LOWER(@Path))
              ) AS SharedDataPerPath
              FULL OUTER JOIN
              (SELECT DISTINCT Paths.PathId
               FROM dbo.aspnet_PersonalizationPerUser PerUser, dbo.aspnet_Paths Paths
               WHERE Paths.ApplicationId = @ApplicationId
                      AND PerUser.PathId = Paths.PathId
                      AND (@Path IS NULL OR Paths.LoweredPath LIKE LOWER(@Path))
              ) AS UserDataPerPath
              ON SharedDataPerPath.PathId = UserDataPerPath.PathId
             )
        WHERE Paths.PathId = SharedDataPerPath.PathId OR Paths.PathId = UserDataPerPath. ↙
    PathId
        ORDER BY Paths.Path ASC

        SELECT @TotalRecords = @@ROWCOUNT

        SELECT Paths.Path,
               SharedDataPerPath.LastUpdatedDate,
               SharedDataPerPath.SharedDataLength,
               UserDataPerPath.UserDataLength,
               UserDataPerPath.UserCount
        FROM dbo.aspnet_Paths Paths,
             ((SELECT PageIndex.ItemId AS PathId,
                      AllUsers.LastUpdatedDate AS LastUpdatedDate,
                      DATALENGTH(AllUsers.PageSettings) AS SharedDataLength
               FROM dbo.aspnet_PersonalizationAllUsers AllUsers, #PageIndex PageIndex
               WHERE AllUsers.PathId = PageIndex.ItemId
                     AND PageIndex.IndexId >= @PageLowerBound AND PageIndex.IndexId <= @ ↙
    PageUpperBound
              ) AS SharedDataPerPath
              FULL OUTER JOIN
              (SELECT PageIndex.ItemId AS PathId,
                      SUM(DATALENGTH(PerUser.PageSettings)) AS UserDataLength,
                      COUNT(*) AS UserCount
               FROM aspnet_PersonalizationPerUser PerUser, #PageIndex PageIndex
               WHERE PerUser.PathId = PageIndex.ItemId
                     AND PageIndex.IndexId >= @PageLowerBound AND PageIndex.IndexId <= @ ↙
    PageUpperBound
               GROUP BY PageIndex.ItemId
              ) AS UserDataPerPath
              ON SharedDataPerPath.PathId = UserDataPerPath.PathId
             )
        WHERE Paths.PathId = SharedDataPerPath.PathId OR Paths.PathId = UserDataPerPath. ↙
    PathId
        ORDER BY Paths.Path ASC
    END
    ELSE
    BEGIN
        -- Insert into our temp table
        INSERT INTO #PageIndex (ItemId)
        SELECT PerUser.Id
        FROM dbo.aspnet_PersonalizationPerUser PerUser, dbo.aspnet_Users Users, dbo. ↙
    aspnet_Paths Paths
        WHERE Paths.ApplicationId = @ApplicationId
              AND PerUser.UserId = Users.UserId
              AND PerUser.PathId = Paths.PathId
              AND (@Path IS NULL OR Paths.LoweredPath LIKE LOWER(@Path))
              AND (@UserName IS NULL OR Users.LoweredUserName LIKE LOWER(@UserName))
              AND (@InactiveSinceDate IS NULL OR Users.LastActivityDate <= @ ↙
    InactiveSinceDate)
        ORDER BY Paths.Path ASC, Users.UserName ASC

        SELECT @TotalRecords = @@ROWCOUNT

        SELECT Paths.Path, PerUser.LastUpdatedDate, DATALENGTH(PerUser.PageSettings), ↙
    Users.UserName, Users.LastActivityDate
        FROM dbo.aspnet_PersonalizationPerUser PerUser, dbo.aspnet_Users Users, dbo. ↙
    aspnet_Paths Paths, #PageIndex PageIndex
        WHERE PerUser.Id = PageIndex.ItemId
              AND PerUser.UserId = Users.UserId
              AND PerUser.PathId = Paths.PathId
              AND PageIndex.IndexId >= @PageLowerBound AND PageIndex.IndexId <= @ ↙
    PageUpperBound
        ORDER BY Paths.Path ASC, Users.UserName ASC
    END

    RETURN @TotalRecords
```

```
END
GO
/****** Object:  StoredProcedure [dbo].                                                        ↙
    [aspnet_PersonalizationAdministration_DeleteAllState]    Script Date: 08/16/2011 01:24↙
    :27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_PersonalizationAdministration_DeleteAllState] (
    @AllUsersScope bit,
    @ApplicationName NVARCHAR(256),
    @Count int OUT)
AS
BEGIN
    DECLARE @ApplicationId UNIQUEIDENTIFIER
    EXEC dbo.aspnet_Personalization_GetApplicationId @ApplicationName, @ApplicationId     ↙
    OUTPUT
    IF (@ApplicationId IS NULL)
        SELECT @Count = 0
    ELSE
    BEGIN
        IF (@AllUsersScope = 1)
            DELETE FROM aspnet_PersonalizationAllUsers
            WHERE PathId IN
               (SELECT Paths.PathId
                FROM dbo.aspnet_Paths Paths
                WHERE Paths.ApplicationId = @ApplicationId)
        ELSE
            DELETE FROM aspnet_PersonalizationPerUser
            WHERE PathId IN
               (SELECT Paths.PathId
                FROM dbo.aspnet_Paths Paths
                WHERE Paths.ApplicationId = @ApplicationId)

        SELECT @Count = @@ROWCOUNT
    END
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_Membership_UpdateUserInfo]    Script Date: ↙
    08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_Membership_UpdateUserInfo]
    @ApplicationName                nvarchar(256),
    @UserName                       nvarchar(256),
    @IsPasswordCorrect              bit,
    @UpdateLastLoginActivityDate    bit,
    @MaxInvalidPasswordAttempts     int,
    @PasswordAttemptWindow          int,
    @CurrentTimeUtc                 datetime,
    @LastLoginDate                  datetime,
    @LastActivityDate               datetime
AS
BEGIN
    DECLARE @UserId                                 uniqueidentifier
    DECLARE @IsApproved                             bit
    DECLARE @IsLockedOut                            bit
    DECLARE @LastLockoutDate                        datetime
    DECLARE @FailedPasswordAttemptCount             int
    DECLARE @FailedPasswordAttemptWindowStart       datetime
    DECLARE @FailedPasswordAnswerAttemptCount       int
    DECLARE @FailedPasswordAnswerAttemptWindowStart datetime

    DECLARE @ErrorCode     int
    SET @ErrorCode = 0

    DECLARE @TranStarted   bit
    SET @TranStarted = 0

    IF( @@TRANCOUNT = 0 )
    BEGIN
        BEGIN TRANSACTION
        SET @TranStarted = 1
    END
    ELSE
        SET @TranStarted = 0

    SELECT  @UserId = u.UserId,
            @IsApproved = m.IsApproved,
            @IsLockedOut = m.IsLockedOut,
            @LastLockoutDate = m.LastLockoutDate,
            @FailedPasswordAttemptCount = m.FailedPasswordAttemptCount,
            @FailedPasswordAttemptWindowStart = m.FailedPasswordAttemptWindowStart,
            @FailedPasswordAnswerAttemptCount = m.FailedPasswordAnswerAttemptCount,
            @FailedPasswordAnswerAttemptWindowStart = m.                            ↙
    FailedPasswordAnswerAttemptWindowStart
    FROM    dbo.aspnet_Applications a, dbo.aspnet_Users u, dbo.aspnet_Membership m WITH ( ↙
    UPDLOCK )
    WHERE   LOWER(@ApplicationName) = a.LoweredApplicationName AND
            u.ApplicationId = a.ApplicationId    AND
            u.UserId = m.UserId AND
            LOWER(@UserName) = u.LoweredUserName

    IF ( @@rowcount = 0 )
    BEGIN
        SET @ErrorCode = 1
        GOTO Cleanup
    END

    IF( @IsLockedOut = 1 )
    BEGIN
        GOTO Cleanup
    END

    IF( @IsPasswordCorrect = 0 )
    BEGIN
        IF( @CurrentTimeUtc > DATEADD( minute, @PasswordAttemptWindow, @             ↙
    FailedPasswordAttemptWindowStart ) )
        BEGIN
            SET @FailedPasswordAttemptWindowStart = @CurrentTimeUtc
            SET @FailedPasswordAttemptCount = 1
        END
        ELSE
        BEGIN
            SET @FailedPasswordAttemptWindowStart = @CurrentTimeUtc
            SET @FailedPasswordAttemptCount = @FailedPasswordAttemptCount + 1
        END

        BEGIN
            IF( @FailedPasswordAttemptCount >= @MaxInvalidPasswordAttempts )
            BEGIN
                SET @IsLockedOut = 1
                SET @LastLockoutDate = @CurrentTimeUtc
            END
        END
    END
    ELSE
    BEGIN
        IF( @FailedPasswordAttemptCount > 0 OR @FailedPasswordAnswerAttemptCount > 0 )
        BEGIN
            SET @FailedPasswordAttemptCount = 0
            SET @FailedPasswordAttemptWindowStart = CONVERT( datetime, '17540101', 112 )
            SET @FailedPasswordAnswerAttemptCount = 0
            SET @FailedPasswordAnswerAttemptWindowStart = CONVERT( datetime, '17540101',  ↙
    112 )
            SET @LastLockoutDate = CONVERT( datetime, '17540101', 112 )
        END
    END

    IF( @UpdateLastLoginActivityDate = 1 )
    BEGIN
        UPDATE  dbo.aspnet_Users
        SET     LastActivityDate = @LastActivityDate
        WHERE   @UserId = UserId

        IF( @@ERROR <> 0 )
        BEGIN
            SET @ErrorCode = -1
            GOTO Cleanup
        END

        UPDATE  dbo.aspnet_Membership
        SET     LastLoginDate = @LastLoginDate
        WHERE   UserId = @UserId

        IF( @@ERROR <> 0 )
        BEGIN
            SET @ErrorCode = -1
            GOTO Cleanup
        END
    END


    UPDATE dbo.aspnet_Membership
    SET IsLockedOut = @IsLockedOut, LastLockoutDate = @LastLockoutDate,
        FailedPasswordAttemptCount = @FailedPasswordAttemptCount,
        FailedPasswordAttemptWindowStart = @FailedPasswordAttemptWindowStart,
        FailedPasswordAnswerAttemptCount = @FailedPasswordAnswerAttemptCount,
        FailedPasswordAnswerAttemptWindowStart = @FailedPasswordAnswerAttemptWindowStart
    WHERE @UserId = UserId

    IF( @@ERROR <> 0 )
    BEGIN
        SET @ErrorCode = -1
        GOTO Cleanup
    END

    IF( @TranStarted = 1 )
    BEGIN
    SET @TranStarted = 0
    COMMIT TRANSACTION
    END

    RETURN @ErrorCode

Cleanup:

    IF( @TranStarted = 1 )
    BEGIN
        SET @TranStarted = 0
        ROLLBACK TRANSACTION
    END

    RETURN @ErrorCode

END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_Membership_UpdateUser]    Script Date: 08/ ↙
```

```
    16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_Membership_UpdateUser]
    @ApplicationName      nvarchar(256),
    @UserName             nvarchar(256),
    @Email                nvarchar(256),
    @Comment              ntext,
    @IsApproved           bit,
    @LastLoginDate        datetime,
    @LastActivityDate     datetime,
    @UniqueEmail          int,
    @CurrentTimeUtc       datetime
AS
BEGIN
    DECLARE @UserId uniqueidentifier
    DECLARE @ApplicationId uniqueidentifier
    SELECT  @UserId = NULL
    SELECT  @UserId = u.UserId, @ApplicationId = a.ApplicationId
    FROM    dbo.aspnet_Users u, dbo.aspnet_Applications a, dbo.aspnet_Membership m
    WHERE   LoweredUserName = LOWER(@UserName) AND
            u.ApplicationId = a.ApplicationId  AND
            LOWER(@ApplicationName) = a.LoweredApplicationName AND
            u.UserId = m.UserId

    IF (@UserId IS NULL)
        RETURN(1)

    IF (@UniqueEmail = 1)
    BEGIN
        IF (EXISTS (SELECT *
                    FROM  dbo.aspnet_Membership WITH (UPDLOCK, HOLDLOCK)
                    WHERE ApplicationId = @ApplicationId  AND @UserId <> UserId AND        ↙
    LoweredEmail = LOWER(@Email)))
        BEGIN
            RETURN(7)
        END
    END

    DECLARE @TranStarted   bit
    SET @TranStarted = 0

    IF( @@TRANCOUNT = 0 )
    BEGIN
        BEGIN TRANSACTION
        SET @TranStarted = 1
    END
    ELSE
    SET @TranStarted = 0

    UPDATE dbo.aspnet_Users WITH (ROWLOCK)
    SET
         LastActivityDate = @LastActivityDate
    WHERE
       @UserId = UserId

    IF( @@ERROR <> 0 )
        GOTO Cleanup

    UPDATE dbo.aspnet_Membership WITH (ROWLOCK)
    SET
         Email            = @Email,
         LoweredEmail     = LOWER(@Email),
         Comment          = @Comment,
         IsApproved       = @IsApproved,
         LastLoginDate    = @LastLoginDate
    WHERE
       @UserId = UserId

    IF( @@ERROR <> 0 )
        GOTO Cleanup

    IF( @TranStarted = 1 )
    BEGIN
    SET @TranStarted = 0
    COMMIT TRANSACTION
    END

    RETURN 0

Cleanup:

    IF( @TranStarted = 1 )
    BEGIN
        SET @TranStarted = 0
        ROLLBACK TRANSACTION
    END

    RETURN -1
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_Membership_UnlockUser]    Script Date: 08/ ↙
    16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_Membership_UnlockUser]
    @ApplicationName                         nvarchar(256),
    @UserName                                nvarchar(256)
AS
BEGIN
    DECLARE @UserId uniqueidentifier
    SELECT  @UserId = NULL
    SELECT  @UserId = u.UserId
    FROM    dbo.aspnet_Users u, dbo.aspnet_Applications a, dbo.aspnet_Membership m
    WHERE   LoweredUserName = LOWER(@UserName) AND
            u.ApplicationId = a.ApplicationId  AND
            LOWER(@ApplicationName) = a.LoweredApplicationName AND
            u.UserId = m.UserId

    IF ( @UserId IS NULL )
        RETURN 1

    UPDATE dbo.aspnet_Membership
    SET IsLockedOut = 0,
        FailedPasswordAttemptCount = 0,
        FailedPasswordAttemptWindowStart = CONVERT( datetime, '17540101', 112 ),
        FailedPasswordAnswerAttemptCount = 0,
        FailedPasswordAnswerAttemptWindowStart = CONVERT( datetime, '17540101', 112 ),
        LastLockoutDate = CONVERT( datetime, '17540101', 112 )
    WHERE @UserId = UserId

    RETURN 0
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_Membership_SetPassword]    Script Date: 08/↙
    16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
```

```
CREATE PROCEDURE [dbo].[aspnet_Membership_SetPassword]
    @ApplicationName  nvarchar(256),
    @UserName         nvarchar(256),
    @NewPassword      nvarchar(128),
    @PasswordSalt     nvarchar(128),
    @CurrentTimeUtc   datetime,
    @PasswordFormat   int = 0
AS
BEGIN
    DECLARE @UserId uniqueidentifier
    SELECT  @UserId = NULL
    SELECT  @UserId = u.UserId
    FROM    dbo.aspnet_Users u, dbo.aspnet_Applications a, dbo.aspnet_Membership m
    WHERE   LoweredUserName = LOWER(@UserName) AND
            u.ApplicationId = a.ApplicationId  AND
            LOWER(@ApplicationName) = a.LoweredApplicationName AND
            u.UserId = m.UserId

    IF (@UserId IS NULL)
        RETURN(1)

    UPDATE dbo.aspnet_Membership
    SET Password = @NewPassword, PasswordFormat = @PasswordFormat, PasswordSalt = @
    PasswordSalt,
        LastPasswordChangedDate = @CurrentTimeUtc
    WHERE @UserId = UserId
    RETURN(0)
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_Membership_ResetPassword]    Script Date: 
    08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_Membership_ResetPassword]
    @ApplicationName             nvarchar(256),
    @UserName                    nvarchar(256),
    @NewPassword                 nvarchar(128),
    @MaxInvalidPasswordAttempts  int,
    @PasswordAttemptWindow       int,
    @PasswordSalt                nvarchar(128),
    @CurrentTimeUtc              datetime,
    @PasswordFormat              int = 0,
    @PasswordAnswer              nvarchar(128) = NULL
AS
BEGIN
    DECLARE @IsLockedOut                            bit
    DECLARE @LastLockoutDate                        datetime
    DECLARE @FailedPasswordAttemptCount             int
    DECLARE @FailedPasswordAttemptWindowStart       datetime
    DECLARE @FailedPasswordAnswerAttemptCount       int
    DECLARE @FailedPasswordAnswerAttemptWindowStart datetime

    DECLARE @UserId                                 uniqueidentifier
    SET     @UserId = NULL

    DECLARE @ErrorCode     int
    SET @ErrorCode = 0

    DECLARE @TranStarted   bit
    SET @TranStarted = 0

    IF( @@TRANCOUNT = 0 )
    BEGIN
        BEGIN TRANSACTION
        SET @TranStarted = 1
    END
    ELSE
        SET @TranStarted = 0

    SELECT  @UserId = u.UserId
    FROM    dbo.aspnet_Users u, dbo.aspnet_Applications a, dbo.aspnet_Membership m
    WHERE   LoweredUserName = LOWER(@UserName) AND
            u.ApplicationId = a.ApplicationId  AND
            LOWER(@ApplicationName) = a.LoweredApplicationName AND
            u.UserId = m.UserId

    IF ( @UserId IS NULL )
    BEGIN
        SET @ErrorCode = 1
        GOTO Cleanup
    END

    SELECT @IsLockedOut = IsLockedOut,
           @LastLockoutDate = LastLockoutDate,
           @FailedPasswordAttemptCount = FailedPasswordAttemptCount,
           @FailedPasswordAttemptWindowStart = FailedPasswordAttemptWindowStart,
           @FailedPasswordAnswerAttemptCount = FailedPasswordAnswerAttemptCount,
           @FailedPasswordAnswerAttemptWindowStart =                                    ↙
    FailedPasswordAnswerAttemptWindowStart
    FROM dbo.aspnet_Membership WITH ( UPDLOCK )
    WHERE @UserId = UserId

    IF( @IsLockedOut = 1 )
    BEGIN
        SET @ErrorCode = 99
        GOTO Cleanup
    END

    UPDATE dbo.aspnet_Membership
    SET    Password = @NewPassword,
           LastPasswordChangedDate = @CurrentTimeUtc,
           PasswordFormat = @PasswordFormat,
           PasswordSalt = @PasswordSalt
    WHERE  @UserId = UserId AND
           ( ( @PasswordAnswer IS NULL ) OR ( LOWER( PasswordAnswer ) = LOWER( @        ↙
    PasswordAnswer ) ) )

    IF ( @@ROWCOUNT = 0 )
        BEGIN
            IF( @CurrentTimeUtc > DATEADD( minute, @PasswordAttemptWindow, @            ↙
    FailedPasswordAnswerAttemptWindowStart ) )
            BEGIN
                SET @FailedPasswordAnswerAttemptWindowStart = @CurrentTimeUtc
                SET @FailedPasswordAnswerAttemptCount = 1
            END
            ELSE
            BEGIN
                SET @FailedPasswordAnswerAttemptWindowStart = @CurrentTimeUtc
                SET @FailedPasswordAnswerAttemptCount = @FailedPasswordAnswerAttemptCount ↙
    + 1
            END

            BEGIN
                IF( @FailedPasswordAnswerAttemptCount >= @MaxInvalidPasswordAttempts )
                BEGIN
                    SET @IsLockedOut = 1
                    SET @LastLockoutDate = @CurrentTimeUtc
                END
            END

            SET @ErrorCode = 3
        END
    ELSE
        BEGIN
            IF( @FailedPasswordAnswerAttemptCount > 0 )
            BEGIN
                SET @FailedPasswordAnswerAttemptCount = 0
                SET @FailedPasswordAnswerAttemptWindowStart = CONVERT( datetime, '17540101↙
    ', 112 )
            END
        END

    IF( NOT ( @PasswordAnswer IS NULL ) )
    BEGIN
        UPDATE dbo.aspnet_Membership
        SET IsLockedOut = @IsLockedOut, LastLockoutDate = @LastLockoutDate,
            FailedPasswordAttemptCount = @FailedPasswordAttemptCount,
            FailedPasswordAttemptWindowStart = @FailedPasswordAttemptWindowStart,
            FailedPasswordAnswerAttemptCount = @FailedPasswordAnswerAttemptCount,
            FailedPasswordAnswerAttemptWindowStart = @                                    ↙
    FailedPasswordAnswerAttemptWindowStart
        WHERE @UserId = UserId

        IF( @@ERROR <> 0 )
        BEGIN
            SET @ErrorCode = -1
            GOTO Cleanup
        END
    END

    IF( @TranStarted = 1 )
    BEGIN
    SET @TranStarted = 0
    COMMIT TRANSACTION
    END

    RETURN @ErrorCode

Cleanup:

    IF( @TranStarted = 1 )
    BEGIN
        SET @TranStarted = 0
        ROLLBACK TRANSACTION
    END

    RETURN @ErrorCode

END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_Membership_GetUserByUserId]    Script Date:↙
     08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_Membership_GetUserByUserId]
    @UserId               uniqueidentifier,
    @CurrentTimeUtc       datetime,
    @UpdateLastActivity   bit = 0
AS
BEGIN
    IF ( @UpdateLastActivity = 1 )
    BEGIN
        UPDATE   dbo.aspnet_Users
        SET      LastActivityDate = @CurrentTimeUtc
        FROM     dbo.aspnet_Users
        WHERE    @UserId = UserId
        IF ( @@ROWCOUNT = 0 ) -- User ID not found
            RETURN -1
    END

    SELECT  m.Email, m.PasswordQuestion, m.Comment, m.IsApproved,
            m.CreateDate, m.LastLoginDate, u.LastActivityDate,
            m.LastPasswordChangedDate, u.UserName, m.IsLockedOut,
            m.LastLockoutDate
    FROM    dbo.aspnet_Users u, dbo.aspnet_Membership m
    WHERE   @UserId = u.UserId AND u.UserId = m.UserId

    IF ( @@ROWCOUNT = 0 ) -- User ID not found
       RETURN -1

    RETURN 0
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_Membership_GetUserByName]    Script Date: ↙
    08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_Membership_GetUserByName]
    @ApplicationName      nvarchar(256),
    @UserName             nvarchar(256),
    @CurrentTimeUtc       datetime,
    @UpdateLastActivity   bit = 0
AS
BEGIN
    DECLARE @UserId uniqueidentifier

    IF (@UpdateLastActivity = 1)
    BEGIN
        -- select user ID from aspnet_users table
        SELECT TOP 1 @UserId = u.UserId
        FROM    dbo.aspnet_Applications a, dbo.aspnet_Users u, dbo.aspnet_Membership m
        WHERE    LOWER(@ApplicationName) = a.LoweredApplicationName AND
                u.ApplicationId = a.ApplicationId    AND
                LOWER(@UserName) = u.LoweredUserName AND u.UserId = m.UserId

        IF (@@ROWCOUNT = 0) -- Username not found
            RETURN -1

        UPDATE   dbo.aspnet_Users
        SET      LastActivityDate = @CurrentTimeUtc
        WHERE    @UserId = UserId

        SELECT m.Email, m.PasswordQuestion, m.Comment, m.IsApproved,
                m.CreateDate, m.LastLoginDate, u.LastActivityDate, m.                   ↙
    LastPasswordChangedDate,
                u.UserId, m.IsLockedOut, m.LastLockoutDate
        FROM    dbo.aspnet_Applications a, dbo.aspnet_Users u, dbo.aspnet_Membership m
        WHERE  @UserId = u.UserId AND u.UserId = m.UserId
    END
    ELSE
    BEGIN
        SELECT TOP 1 m.Email, m.PasswordQuestion, m.Comment, m.IsApproved,
                m.CreateDate, m.LastLoginDate, u.LastActivityDate, m.                   ↙
    LastPasswordChangedDate,
                u.UserId, m.IsLockedOut,m.LastLockoutDate
        FROM    dbo.aspnet_Applications a, dbo.aspnet_Users u, dbo.aspnet_Membership m
        WHERE    LOWER(@ApplicationName) = a.LoweredApplicationName AND
                u.ApplicationId = a.ApplicationId    AND
                LOWER(@UserName) = u.LoweredUserName AND u.UserId = m.UserId

        IF (@@ROWCOUNT = 0) -- Username not found
```

```
            RETURN -1
    END

    RETURN 0
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_Membership_GetUserByEmail]    Script Date: ↙
    08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_Membership_GetUserByEmail]
    @ApplicationName  nvarchar(256),
    @Email            nvarchar(256)
AS
BEGIN
    IF( @Email IS NULL )
        SELECT  u.UserName
        FROM    dbo.aspnet_Applications a, dbo.aspnet_Users u, dbo.aspnet_Membership m
        WHERE   LOWER(@ApplicationName) = a.LoweredApplicationName AND
                u.ApplicationId = a.ApplicationId    AND
                u.UserId = m.UserId AND
                m.LoweredEmail IS NULL
    ELSE
        SELECT  u.UserName
        FROM    dbo.aspnet_Applications a, dbo.aspnet_Users u, dbo.aspnet_Membership m
        WHERE   LOWER(@ApplicationName) = a.LoweredApplicationName AND
                u.ApplicationId = a.ApplicationId    AND
                u.UserId = m.UserId AND
                LOWER(@Email) = m.LoweredEmail

    IF (@@rowcount = 0)
        RETURN(1)
    RETURN(0)
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_Membership_GetPasswordWithFormat]    Script↙
     Date: 08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_Membership_GetPasswordWithFormat]
    @ApplicationName                nvarchar(256),
    @UserName                       nvarchar(256),
    @UpdateLastLoginActivityDate    bit,
    @CurrentTimeUtc                 datetime
AS
BEGIN
    DECLARE @IsLockedOut                        bit
    DECLARE @UserId                             uniqueidentifier
    DECLARE @Password                           nvarchar(128)
    DECLARE @PasswordSalt                       nvarchar(128)
    DECLARE @PasswordFormat                     int
    DECLARE @FailedPasswordAttemptCount         int
    DECLARE @FailedPasswordAnswerAttemptCount   int
    DECLARE @IsApproved                         bit
    DECLARE @LastActivityDate                   datetime
    DECLARE @LastLoginDate                      datetime

    SELECT  @UserId          = NULL

    SELECT  @UserId = u.UserId, @IsLockedOut = m.IsLockedOut, @Password=Password, @       ↙
    PasswordFormat=PasswordFormat,
            @PasswordSalt=PasswordSalt, @FailedPasswordAttemptCount=                      ↙
    FailedPasswordAttemptCount,
            @FailedPasswordAnswerAttemptCount=FailedPasswordAnswerAttemptCount, @ ↙
    IsApproved=IsApproved,
            @LastActivityDate = LastActivityDate, @LastLoginDate = LastLoginDate
    FROM    dbo.aspnet_Applications a, dbo.aspnet_Users u, dbo.aspnet_Membership m
    WHERE   LOWER(@ApplicationName) = a.LoweredApplicationName AND
            u.ApplicationId = a.ApplicationId    AND
            u.UserId = m.UserId AND
            LOWER(@UserName) = u.LoweredUserName

    IF (@UserId IS NULL)
        RETURN 1

    IF (@IsLockedOut = 1)
        RETURN 99

    SELECT   @Password, @PasswordFormat, @PasswordSalt, @FailedPasswordAttemptCount,
             @FailedPasswordAnswerAttemptCount, @IsApproved, @LastLoginDate, @ ↙
    LastActivityDate

    IF (@UpdateLastLoginActivityDate = 1 AND @IsApproved = 1)
    BEGIN
        UPDATE  dbo.aspnet_Membership
        SET     LastLoginDate = @CurrentTimeUtc
        WHERE   UserId = @UserId

        UPDATE  dbo.aspnet_Users
        SET     LastActivityDate = @CurrentTimeUtc
        WHERE   @UserId = UserId
    END


    RETURN 0
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_Membership_GetPassword]    Script Date: 08/↙
    16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_Membership_GetPassword]
    @ApplicationName                nvarchar(256),
    @UserName                       nvarchar(256),
    @MaxInvalidPasswordAttempts     int,
    @PasswordAttemptWindow          int,
    @CurrentTimeUtc                 datetime,
    @PasswordAnswer                 nvarchar(128) = NULL
AS
BEGIN
    DECLARE @UserId                                 uniqueidentifier
    DECLARE @PasswordFormat                         int
    DECLARE @Password                               nvarchar(128)
    DECLARE @passAns                                nvarchar(128)
    DECLARE @IsLockedOut                            bit
    DECLARE @LastLockoutDate                        datetime
    DECLARE @FailedPasswordAttemptCount             int
    DECLARE @FailedPasswordAttemptWindowStart       datetime
    DECLARE @FailedPasswordAnswerAttemptCount       int
    DECLARE @FailedPasswordAnswerAttemptWindowStart datetime

    DECLARE @ErrorCode     int
    SET @ErrorCode = 0

    DECLARE @TranStarted   bit
    SET @TranStarted = 0

    IF( @@TRANCOUNT = 0 )
```

```
BEGIN
    BEGIN TRANSACTION
    SET @TranStarted = 1
END
ELSE
    SET @TranStarted = 0

SELECT  @UserId = u.UserId,
        @Password = m.Password,
        @passAns = m.PasswordAnswer,
        @PasswordFormat = m.PasswordFormat,
        @IsLockedOut = m.IsLockedOut,
        @LastLockoutDate = m.LastLockoutDate,
        @FailedPasswordAttemptCount = m.FailedPasswordAttemptCount,
        @FailedPasswordAttemptWindowStart = m.FailedPasswordAttemptWindowStart,
        @FailedPasswordAnswerAttemptCount = m.FailedPasswordAnswerAttemptCount,
        @FailedPasswordAnswerAttemptWindowStart = m.                         ↙
FailedPasswordAnswerAttemptWindowStart
FROM    dbo.aspnet_Applications a, dbo.aspnet_Users u, dbo.aspnet_Membership m WITH ( ↙
UPDLOCK )
WHERE   LOWER(@ApplicationName) = a.LoweredApplicationName AND
        u.ApplicationId = a.ApplicationId    AND
        u.UserId = m.UserId AND
        LOWER(@UserName) = u.LoweredUserName

IF ( @@rowcount = 0 )
BEGIN
    SET @ErrorCode = 1
    GOTO Cleanup
END

IF( @IsLockedOut = 1 )
BEGIN
    SET @ErrorCode = 99
    GOTO Cleanup
END

IF ( NOT( @PasswordAnswer IS NULL ) )
BEGIN
    IF( ( @passAns IS NULL ) OR ( LOWER( @passAns ) <> LOWER( @PasswordAnswer ) ) )
    BEGIN
        IF( @CurrentTimeUtc > DATEADD( minute, @PasswordAttemptWindow, @          ↙
FailedPasswordAnswerAttemptWindowStart ) )
        BEGIN
            SET @FailedPasswordAnswerAttemptWindowStart = @CurrentTimeUtc
            SET @FailedPasswordAnswerAttemptCount = 1
        END
        ELSE
        BEGIN
            SET @FailedPasswordAnswerAttemptCount = @FailedPasswordAnswerAttemptCount ↙
+ 1
            SET @FailedPasswordAnswerAttemptWindowStart = @CurrentTimeUtc
        END

        BEGIN
            IF( @FailedPasswordAnswerAttemptCount >= @MaxInvalidPasswordAttempts )
            BEGIN
                SET @IsLockedOut = 1
                SET @LastLockoutDate = @CurrentTimeUtc
            END
        END

        SET @ErrorCode = 3
    END
    ELSE
    BEGIN
        IF( @FailedPasswordAnswerAttemptCount > 0 )
```

```
            BEGIN
                SET @FailedPasswordAnswerAttemptCount = 0
                SET @FailedPasswordAnswerAttemptWindowStart = CONVERT( datetime, '17540101↙
    ', 112 )
            END
        END

        UPDATE dbo.aspnet_Membership
        SET IsLockedOut = @IsLockedOut, LastLockoutDate = @LastLockoutDate,
            FailedPasswordAttemptCount = @FailedPasswordAttemptCount,
            FailedPasswordAttemptWindowStart = @FailedPasswordAttemptWindowStart,
            FailedPasswordAnswerAttemptCount = @FailedPasswordAnswerAttemptCount,
            FailedPasswordAnswerAttemptWindowStart = @                                ↙
    FailedPasswordAnswerAttemptWindowStart
        WHERE @UserId = UserId

        IF( @@ERROR <> 0 )
        BEGIN
            SET @ErrorCode = -1
            GOTO Cleanup
        END
    END

    IF( @TranStarted = 1 )
    BEGIN
    SET @TranStarted = 0
    COMMIT TRANSACTION
    END

    IF( @ErrorCode = 0 )
        SELECT @Password, @PasswordFormat

    RETURN @ErrorCode

Cleanup:

    IF( @TranStarted = 1 )
    BEGIN
        SET @TranStarted = 0
        ROLLBACK TRANSACTION
    END

    RETURN @ErrorCode

END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_Membership_GetNumberOfUsersOnline]        ↙
    Script Date: 08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_Membership_GetNumberOfUsersOnline]
    @ApplicationName            nvarchar(256),
    @MinutesSinceLastInActive   int,
    @CurrentTimeUtc             datetime
AS
BEGIN
    DECLARE @DateActive datetime
    SELECT  @DateActive = DATEADD(minute,  -(@MinutesSinceLastInActive), @CurrentTimeUtc)

    DECLARE @NumOnline int
    SELECT  @NumOnline = COUNT(*)
    FROM    dbo.aspnet_Users u(NOLOCK),
            dbo.aspnet_Applications a(NOLOCK),
            dbo.aspnet_Membership m(NOLOCK)
    WHERE   u.ApplicationId = a.ApplicationId                  AND
```

```
            LastActivityDate > @DateActive                     AND
            a.LoweredApplicationName = LOWER(@ApplicationName) AND
            u.UserId = m.UserId
    RETURN(@NumOnline)
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_Membership_GetAllUsers]    Script Date: 08/↙
    16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_Membership_GetAllUsers]
    @ApplicationName       nvarchar(256),
    @PageIndex             int,
    @PageSize              int
AS
BEGIN
    DECLARE @ApplicationId uniqueidentifier
    SELECT  @ApplicationId = NULL
    SELECT  @ApplicationId = ApplicationId FROM dbo.aspnet_Applications WHERE LOWER(@     ↙
    ApplicationName) = LoweredApplicationName
    IF (@ApplicationId IS NULL)
        RETURN 0


    -- Set the page bounds
    DECLARE @PageLowerBound int
    DECLARE @PageUpperBound int
    DECLARE @TotalRecords   int
    SET @PageLowerBound = @PageSize * @PageIndex
    SET @PageUpperBound = @PageSize - 1 + @PageLowerBound

    -- Create a temp table TO store the select results
    CREATE TABLE #PageIndexForUsers
    (
        IndexId int IDENTITY (0, 1) NOT NULL,
        UserId uniqueidentifier
    )

    -- Insert into our temp table
    INSERT INTO #PageIndexForUsers (UserId)
    SELECT u.UserId
    FROM   dbo.aspnet_Membership m, dbo.aspnet_Users u
    WHERE  u.ApplicationId = @ApplicationId AND u.UserId = m.UserId
    ORDER BY u.UserName

    SELECT @TotalRecords = @@ROWCOUNT

    SELECT u.UserName, m.Email, m.PasswordQuestion, m.Comment, m.IsApproved,
            m.CreateDate,
            m.LastLoginDate,
            u.LastActivityDate,
            m.LastPasswordChangedDate,
            u.UserId, m.IsLockedOut,
            m.LastLockoutDate
    FROM   dbo.aspnet_Membership m, dbo.aspnet_Users u, #PageIndexForUsers p
    WHERE  u.UserId = p.UserId AND u.UserId = m.UserId AND
           p.IndexId >= @PageLowerBound AND p.IndexId <= @PageUpperBound
    ORDER BY u.UserName
    RETURN @TotalRecords
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_Membership_FindUsersByName]    Script Date:↙
     08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
```

```
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_Membership_FindUsersByName]
    @ApplicationName       nvarchar(256),
    @UserNameToMatch       nvarchar(256),
    @PageIndex             int,
    @PageSize              int
AS
BEGIN
    DECLARE @ApplicationId uniqueidentifier
    SELECT  @ApplicationId = NULL
    SELECT  @ApplicationId = ApplicationId FROM dbo.aspnet_Applications WHERE LOWER(@     ↙
    ApplicationName) = LoweredApplicationName
    IF (@ApplicationId IS NULL)
        RETURN 0

    -- Set the page bounds
    DECLARE @PageLowerBound int
    DECLARE @PageUpperBound int
    DECLARE @TotalRecords   int
    SET @PageLowerBound = @PageSize * @PageIndex
    SET @PageUpperBound = @PageSize - 1 + @PageLowerBound

    -- Create a temp table TO store the select results
    CREATE TABLE #PageIndexForUsers
    (
        IndexId int IDENTITY (0, 1) NOT NULL,
        UserId uniqueidentifier
    )

    -- Insert into our temp table
    INSERT INTO #PageIndexForUsers (UserId)
        SELECT u.UserId
        FROM   dbo.aspnet_Users u, dbo.aspnet_Membership m
        WHERE  u.ApplicationId = @ApplicationId AND m.UserId = u.UserId AND u.            ↙
    LoweredUserName LIKE LOWER(@UserNameToMatch)
        ORDER BY u.UserName


    SELECT  u.UserName, m.Email, m.PasswordQuestion, m.Comment, m.IsApproved,
            m.CreateDate,
            m.LastLoginDate,
            u.LastActivityDate,
            m.LastPasswordChangedDate,
            u.UserId, m.IsLockedOut,
            m.LastLockoutDate
    FROM   dbo.aspnet_Membership m, dbo.aspnet_Users u, #PageIndexForUsers p
    WHERE  u.UserId = p.UserId AND u.UserId = m.UserId AND
           p.IndexId >= @PageLowerBound AND p.IndexId <= @PageUpperBound
    ORDER BY u.UserName

    SELECT  @TotalRecords = COUNT(*)
    FROM    #PageIndexForUsers
    RETURN @TotalRecords
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_Membership_FindUsersByEmail]    Script Date↙
    : 08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_Membership_FindUsersByEmail]
    @ApplicationName       nvarchar(256),
    @EmailToMatch          nvarchar(256),
    @PageIndex             int,
    @PageSize              int
```

```
AS
BEGIN
    DECLARE @ApplicationId uniqueidentifier
    SELECT  @ApplicationId = NULL
    SELECT  @ApplicationId = ApplicationId FROM dbo.aspnet_Applications WHERE LOWER(@ ↙
    ApplicationName) = LoweredApplicationName
    IF (@ApplicationId IS NULL)
        RETURN 0

    -- Set the page bounds
    DECLARE @PageLowerBound int
    DECLARE @PageUpperBound int
    DECLARE @TotalRecords   int
    SET @PageLowerBound = @PageSize * @PageIndex
    SET @PageUpperBound = @PageSize - 1 + @PageLowerBound

    -- Create a temp table TO store the select results
    CREATE TABLE #PageIndexForUsers
    (
        IndexId int IDENTITY (0, 1) NOT NULL,
        UserId uniqueidentifier
    )

    -- Insert into our temp table
    IF( @EmailToMatch IS NULL )
        INSERT INTO #PageIndexForUsers (UserId)
            SELECT u.UserId
            FROM   dbo.aspnet_Users u, dbo.aspnet_Membership m
            WHERE  u.ApplicationId = @ApplicationId AND m.UserId = u.UserId AND m.Email IS↙
     NULL
            ORDER BY m.LoweredEmail
    ELSE
        INSERT INTO #PageIndexForUsers (UserId)
            SELECT u.UserId
            FROM   dbo.aspnet_Users u, dbo.aspnet_Membership m
            WHERE  u.ApplicationId = @ApplicationId AND m.UserId = u.UserId AND m.      ↙
    LoweredEmail LIKE LOWER(@EmailToMatch)
            ORDER BY m.LoweredEmail

    SELECT  u.UserName, m.Email, m.PasswordQuestion, m.Comment, m.IsApproved,
            m.CreateDate,
            m.LastLoginDate,
            u.LastActivityDate,
            m.LastPasswordChangedDate,
            u.UserId, m.IsLockedOut,
            m.LastLockoutDate
    FROM   dbo.aspnet_Membership m, dbo.aspnet_Users u, #PageIndexForUsers p
    WHERE  u.UserId = p.UserId AND u.UserId = m.UserId AND
           p.IndexId >= @PageLowerBound AND p.IndexId <= @PageUpperBound
    ORDER BY m.LoweredEmail

    SELECT  @TotalRecords = COUNT(*)
    FROM    #PageIndexForUsers
    RETURN @TotalRecords
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_Membership_CreateUser]    Script Date: 08/ ↙
    16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_Membership_CreateUser]
    @ApplicationName                        nvarchar(256),
    @UserName                               nvarchar(256),
    @Password                               nvarchar(128),
    @PasswordSalt                           nvarchar(128),
```

```sql
    @Email                                  nvarchar(256),
    @PasswordQuestion                       nvarchar(256),
    @PasswordAnswer                         nvarchar(128),
    @IsApproved                             bit,
    @CurrentTimeUtc                         datetime,
    @CreateDate                             datetime = NULL,
    @UniqueEmail                            int      = 0,
    @PasswordFormat                         int      = 0,
    @UserId                                 uniqueidentifier OUTPUT
AS
BEGIN
    DECLARE @ApplicationId uniqueidentifier
    SELECT  @ApplicationId = NULL

    DECLARE @NewUserId uniqueidentifier
    SELECT @NewUserId = NULL

    DECLARE @IsLockedOut bit
    SET @IsLockedOut = 0

    DECLARE @LastLockoutDate  datetime
    SET @LastLockoutDate = CONVERT( datetime, '17540101', 112 )

    DECLARE @FailedPasswordAttemptCount int
    SET @FailedPasswordAttemptCount = 0

    DECLARE @FailedPasswordAttemptWindowStart  datetime
    SET @FailedPasswordAttemptWindowStart = CONVERT( datetime, '17540101', 112 )

    DECLARE @FailedPasswordAnswerAttemptCount int
    SET @FailedPasswordAnswerAttemptCount = 0

    DECLARE @FailedPasswordAnswerAttemptWindowStart  datetime
    SET @FailedPasswordAnswerAttemptWindowStart = CONVERT( datetime, '17540101', 112 )

    DECLARE @NewUserCreated bit
    DECLARE @ReturnValue   int
    SET @ReturnValue = 0

    DECLARE @ErrorCode     int
    SET @ErrorCode = 0

    DECLARE @TranStarted   bit
    SET @TranStarted = 0

    IF( @@TRANCOUNT = 0 )
    BEGIN
        BEGIN TRANSACTION
        SET @TranStarted = 1
    END
    ELSE
        SET @TranStarted = 0

    EXEC dbo.aspnet_Applications_CreateApplication @ApplicationName, @ApplicationId OUTPUT

    IF( @@ERROR <> 0 )
    BEGIN
        SET @ErrorCode = -1
        GOTO Cleanup
    END

    SET @CreateDate = @CurrentTimeUtc

    SELECT  @NewUserId = UserId FROM dbo.aspnet_Users WHERE LOWER(@UserName) = 
    LoweredUserName AND @ApplicationId = ApplicationId
    IF ( @NewUserId IS NULL )
    BEGIN
```

```
        SET @NewUserId = @UserId
        EXEC @ReturnValue = dbo.aspnet_Users_CreateUser @ApplicationId, @UserName, 0, @   ↙
    CreateDate, @NewUserId OUTPUT
        SET @NewUserCreated = 1
    END
    ELSE
    BEGIN
        SET @NewUserCreated = 0
        IF( @NewUserId <> @UserId AND @UserId IS NOT NULL )
        BEGIN
            SET @ErrorCode = 6
            GOTO Cleanup
        END
    END

    IF( @@ERROR <> 0 )
    BEGIN
        SET @ErrorCode = -1
        GOTO Cleanup
    END

    IF( @ReturnValue = -1 )
    BEGIN
        SET @ErrorCode = 10
        GOTO Cleanup
    END

    IF ( EXISTS ( SELECT UserId
                  FROM   dbo.aspnet_Membership
                  WHERE  @NewUserId = UserId ) )
    BEGIN
        SET @ErrorCode = 6
        GOTO Cleanup
    END

    SET @UserId = @NewUserId

    IF (@UniqueEmail = 1)
    BEGIN
        IF (EXISTS (SELECT *
                    FROM  dbo.aspnet_Membership m WITH ( UPDLOCK, HOLDLOCK )
                    WHERE ApplicationId = @ApplicationId AND LoweredEmail = LOWER(@      ↙
    Email)))
        BEGIN
            SET @ErrorCode = 7
            GOTO Cleanup
        END
    END

    IF (@NewUserCreated = 0)
    BEGIN
        UPDATE dbo.aspnet_Users
        SET    LastActivityDate = @CreateDate
        WHERE  @UserId = UserId
        IF( @@ERROR <> 0 )
        BEGIN
            SET @ErrorCode = -1
            GOTO Cleanup
        END
    END

    INSERT INTO dbo.aspnet_Membership
                ( ApplicationId,
                  UserId,
                  Password,
                  PasswordSalt,
                  Email,
                  LoweredEmail,
                  PasswordQuestion,
                  PasswordAnswer,
                  PasswordFormat,
                  IsApproved,
                  IsLockedOut,
                  CreateDate,
                  LastLoginDate,
                  LastPasswordChangedDate,
                  LastLockoutDate,
                  FailedPasswordAttemptCount,
                  FailedPasswordAttemptWindowStart,
                  FailedPasswordAnswerAttemptCount,
                  FailedPasswordAnswerAttemptWindowStart )
         VALUES ( @ApplicationId,
                  @UserId,
                  @Password,
                  @PasswordSalt,
                  @Email,
                  LOWER(@Email),
                  @PasswordQuestion,
                  @PasswordAnswer,
                  @PasswordFormat,
                  @IsApproved,
                  @IsLockedOut,
                  @CreateDate,
                  @CreateDate,
                  @CreateDate,
                  @LastLockoutDate,
                  @FailedPasswordAttemptCount,
                  @FailedPasswordAttemptWindowStart,
                  @FailedPasswordAnswerAttemptCount,
                  @FailedPasswordAnswerAttemptWindowStart )

    IF( @@ERROR <> 0 )
    BEGIN
        SET @ErrorCode = -1
        GOTO Cleanup
    END

    IF( @TranStarted = 1 )
    BEGIN
        SET @TranStarted = 0
        COMMIT TRANSACTION
    END

    RETURN 0

Cleanup:

    IF( @TranStarted = 1 )
    BEGIN
        SET @TranStarted = 0
        ROLLBACK TRANSACTION
    END

    RETURN @ErrorCode

END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_Membership_ChangePasswordQuestionAndAnswer]↙
        Script Date: 08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_Membership_ChangePasswordQuestionAndAnswer]
```

```
    @ApplicationName       nvarchar(256),
    @UserName              nvarchar(256),
    @NewPasswordQuestion   nvarchar(256),
    @NewPasswordAnswer     nvarchar(128)
AS
BEGIN
    DECLARE @UserId uniqueidentifier
    SELECT  @UserId = NULL
    SELECT  @UserId = u.UserId
    FROM    dbo.aspnet_Membership m, dbo.aspnet_Users u, dbo.aspnet_Applications a
    WHERE   LoweredUserName = LOWER(@UserName) AND
            u.ApplicationId = a.ApplicationId  AND
            LOWER(@ApplicationName) = a.LoweredApplicationName AND
            u.UserId = m.UserId
    IF (@UserId IS NULL)
    BEGIN
        RETURN(1)
    END

    UPDATE dbo.aspnet_Membership
    SET    PasswordQuestion = @NewPasswordQuestion, PasswordAnswer = @NewPasswordAnswer
    WHERE  UserId=@UserId
    RETURN(0)
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_AnyDataInTables]    Script Date: 08/16/2011↙
     01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_AnyDataInTables]
    @TablesToCheck int
AS
BEGIN
    -- Check Membership table if (@TablesToCheck & 1) is set
    IF ((@TablesToCheck & 1) <> 0 AND
        (EXISTS (SELECT name FROM sysobjects WHERE (name = N'vw_aspnet_MembershipUsers')   ↙
    AND (type = 'V'))))
    BEGIN
        IF (EXISTS(SELECT TOP 1 UserId FROM dbo.aspnet_Membership))
        BEGIN
            SELECT N'aspnet_Membership'
            RETURN
        END
    END

    -- Check aspnet_Roles table if (@TablesToCheck & 2) is set
    IF ((@TablesToCheck & 2) <> 0  AND
        (EXISTS (SELECT name FROM sysobjects WHERE (name = N'vw_aspnet_Roles') AND (type =↙
     'V'))) )
    BEGIN
        IF (EXISTS(SELECT TOP 1 RoleId FROM dbo.aspnet_Roles))
        BEGIN
            SELECT N'aspnet_Roles'
            RETURN
        END
    END

    -- Check aspnet_Profile table if (@TablesToCheck & 4) is set
    IF ((@TablesToCheck & 4) <> 0  AND
        (EXISTS (SELECT name FROM sysobjects WHERE (name = N'vw_aspnet_Profiles') AND      ↙
    (type = 'V'))) )
    BEGIN
        IF (EXISTS(SELECT TOP 1 UserId FROM dbo.aspnet_Profile))
        BEGIN
            SELECT N'aspnet_Profile'
            RETURN
        END
    END

    -- Check aspnet_PersonalizationPerUser table if (@TablesToCheck & 8) is set
    IF ((@TablesToCheck & 8) <> 0  AND
        (EXISTS (SELECT name FROM sysobjects WHERE (name = N'vw_aspnet_WebPartState_User')↙
     AND (type = 'V'))) )
    BEGIN
        IF (EXISTS(SELECT TOP 1 UserId FROM dbo.aspnet_PersonalizationPerUser))
        BEGIN
            SELECT N'aspnet_PersonalizationPerUser'
            RETURN
        END
    END

    -- Check aspnet_PersonalizationPerUser table if (@TablesToCheck & 16) is set
    IF ((@TablesToCheck & 16) <> 0  AND
        (EXISTS (SELECT name FROM sysobjects WHERE (name = N'aspnet_WebEvent_LogEvent')   ↙
    AND (type = 'P'))) )
    BEGIN
        IF (EXISTS(SELECT TOP 1 * FROM dbo.aspnet_WebEvent_Events))
        BEGIN
            SELECT N'aspnet_WebEvent_Events'
            RETURN
        END
    END

    -- Check aspnet_Users table if (@TablesToCheck & 1,2,4 & 8) are all set
    IF ((@TablesToCheck & 1) <> 0 AND
        (@TablesToCheck & 2) <> 0 AND
        (@TablesToCheck & 4) <> 0 AND
        (@TablesToCheck & 8) <> 0 AND
        (@TablesToCheck & 32) <> 0 AND
        (@TablesToCheck & 128) <> 0 AND
        (@TablesToCheck & 256) <> 0 AND
        (@TablesToCheck & 512) <> 0 AND
        (@TablesToCheck & 1024) <> 0)
    BEGIN
        IF (EXISTS(SELECT TOP 1 UserId FROM dbo.aspnet_Users))
        BEGIN
            SELECT N'aspnet_Users'
            RETURN
        END
        IF (EXISTS(SELECT TOP 1 ApplicationId FROM dbo.aspnet_Applications))
        BEGIN
            SELECT N'aspnet_Applications'
            RETURN
        END
    END
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_UsersInBranchesSelect]    Script Date: 08/ ↙
    16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Godwin Mathias
--Reviewer            :   Godwin Mathias
--Date Created        :   05/18/2011
--Last Updated        :   05/18/2011
--Last Updated By     :   Godwin Mathias
--Description         :   Stored procedure for retrieving values in the dbo.             ↙
    aspnet_UsersInBranches  Table

CREATE PROC [dbo].[aspnet_UsersInBranchesSelect]
```

```
(
    @Code uniqueidentifier = null,
    @UserId uniqueidentifier = null,
    @SchoolCode nvarchar(50) = null,
    @BranchCode NVARCHAR(50) = NULL,
    @Deleted BIT = NULL
)

AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM  [dbo].[aspnet_UsersInBranches] WHERE ((Code = @Code) AND↙
     (UserId=@UserId) AND (SchoolCode=@SchoolCode) AND (BranchCode=@BranchCode) AND      ↙
    (Deleted=@Deleted)))

            BEGIN
                SELECT * FROM  [dbo].[aspnet_UsersInBranches]
                WHERE ((Code = @Code) AND (UserId=@UserId) AND (SchoolCode=@SchoolCode)   ↙
    AND (BranchCode=@BranchCode) AND (Deleted=@Deleted))
            END
        ELSE
            BEGIN
                SELECT * FROM  [dbo].[aspnet_UsersInBranches]
                WHERE ((SchoolCode=@SchoolCode) AND (BranchCode=@BranchCode) AND (Deleted=↙
    @Deleted))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [dbo].[aspnet_UsersInBranchesSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [dbo].[aspnet_UsersInBranchesInsertUpdate]    Script Date↙
    : 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Godwin Mathias
--Reviewer          :   Godwin Mathias
--Date Created      :   05/18/2011
--Last Updated      :   05/18/2011
--Last Updated By   :   Godwin Mathias
--Description       :   Stored procedure for inserting/updating values in  [dbo].        ↙
    [aspnet_UsersInBranches] Table
```

```
CREATE PROC [dbo].[aspnet_UsersInBranchesInsertUpdate]
(
    @Code UNIQUEIDENTIFIER = Null,
    @UserId UNIQUEIDENTIFIER = null,
    @SchoolCode NVARCHAR(50) = null,
    @BranchCode NVARCHAR(50) = null,
    @DefaultBranch BIT=Null,
    @CreatedOn DATETIME2(7) = null,
    @CreatedBy NVARCHAR(50) = null,
    @ModifiedOn DATETIME2(7) = null,
    @ModifiedBy NVARCHAR(50) = null,
    @Deleted BIT = null
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [dbo].[aspnet_UsersInBranches] WHERE ((Code = @Code) ↙
    AND (UserId=@UserId) AND (SchoolCode=@SchoolCode) AND (BranchCode=@BranchCode)))
            BEGIN
                INSERT INTO [dbo].[aspnet_UsersInBranches]
                (
                    [Code],
                    [UserId],
                    [SchoolCode],
                    [BranchCode],
                    [DefaultBranch],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    NEWID(),
                    @UserId,
                    @SchoolCode,
                    @BranchCode,
                    @DefaultBranch,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [dbo].[aspnet_UsersInBranches]
                SET
                    [UserId]=@UserId,
                    [SchoolCode]=@SchoolCode,
                    [BranchCode]=@BranchCode,
                    [DefaultBranch]=@DefaultBranch,
                    [ModifiedOn] = @ModifiedOn,
                    [ModifiedBy] = @ModifiedBy,
                    [Deleted]=@Deleted
                WHERE ([Code]=@Code)
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON  [dbo].[aspnet_UsersInBranchesInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [dbo].[aspnet_UsersInBranchesDeletePermanently]    Script↙
     Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Godwin Mathias
--Reviewer          :   Godwin Mathias
--Date Created      :   05/18/2011
--Last Updated      :   05/18/2011
--Last Updated By   :   Godwin Mathias
--Description       :   Stored procedure for deleting permanently values in the dbo.      ↙
    aspnet_UsersInRoles  Table

CREATE PROC [dbo].[aspnet_UsersInBranchesDeletePermanently]
(
    @Code uniqueidentifier = null,
    @UserId uniqueidentifier = null,
    @SchoolCode nvarchar(50) = null,
    @BranchCode NVARCHAR(50) = NULL
)

AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM  [dbo].[aspnet_UsersInBranches] WHERE ((Code = @Code) AND↙
     (UserId=@UserId) AND (SchoolCode=@SchoolCode) AND (BranchCode=@BranchCode)))

            BEGIN
                DELETE FROM [dbo].[aspnet_UsersInBranches]
                WHERE (([Code]=@Code) AND (UserId=@UserId) AND (SchoolCode=@SchoolCode)   ↙
    AND (BranchCode=@BranchCode))
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [dbo].[aspnet_UsersInBranchesDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [dbo].[aspnet_UsersInBranchesDelete]    Script Date: 08/ ↙
    16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Godwin Mathias
--Reviewer          :   Godwin Mathias
--Date Created      :   05/18/2011
--Last Updated      :   05/18/2011
--Last Updated By   :   Godwin Mathias
--Description       :   Stored procedure for retrieving values in the dbo.                   ↙
    aspnet_UsersInBranches  Table

CREATE PROC [dbo].[aspnet_UsersInBranchesDelete]
(
    @Code uniqueidentifier = null,
    @UserId uniqueidentifier = null,
    @SchoolCode nvarchar(50) = null,
    @BranchCode NVARCHAR(50) = NULL,
    @Deleted BIT = NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)

AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM  [dbo].[aspnet_UsersInBranches] WHERE ((Code = @Code) AND↙
     (UserId=@UserId) AND (SchoolCode=@SchoolCode) AND (BranchCode=@BranchCode)))

            BEGIN
                UPDATE  [dbo].[aspnet_UsersInBranches]
                SET
                    Deleted=@Deleted,
                    DeletedOn=@DeletedOn,
                    DeletedBy=@DeletedBy
                WHERE ((Code = @Code) AND (UserId=@UserId) AND (SchoolCode=@SchoolCode)   ↙
    AND (BranchCode=@BranchCode))
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [dbo].[aspnet_UsersInBranchesDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [dbo].[aspnet_Roles_DeleteRole]    Script Date: 08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_Roles_DeleteRole]
    @ApplicationName            nvarchar(256),
    @RoleName                   nvarchar(256),
    @DeleteOnlyIfRoleIsEmpty    bit
AS
BEGIN
    DECLARE @ApplicationId uniqueidentifier
    SELECT  @ApplicationId = NULL
    SELECT  @ApplicationId = ApplicationId FROM aspnet_Applications WHERE LOWER(@ApplicationName) = LoweredApplicationName
    IF (@ApplicationId IS NULL)
        RETURN(1)

    DECLARE @ErrorCode     int
    SET @ErrorCode = 0

    DECLARE @TranStarted   bit
    SET @TranStarted = 0

    IF( @@TRANCOUNT = 0 )
    BEGIN
        BEGIN TRANSACTION
        SET @TranStarted = 1
    END
    ELSE
        SET @TranStarted = 0

    DECLARE @RoleId   uniqueidentifier
    SELECT  @RoleId = NULL
    SELECT  @RoleId = RoleId FROM dbo.aspnet_Roles WHERE LoweredRoleName = LOWER(@RoleName) AND ApplicationId = @ApplicationId

    IF (@RoleId IS NULL)
    BEGIN
        SELECT @ErrorCode = 1
        GOTO Cleanup
    END
    IF (@DeleteOnlyIfRoleIsEmpty <> 0)
    BEGIN
        IF (EXISTS (SELECT RoleId FROM dbo.aspnet_UsersInRoles  WHERE @RoleId = RoleId))
        BEGIN
            SELECT @ErrorCode = 2
            GOTO Cleanup
        END
    END


    DELETE FROM dbo.aspnet_UsersInRoles  WHERE @RoleId = RoleId

    IF( @@ERROR <> 0 )
    BEGIN
```

```
        SET @ErrorCode = -1
        GOTO Cleanup
    END

    DELETE FROM dbo.aspnet_Roles WHERE @RoleId = RoleId  AND ApplicationId = @ ↙
    ApplicationId

    IF( @@ERROR <> 0 )
    BEGIN
        SET @ErrorCode = -1
        GOTO Cleanup
    END

    IF( @TranStarted = 1 )
    BEGIN
        SET @TranStarted = 0
        COMMIT TRANSACTION
    END

    RETURN(0)

Cleanup:

    IF( @TranStarted = 1 )
    BEGIN
        SET @TranStarted = 0
        ROLLBACK TRANSACTION
    END

    RETURN @ErrorCode
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_PersonalizationPerUser_SetPageSettings]    ↙
    Script Date: 08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_PersonalizationPerUser_SetPageSettings] (
    @ApplicationName  NVARCHAR(256),
    @UserName         NVARCHAR(256),
    @Path             NVARCHAR(256),
    @PageSettings     IMAGE,
    @CurrentTimeUtc   DATETIME)
AS
BEGIN
    DECLARE @ApplicationId UNIQUEIDENTIFIER
    DECLARE @PathId UNIQUEIDENTIFIER
    DECLARE @UserId UNIQUEIDENTIFIER

    SELECT @ApplicationId = NULL
    SELECT @PathId = NULL
    SELECT @UserId = NULL

    EXEC dbo.aspnet_Applications_CreateApplication @ApplicationName, @ApplicationId OUTPUT

    SELECT @PathId = u.PathId FROM dbo.aspnet_Paths u WHERE u.ApplicationId = @ ↙
    ApplicationId AND u.LoweredPath = LOWER(@Path)
    IF (@PathId IS NULL)
    BEGIN
        EXEC dbo.aspnet_Paths_CreatePath @ApplicationId, @Path, @PathId OUTPUT
    END

    SELECT @UserId = u.UserId FROM dbo.aspnet_Users u WHERE u.ApplicationId = @ ↙
    ApplicationId AND u.LoweredUserName = LOWER(@UserName)
    IF (@UserId IS NULL)
    BEGIN
```

```
        EXEC dbo.aspnet_Users_CreateUser @ApplicationId, @UserName, 0, @CurrentTimeUtc, @ ↙
    UserId OUTPUT
    END

    UPDATE   dbo.aspnet_Users WITH (ROWLOCK)
    SET      LastActivityDate = @CurrentTimeUtc
    WHERE    UserId = @UserId
    IF (@@ROWCOUNT = 0) -- Username not found
        RETURN

    IF (EXISTS(SELECT PathId FROM dbo.aspnet_PersonalizationPerUser WHERE UserId = @UserId↙
     AND PathId = @PathId))
        UPDATE dbo.aspnet_PersonalizationPerUser SET PageSettings = @PageSettings,        ↙
    LastUpdatedDate = @CurrentTimeUtc WHERE UserId = @UserId AND PathId = @PathId
    ELSE
        INSERT INTO dbo.aspnet_PersonalizationPerUser(UserId, PathId, PageSettings,       ↙
    LastUpdatedDate) VALUES (@UserId, @PathId, @PageSettings, @CurrentTimeUtc)
    RETURN 0
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_PersonalizationPerUser_ResetPageSettings]  ↙
      Script Date: 08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_PersonalizationPerUser_ResetPageSettings] (
    @ApplicationName  NVARCHAR(256),
    @UserName         NVARCHAR(256),
    @Path             NVARCHAR(256),
    @CurrentTimeUtc   DATETIME)
AS
BEGIN
    DECLARE @ApplicationId UNIQUEIDENTIFIER
    DECLARE @PathId UNIQUEIDENTIFIER
    DECLARE @UserId UNIQUEIDENTIFIER

    SELECT @ApplicationId = NULL
    SELECT @PathId = NULL
    SELECT @UserId = NULL

    EXEC dbo.aspnet_Personalization_GetApplicationId @ApplicationName, @ApplicationId     ↙
    OUTPUT
    IF (@ApplicationId IS NULL)
    BEGIN
        RETURN
    END

    SELECT @PathId = u.PathId FROM dbo.aspnet_Paths u WHERE u.ApplicationId = @           ↙
    ApplicationId AND u.LoweredPath = LOWER(@Path)
    IF (@PathId IS NULL)
    BEGIN
        RETURN
    END

    SELECT @UserId = u.UserId FROM dbo.aspnet_Users u WHERE u.ApplicationId = @           ↙
    ApplicationId AND u.LoweredUserName = LOWER(@UserName)
    IF (@UserId IS NULL)
    BEGIN
        RETURN
    END

    UPDATE   dbo.aspnet_Users WITH (ROWLOCK)
    SET      LastActivityDate = @CurrentTimeUtc
    WHERE    UserId = @UserId
    IF (@@ROWCOUNT = 0) -- Username not found
        RETURN
    DELETE FROM dbo.aspnet_PersonalizationPerUser WHERE PathId = @PathId AND UserId = @UserId
    RETURN 0
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_PersonalizationPerUser_GetPageSettings]    Script Date: 08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_PersonalizationPerUser_GetPageSettings] (
    @ApplicationName  NVARCHAR(256),
    @UserName         NVARCHAR(256),
    @Path             NVARCHAR(256),
    @CurrentTimeUtc   DATETIME)
AS
BEGIN
    DECLARE @ApplicationId UNIQUEIDENTIFIER
    DECLARE @PathId UNIQUEIDENTIFIER
    DECLARE @UserId UNIQUEIDENTIFIER

    SELECT @ApplicationId = NULL
    SELECT @PathId = NULL
    SELECT @UserId = NULL

    EXEC dbo.aspnet_Personalization_GetApplicationId @ApplicationName, @ApplicationId OUTPUT
    IF (@ApplicationId IS NULL)
    BEGIN
        RETURN
    END

    SELECT @PathId = u.PathId FROM dbo.aspnet_Paths u WHERE u.ApplicationId = @ApplicationId AND u.LoweredPath = LOWER(@Path)
    IF (@PathId IS NULL)
    BEGIN
        RETURN
    END

    SELECT @UserId = u.UserId FROM dbo.aspnet_Users u WHERE u.ApplicationId = @ApplicationId AND u.LoweredUserName = LOWER(@UserName)
    IF (@UserId IS NULL)
    BEGIN
        RETURN
    END

    UPDATE   dbo.aspnet_Users WITH (ROWLOCK)
    SET      LastActivityDate = @CurrentTimeUtc
    WHERE    UserId = @UserId
    IF (@@ROWCOUNT = 0) -- Username not found
        RETURN

    SELECT p.PageSettings FROM dbo.aspnet_PersonalizationPerUser p WHERE p.PathId = @PathId AND p.UserId = @UserId
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_PersonalizationAllUsers_SetPageSettings]    Script Date: 08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_PersonalizationAllUsers_SetPageSettings] (
    @ApplicationName  NVARCHAR(256),
    @Path             NVARCHAR(256),
    @PageSettings     IMAGE,
    @CurrentTimeUtc   DATETIME)
AS
BEGIN
    DECLARE @ApplicationId UNIQUEIDENTIFIER
    DECLARE @PathId UNIQUEIDENTIFIER

    SELECT @ApplicationId = NULL
    SELECT @PathId = NULL

    EXEC dbo.aspnet_Applications_CreateApplication @ApplicationName, @ApplicationId OUTPUT

    SELECT @PathId = u.PathId FROM dbo.aspnet_Paths u WHERE u.ApplicationId = @ApplicationId AND u.LoweredPath = LOWER(@Path)
    IF (@PathId IS NULL)
    BEGIN
        EXEC dbo.aspnet_Paths_CreatePath @ApplicationId, @Path, @PathId OUTPUT
    END

    IF (EXISTS(SELECT PathId FROM dbo.aspnet_PersonalizationAllUsers WHERE PathId = @PathId))
        UPDATE dbo.aspnet_PersonalizationAllUsers SET PageSettings = @PageSettings, LastUpdatedDate = @CurrentTimeUtc WHERE PathId = @PathId
    ELSE
        INSERT INTO dbo.aspnet_PersonalizationAllUsers(PathId, PageSettings, LastUpdatedDate) VALUES (@PathId, @PageSettings, @CurrentTimeUtc)
    RETURN 0
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_PersonalizationAllUsers_ResetPageSettings]    Script Date: 08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_PersonalizationAllUsers_ResetPageSettings] (
    @ApplicationName  NVARCHAR(256),
    @Path              NVARCHAR(256))
AS
BEGIN
    DECLARE @ApplicationId UNIQUEIDENTIFIER
    DECLARE @PathId UNIQUEIDENTIFIER

    SELECT @ApplicationId = NULL
    SELECT @PathId = NULL

    EXEC dbo.aspnet_Personalization_GetApplicationId @ApplicationName, @ApplicationId OUTPUT
    IF (@ApplicationId IS NULL)
    BEGIN
        RETURN
    END

    SELECT @PathId = u.PathId FROM dbo.aspnet_Paths u WHERE u.ApplicationId = @ApplicationId AND u.LoweredPath = LOWER(@Path)
    IF (@PathId IS NULL)
    BEGIN
        RETURN
    END

    DELETE FROM dbo.aspnet_PersonalizationAllUsers WHERE PathId = @PathId
    RETURN 0
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_PersonalizationAllUsers_GetPageSettings]    Script Date: 08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
```

```
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_PersonalizationAllUsers_GetPageSettings] (
    @ApplicationName  NVARCHAR(256),
    @Path              NVARCHAR(256))
AS
BEGIN
    DECLARE @ApplicationId UNIQUEIDENTIFIER
    DECLARE @PathId UNIQUEIDENTIFIER

    SELECT @ApplicationId = NULL
    SELECT @PathId = NULL

    EXEC dbo.aspnet_Personalization_GetApplicationId @ApplicationName, @ApplicationId     ↙
    OUTPUT
    IF (@ApplicationId IS NULL)
    BEGIN
        RETURN
    END

    SELECT @PathId = u.PathId FROM dbo.aspnet_Paths u WHERE u.ApplicationId = @          ↙
    ApplicationId AND u.LoweredPath = LOWER(@Path)
    IF (@PathId IS NULL)
    BEGIN
        RETURN
    END

    SELECT p.PageSettings FROM dbo.aspnet_PersonalizationAllUsers p WHERE p.PathId = @    ↙
    PathId
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_Profile_SetProperties]    Script Date: 08/ ↙
    16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_Profile_SetProperties]
    @ApplicationName        nvarchar(256),
    @PropertyNames          ntext,
    @PropertyValuesString   ntext,
    @PropertyValuesBinary   image,
    @UserName               nvarchar(256),
    @IsUserAnonymous        bit,
    @CurrentTimeUtc         datetime
AS
BEGIN
    DECLARE @ApplicationId uniqueidentifier
    SELECT  @ApplicationId = NULL

    DECLARE @ErrorCode     int
    SET @ErrorCode = 0

    DECLARE @TranStarted   bit
    SET @TranStarted = 0

    IF( @@TRANCOUNT = 0 )
    BEGIN
       BEGIN TRANSACTION
       SET @TranStarted = 1
    END
    ELSE
        SET @TranStarted = 0

    EXEC dbo.aspnet_Applications_CreateApplication @ApplicationName, @ApplicationId OUTPUT
    IF( @@ERROR <> 0 )
    BEGIN
        SET @ErrorCode = -1
        GOTO Cleanup
    END

    DECLARE @UserId uniqueidentifier
    DECLARE @LastActivityDate datetime
    SELECT  @UserId = NULL
    SELECT  @LastActivityDate = @CurrentTimeUtc

    SELECT @UserId = UserId
    FROM   dbo.aspnet_Users
    WHERE  ApplicationId = @ApplicationId AND LoweredUserName = LOWER(@UserName)
    IF (@UserId IS NULL)
        EXEC dbo.aspnet_Users_CreateUser @ApplicationId, @UserName, @IsUserAnonymous, @   ↙
    LastActivityDate, @UserId OUTPUT

    IF( @@ERROR <> 0 )
    BEGIN
        SET @ErrorCode = -1
        GOTO Cleanup
    END

    UPDATE dbo.aspnet_Users
    SET    LastActivityDate=@CurrentTimeUtc
    WHERE  UserId = @UserId

    IF( @@ERROR <> 0 )
    BEGIN
        SET @ErrorCode = -1
        GOTO Cleanup
    END

    IF (EXISTS( SELECT *
               FROM   dbo.aspnet_Profile
               WHERE  UserId = @UserId))
        UPDATE dbo.aspnet_Profile
        SET    PropertyNames=@PropertyNames, PropertyValuesString = @PropertyValuesString,
               PropertyValuesBinary = @PropertyValuesBinary, LastUpdatedDate=@          ↙
    CurrentTimeUtc
        WHERE  UserId = @UserId
    ELSE
        INSERT INTO dbo.aspnet_Profile(UserId, PropertyNames, PropertyValuesString,        ↙
    PropertyValuesBinary, LastUpdatedDate)
             VALUES (@UserId, @PropertyNames, @PropertyValuesString, @PropertyValuesBinary↙
    , @CurrentTimeUtc)

    IF( @@ERROR <> 0 )
    BEGIN
        SET @ErrorCode = -1
        GOTO Cleanup
    END

    IF( @TranStarted = 1 )
    BEGIN
        SET @TranStarted = 0
        COMMIT TRANSACTION
    END

    RETURN 0

Cleanup:

    IF( @TranStarted = 1 )
    BEGIN
        SET @TranStarted = 0
        ROLLBACK TRANSACTION
    END

    RETURN @ErrorCode

END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_Profile_GetProperties]    Script Date: 08/ ↙
    16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_Profile_GetProperties]
    @ApplicationName      nvarchar(256),
    @UserName             nvarchar(256),
    @CurrentTimeUtc       datetime
AS
BEGIN
    DECLARE @ApplicationId uniqueidentifier
    SELECT  @ApplicationId = NULL
    SELECT  @ApplicationId = ApplicationId FROM dbo.aspnet_Applications WHERE LOWER(@     ↙
    ApplicationName) = LoweredApplicationName
    IF (@ApplicationId IS NULL)
        RETURN

    DECLARE @UserId uniqueidentifier
    SELECT  @UserId = NULL

    SELECT @UserId = UserId
    FROM   dbo.aspnet_Users
    WHERE  ApplicationId = @ApplicationId AND LoweredUserName = LOWER(@UserName)

    IF (@UserId IS NULL)
        RETURN
    SELECT TOP 1 PropertyNames, PropertyValuesString, PropertyValuesBinary
    FROM         dbo.aspnet_Profile
    WHERE        UserId = @UserId

    IF (@@ROWCOUNT > 0)
    BEGIN
        UPDATE dbo.aspnet_Users
        SET    LastActivityDate=@CurrentTimeUtc
        WHERE  UserId = @UserId
    END
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_Profile_GetProfiles]    Script Date: 08/16/↙
    2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_Profile_GetProfiles]
    @ApplicationName        nvarchar(256),
    @ProfileAuthOptions     int,
    @PageIndex              int,
    @PageSize               int,
    @UserNameToMatch        nvarchar(256) = NULL,
    @InactiveSinceDate      datetime      = NULL
AS
BEGIN
    DECLARE @ApplicationId uniqueidentifier
    SELECT  @ApplicationId = NULL
    SELECT  @ApplicationId = ApplicationId FROM aspnet_Applications WHERE LOWER(@         ↙
    ApplicationName) = LoweredApplicationName
    IF (@ApplicationId IS NULL)
```

```
        RETURN

    -- Set the page bounds
    DECLARE @PageLowerBound int
    DECLARE @PageUpperBound int
    DECLARE @TotalRecords   int
    SET @PageLowerBound = @PageSize * @PageIndex
    SET @PageUpperBound = @PageSize - 1 + @PageLowerBound

    -- Create a temp table TO store the select results
    CREATE TABLE #PageIndexForUsers
    (
        IndexId int IDENTITY (0, 1) NOT NULL,
        UserId uniqueidentifier
    )

    -- Insert into our temp table
    INSERT INTO #PageIndexForUsers (UserId)
        SELECT  u.UserId
        FROM    dbo.aspnet_Users u, dbo.aspnet_Profile p
        WHERE   ApplicationId = @ApplicationId
            AND u.UserId = p.UserId
            AND (@InactiveSinceDate IS NULL OR LastActivityDate <= @InactiveSinceDate)
            AND (     (@ProfileAuthOptions = 2)
                   OR (@ProfileAuthOptions = 0 AND IsAnonymous = 1)
                   OR (@ProfileAuthOptions = 1 AND IsAnonymous = 0)
                 )
            AND (@UserNameToMatch IS NULL OR LoweredUserName LIKE LOWER(@UserNameToMatch))
        ORDER BY UserName

    SELECT  u.UserName, u.IsAnonymous, u.LastActivityDate, p.LastUpdatedDate,
            DATALENGTH(p.PropertyNames) + DATALENGTH(p.PropertyValuesString) + DATALENGTH ↙
    (p.PropertyValuesBinary)
    FROM    dbo.aspnet_Users u, dbo.aspnet_Profile p, #PageIndexForUsers i
    WHERE   u.UserId = p.UserId AND p.UserId = i.UserId AND i.IndexId >= @PageLowerBound ↙
    AND i.IndexId <= @PageUpperBound

    SELECT COUNT(*)
    FROM   #PageIndexForUsers

    DROP TABLE #PageIndexForUsers
END
GO
/****** Object:  StoredProcedure [dbo].[aspnet_Profile_GetNumberOfInactiveProfiles]     ↙
    Script Date: 08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_Profile_GetNumberOfInactiveProfiles]
    @ApplicationName        nvarchar(256),
    @ProfileAuthOptions     int,
    @InactiveSinceDate      datetime
AS
BEGIN
    DECLARE @ApplicationId uniqueidentifier
    SELECT  @ApplicationId = NULL
    SELECT  @ApplicationId = ApplicationId FROM aspnet_Applications WHERE LOWER(@       ↙
    ApplicationName) = LoweredApplicationName
    IF (@ApplicationId IS NULL)
    BEGIN
        SELECT 0
        RETURN
    END

    SELECT  COUNT(*)
    FROM    dbo.aspnet_Users u, dbo.aspnet_Profile p
    WHERE   ApplicationId = @ApplicationId
        AND u.UserId = p.UserId
        AND (LastActivityDate <= @InactiveSinceDate)
        AND (
                (@ProfileAuthOptions = 2)
                OR (@ProfileAuthOptions = 0 AND IsAnonymous = 1)
                OR (@ProfileAuthOptions = 1 AND IsAnonymous = 0)
            )
END
GO
/****** Object:  View [dbo].[vw_aspnet_WebPartState_User]    Script Date: 08/16/2011 01:24↙
    :34 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE VIEW [dbo].[vw_aspnet_WebPartState_User]
  AS SELECT [dbo].[aspnet_PersonalizationPerUser].[PathId], [dbo].                       ↙
    [aspnet_PersonalizationPerUser].[UserId], [DataSize]=DATALENGTH([dbo].               ↙
    [aspnet_PersonalizationPerUser].[PageSettings]), [dbo].[aspnet_PersonalizationPerUser]↙
    .[LastUpdatedDate]
  FROM [dbo].[aspnet_PersonalizationPerUser]
GO
/****** Object:  View [dbo].[vw_aspnet_WebPartState_Shared]    Script Date: 08/16/2011 01:↙
    24:34 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE VIEW [dbo].[vw_aspnet_WebPartState_Shared]
  AS SELECT [dbo].[aspnet_PersonalizationAllUsers].[PathId], [DataSize]=DATALENGTH([dbo]. ↙
    [aspnet_PersonalizationAllUsers].[PageSettings]), [dbo].                             ↙
    [aspnet_PersonalizationAllUsers].[LastUpdatedDate]
  FROM [dbo].[aspnet_PersonalizationAllUsers]
GO
/****** Object:  View [dbo].[vw_aspnet_Profiles]    Script Date: 08/16/2011 01:24:34 *****↙
    */
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE VIEW [dbo].[vw_aspnet_Profiles]
  AS SELECT [dbo].[aspnet_Profile].[UserId], [dbo].[aspnet_Profile].[LastUpdatedDate],
      [DataSize]=  DATALENGTH([dbo].[aspnet_Profile].[PropertyNames])
                 + DATALENGTH([dbo].[aspnet_Profile].[PropertyValuesString])
                 + DATALENGTH([dbo].[aspnet_Profile].[PropertyValuesBinary])
  FROM [dbo].[aspnet_Profile]
GO
/****** Object:  View [dbo].[vw_aspnet_MembershipUsers]    Script Date: 08/16/2011 01:24: ↙
    34 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE VIEW [dbo].[vw_aspnet_MembershipUsers]
  AS SELECT [dbo].[aspnet_Membership].[UserId],
            [dbo].[aspnet_Membership].[PasswordFormat],
            [dbo].[aspnet_Membership].[MobilePIN],
            [dbo].[aspnet_Membership].[Email],
            [dbo].[aspnet_Membership].[LoweredEmail],
            [dbo].[aspnet_Membership].[PasswordQuestion],
            [dbo].[aspnet_Membership].[PasswordAnswer],
            [dbo].[aspnet_Membership].[IsApproved],
            [dbo].[aspnet_Membership].[IsLockedOut],
            [dbo].[aspnet_Membership].[CreateDate],
            [dbo].[aspnet_Membership].[LastLoginDate],
            [dbo].[aspnet_Membership].[LastPasswordChangedDate],
            [dbo].[aspnet_Membership].[LastLockoutDate],
            [dbo].[aspnet_Membership].[FailedPasswordAttemptCount],
            [dbo].[aspnet_Membership].[FailedPasswordAttemptWindowStart],
            [dbo].[aspnet_Membership].[FailedPasswordAnswerAttemptCount],
            [dbo].[aspnet_Membership].[FailedPasswordAnswerAttemptWindowStart],
            [dbo].[aspnet_Membership].[Comment],
            [dbo].[aspnet_Users].[ApplicationId],
            [dbo].[aspnet_Users].[UserName],
            [dbo].[aspnet_Users].[MobileAlias],
            [dbo].[aspnet_Users].[IsAnonymous],
            [dbo].[aspnet_Users].[LastActivityDate]
  FROM [dbo].[aspnet_Membership] INNER JOIN [dbo].[aspnet_Users]
      ON [dbo].[aspnet_Membership].[UserId] = [dbo].[aspnet_Users].[UserId]
GO
/****** Object:  View [dbo].[vw_aspnet_UsersInRoles]    Script Date: 08/16/2011 01:24:34 *↙
    *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE VIEW [dbo].[vw_aspnet_UsersInRoles]
  AS SELECT [dbo].[aspnet_UsersInRoles].[UserId], [dbo].[aspnet_UsersInRoles].[RoleId]
  FROM [dbo].[aspnet_UsersInRoles]
GO
/****** Object:  StoredProcedure [Academics].[SPSubmissionsSelect]    Script Date: 08/16/ ↙
    2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Academics.Submissions ↙
    Table

CREATE PROC [Academics].[SPSubmissionsSelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Deleted BIT=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Academics].[Submissions] WHERE (([Code] = @Code) AND (  ↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted)))
            BEGIN
            SELECT * FROM [Academics].[Submissions]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Academics].[Submissions]
            WHERE
                (
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Academics].[SPSubmissionsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Academics].[SPSubmissionsInsertUpdate]    Script Date:  ↙
    08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/18/2011
--Last Updated        :   07/18/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for saving values into Academics.Submissions     ↙
    Table

CREATE PROC [Academics].[SPSubmissionsInsertUpdate]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @FacultyCode BIGINT=NULL,
    @DepartmentCode BIGINT=NULL,
    @CourseCode BIGINT=NULL,
    @ProgramCode BIGINT=NULL,
    @LevelCode INT=NULL,
    @StudentCode BIGINT=NULL,
    @MatricNo NVARCHAR(50)=NULL,
    @SessionCode BIGINT=NULL,
    @SemesterCode BIGINT=NULL,
    @Submitted BIT=NULL,
    @ScreenCode NVARCHAR(50)=NULL,
    @Accepted BIT=NULL,
    @AcceptedOn DATETIME2(7)=NULL,
    @AcceptedBy NVARCHAR(50)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
```

```
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Academics].[Submissions] WHERE ([Code] = @Code) AND ↙
    ([UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
                INSERT INTO [Academics].[Submissions]
                (
                    [UniversityCode],
                    [FacultyCode],
                    [DepartmentCode],
                    [CourseCode],
                    [ProgramCode],
                    [LevelCode],
                    [StudentCode],
                    [MatricNo],
                    [SessionCode],
                    [SemesterCode],
                    [Submitted],
                    [ScreenCode],
                    [Accepted],
                    [AcceptedOn],
                    [AcceptedBy],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]

                )
                VALUES
                (
                    @UniversityCode,
                    @FacultyCode,
                    @DepartmentCode,
                    @CourseCode,
                    @ProgramCode,
                    @LevelCode,
                    @StudentCode,
                    @MatricNo,
                    @SessionCode,
                    @SemesterCode,
                    @Submitted,
                    @ScreenCode,
                    @Accepted,
                    @AcceptedOn,
                    @AcceptedBy,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted

                )
            END
        ELSE
            BEGIN
                UPDATE [Academics].[Submissions]
                SET
                    [UniversityCode]=@UniversityCode,
                    [FacultyCode]=@FacultyCode,
                    [DepartmentCode]=@DepartmentCode,
                    [CourseCode]=@CourseCode,
                    [ProgramCode]=@ProgramCode,
                    [LevelCode]=@LevelCode,
                    [StudentCode]=@StudentCode,
                    [MatricNo]=@MatricNo,
                    [SessionCode]=@SessionCode,
                    [SemesterCode]=@SemesterCode,
                    [Submitted]=@Submitted,
                    [ScreenCode]=@ScreenCode,
                    [Accepted]=@Accepted,
                    [AcceptedOn]=@AcceptedOn,
                    [AcceptedBy]=@AcceptedBy,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted

                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Academics].[SPSubmissionsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Academics].[SPSubmissionsDeletePermanently]    Script   ↙
    Date: 08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/18/2011
--Last Updated        :   07/18/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for deleting values from Academics.Submissions   ↙
    Table

CREATE PROC [Academics].[SPSubmissionsDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Academics].[Submissions] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Academics].[Submissions]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Academics].[SPSubmissionsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Academics].[SPSubmissionsDelete]    Script Date: 08/16/ ↙
    2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Academics.  ↙
    Submissions Table

CREATE PROC [Academics].[SPSubmissionsDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Academics].[Submissions] WHERE ([Code] = @Code) AND     ↙
    (Deleted=@Deleted))
            BEGIN
            UPDATE [Academics].[Submissions]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Academics].[SPSubmissionsDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPScreensSelect]    Script Date: 08/16/2011 01: ↙
    24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from SetUp.Screens Table

CREATE PROC [SetUp].[SPScreensSelect]
(
    @Code NVARCHAR(50)=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Screens] WHERE ([Code] = @Code) AND (          ↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
            SELECT * FROM [SetUp].[Screens]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [SetUp].[Screens]
            WHERE
                (
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPScreensSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPScreensInsertUpdate]    Script Date: 08/16/  ↙
    2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into SetUp.Screens Table

CREATE PROC [SetUp].[SPScreensInsertUpdate]
(
    @Code NVARCHAR(50)=NULL,
    @ModuleCode BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @ScreenDescription NVARCHAR(256)=NULL,
    @Url NVARCHAR(256)=NULL,
    @Notes NVARCHAR(500)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[Screens]  WHERE ([Code] = @Code) AND (      ↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
```

```
                INSERT INTO [SetUp].[Screens]
                (
                    [Code],
                    [ModuleCode],
                    [UniversityCode],
                    [ScreenDescription],
                    [Url],
                    [Notes],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]

                )
                VALUES
                (
                    @Code,
                    @ModuleCode,
                    @UniversityCode,
                    @ScreenDescription,
                    @Url,
                    @Notes,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [SetUp].[Screens]
                SET
                    [ModuleCode]=@ModuleCode,
                    [UniversityCode]=@UniversityCode,
                    [ScreenDescription]=@ScreenDescription,
                    [Url]=@Url,
                    [Notes]=@Notes,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPScreensInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPScreensDeletePermanently]    Script Date: 08/ ↙
```

```
    16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/12/2011
--Last Updated        :   07/12/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for deleting values from SetUp.Screens Table

CREATE PROC [SetUp].[SPScreensDeletePermanently]
(
    @Code NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Screens] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [SetUp].[Screens]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPScreensDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPScreensDelete]    Script Date: 08/16/2011 01: ↙
    24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/28/2011
--Last Updated        :   07/28/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for deleting values temporarily from SetUp.      ↙
```

```
    Screens Table

CREATE PROC [SetUp].[SPScreensDelete]
(
    @Code NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[Screens] WHERE (([Code] = @Code) AND       ↙
    (Deleted=@Deleted)))
            BEGIN
            UPDATE [SetUp].[Screens]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPScreensDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Assessment].[SPRecordsSelect]    Script Date: 08/16/2011↙
     01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/18/2011
--Last Updated        :   07/18/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for retrieving values from Assessment.Records    ↙
    Table
```

```
CREATE PROC [Assessment].[SPRecordsSelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Deleted BIT=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Assessment].[Records] WHERE (([Code] = @Code) AND (     ↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted)))
            BEGIN
            SELECT * FROM [Assessment].[Records]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Assessment].[Records]
            WHERE
                (
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Assessment].[SPRecordsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Assessment].[SPRecordsInsertUpdate]    Script Date: 08/ ↙
    16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
```

```
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Assessment.Records Table

CREATE PROC [Assessment].[SPRecordsInsertUpdate]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @FacultyCode BIGINT=NULL,
    @DepartmentCode BIGINT=NULL,
    @CourseCode BIGINT=NULL,
    @ProgramCode BIGINT=NULL,
    @SessionCode BIGINT=NULL,
    @SemesterCode BIGINT=NULL,
    @LevelCode INT=NULL,
    @AssessmentTypeCode NVARCHAR(50)=NULL,
    @SubCourseCode NVARCHAR(50)=NULL,
    @StaffCode NVARCHAR(50)=NULL,
    @Score DECIMAL(18,2)=NULL,
    @GradeDescription NVARCHAR(50)=NULL,
    @GP DECIMAL(18,0)=NULL,
    @Position BIGINT=NULL,
    @StudentCode BIGINT=NULL,
    @MatricNo NVARCHAR(50)=NULL,
    @Remark NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Assessment].[Records] WHERE ([Code] = @Code) AND (   ↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
                INSERT INTO [Assessment].[Records]
                (
                    [UniversityCode],
                    [FacultyCode],
                    [DepartmentCode],
                    [CourseCode],
                    [ProgramCode],
                    [SessionCode],
                    [SemesterCode],
                    [LevelCode],
                    [AssessmentTypeCode],
                    [SubCourseCode],
                    [StaffCode],
                    [Score],
                    [GradeDescription],
                    [GP],
                    [Position],
                    [StudentCode],
                    [MatricNo],
                    [Remark],
                    [ScreenCode],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    @UniversityCode,
                    @FacultyCode,
                    @DepartmentCode,
                    @CourseCode,
                    @ProgramCode,
                    @SessionCode,
                    @SemesterCode,
                    @LevelCode,
                    @AssessmentTypeCode,
                    @SubCourseCode,
                    @StaffCode,
                    @Score,
                    @GradeDescription,
                    @GP,
                    @Position,
                    @StudentCode,
                    @MatricNo,
                    @Remark,
                    @ScreenCode,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted

                )
            END
        ELSE
            BEGIN
                UPDATE [Assessment].[Records]
                SET
                    [UniversityCode]=@UniversityCode,
                    [FacultyCode]=@FacultyCode,
                    [DepartmentCode]=@DepartmentCode,
                    [CourseCode]=@CourseCode,
                    [ProgramCode]=@ProgramCode,
                    [SessionCode]=@SessionCode,
                    [SemesterCode]=@SemesterCode,
                    [LevelCode]=@LevelCode,
                    [AssessmentTypeCode]=@AssessmentTypeCode,
                    [SubCourseCode]=@SubCourseCode,
                    [StaffCode]=@StaffCode,
                    [Score]=@Score,
                    [GradeDescription]=@GradeDescription,
                    [GP]=@GP,
                    [Position]=@Position,
                    [StudentCode]=@StudentCode,
                    [MatricNo]=@MatricNo,
                    [Remark]=@Remark,
                    [ScreenCode]=@ScreenCode,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted

                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Assessment].[SPRecordsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Assessment].[SPRecordsDeletePermanently]    Script Date:↙
     08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Assessment.Records Table

CREATE PROC [Assessment].[SPRecordsDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Assessment].[Records] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Assessment].[Records]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);
```

```
END CATCH

GRANT EXECUTE ON [Assessment].[SPRecordsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Assessment].[SPRecordsDelete]   Script Date: 08/16/2011↙
     01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Assessment. ↙
    Records Table

CREATE PROC [Assessment].[SPRecordsDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Assessment].[Records] WHERE ([Code] = @Code) AND        ↙
    (Deleted=@Deleted))
            BEGIN
            UPDATE [Assessment].[Records]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);
```

```
END CATCH

GRANT EXECUTE ON [Assessment].[SPRecordsDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Academics].[SPRegistrationsSelect]    Script Date: 08/16↙
    /2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Academics.      ↙
    Registrations Table

CREATE PROC [Academics].[SPRegistrationsSelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Deleted BIT=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Academics].[Registrations] WHERE (([Code] = @Code) AND (↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted)))
            BEGIN
            SELECT * FROM [Academics].[Registrations]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Academics].[Registrations]
            WHERE
                (
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Academics].[SPRegistrationsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Academics].[SPRegistrationsInsertUpdate]    Script Date:↙
     08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Academics.Registrations   ↙
    Table

CREATE PROC [Academics].[SPRegistrationsInsertUpdate]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @FacultyCode BIGINT=NULL,
    @DepartmentCode BIGINT=NULL,
    @CourseCode BIGINT=NULL,
    @ProgramCode BIGINT=NULL,
    @LevelCode INT=NULL,
    @StudentCode BIGINT=NULL,
    @MatricNo NVARCHAR(50)=NULL,
    @SessionCode BIGINT=NULL,
    @SemesterCode BIGINT=NULL,
    @TypeCode NVARCHAR(50)=NULL,
    @SubCourseCode BIGINT=NULL,
    @Status NVARCHAR(50)=NULL,
    @TicketCode NUMERIC(18,0)=NULL,
    @PinCode NUMERIC(18,0)=NULL,
    @Signed BIT=NULL,
    @SignedBy NVARCHAR(50)=NULL,
    @SignedOn DATETIME2(7)=NULL,
    @SelfSigned BIT=NULL,
    @SelfSignedOn DATETIME2(7)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Academics].[Registrations] WHERE ([Code] = @Code)   ↙
    AND ([UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
                INSERT INTO [Academics].[Registrations]
                (
                    [UniversityCode],
                    [FacultyCode],
                    [DepartmentCode],
                    [CourseCode],
                    [ProgramCode],
                    [LevelCode],
                    [StudentCode],
                    [MatricNo],
                    [SessionCode],
                    [SemesterCode],
                    [TypeCode],
                    [SubCourseCode],
                    [Status],
                    [TicketCode],
                    [PinCode],
                    [Signed],
                    [SignedBy],
                    [SignedOn],
                    [SelfSigned],
                    [SelfSignedOn],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]

                )
                VALUES
                (
                    @UniversityCode,
                    @FacultyCode,
                    @DepartmentCode,
                    @CourseCode,
                    @ProgramCode,
                    @LevelCode,
                    @StudentCode,
                    @MatricNo,
                    @SessionCode,
                    @SemesterCode,
                    @TypeCode,
                    @SubCourseCode,
                    @Status,
                    @TicketCode,
                    @PinCode,
                    @Signed,
                    @SignedBy,
                    @SignedOn,
                    @SelfSigned,
                    @SelfSignedOn,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted

                )
            END
        ELSE
            BEGIN
                UPDATE [Academics].[Registrations]
                SET
                    [UniversityCode]=@UniversityCode,
                    [FacultyCode]=@FacultyCode,
                    [DepartmentCode]=@DepartmentCode,
                    [CourseCode]=@CourseCode,
                    [ProgramCode]=@ProgramCode,
                    [LevelCode]=@LevelCode,
                    [StudentCode]=@StudentCode,
                    [MatricNo]=@MatricNo,
                    [SessionCode]=@SessionCode,
                    [SemesterCode]=@SemesterCode,
                    [TypeCode]=@TypeCode,
                    [SubCourseCode]=@SubCourseCode,
                    [Status]=@Status,
                    [TicketCode]=@TicketCode,
                    [PinCode]=@PinCode,
                    [Signed]=@Signed,
                    [SignedBy]=@SignedBy,
                    [SignedOn]=@SignedOn,
                    [SelfSigned]=@SelfSigned,
                    [SelfSignedOn]=@SelfSignedOn,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted

                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Academics].[SPRegistrationsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Academics].[SPRegistrationsDeletePermanently]    Script ↙
    Date: 08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Academics.Registrations ↙
    Table

CREATE PROC [Academics].[SPRegistrationsDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Academics].[Registrations] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Academics].[Registrations]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Academics].[SPRegistrationsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Academics].[SPRegistrationsDelete]    Script Date: 08/16↙
    /2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Academics.  ↙
    Registrations Table

CREATE PROC [Academics].[SPRegistrationsDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Academics].[Registrations] WHERE ([Code] = @Code) AND   ↙
    (Deleted=@Deleted))
            BEGIN
            UPDATE [Academics].[Registrations]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Academics].[SPRegistrationsDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPLGAsSelect]    Script Date: 08/16/2011 01:24: ↙
    29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from SetUp.LGAs Table

CREATE PROC [SetUp].[SPLGAsSelect]
(
    @Code NVARCHAR(50)=NULL,
    @StatesCode NVARCHAR(50)=NULL,
    @CountriesCode NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[LGAs] WHERE ([Code] = @Code))
            BEGIN
            SELECT * FROM [SetUp].[LGAs]
            WHERE
                (
                ([Code] = @Code) AND
                ([StatesCode]=@StatesCode) AND
                ([CountriesCode]=@CountriesCode)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [SetUp].[LGAs]
            WHERE
                (
                ([StatesCode]=@StatesCode) AND
                ([CountriesCode]=@CountriesCode)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPLGAsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPLGAsInsertUpdate]    Script Date: 08/16/2011  ↙
    01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/14/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into SetUp.LGAs Table

CREATE PROC [SetUp].[SPLGAsInsertUpdate]
(
    @Code NVARCHAR(50)=NULL,
    @StatesCode NVARCHAR(50)=NULL,
    @CountriesCode NVARCHAR(50)=NULL,
    @LgName NVARCHAR(256)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[LGAs]  WHERE (([Code] = @Code) AND (       ↙
    [StatesCode] = @StatesCode) AND ([CountriesCode] = @CountriesCode)) )
            BEGIN
                INSERT INTO [SetUp].[LGAs]
                (
                    [Code],
                    [StatesCode],
                    [CountriesCode],
                    [LgName]
                )
                VALUES
                (
                    @Code,
                    @StatesCode,
                    @CountriesCode,
                    @LgName
                )
            END
        ELSE
            BEGIN
                UPDATE [SetUp].[LGAs]
                SET

                    [LgName]=@LgName
                WHERE
                    (([Code] = @Code) AND ([StatesCode]=@StatesCode) AND ([CountriesCode]=↙
    @CountriesCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPLGAsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPLGAsDeletePermanently]    Script Date: 08/16/ ↙
    2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/12/2011
--Last Updated      :   07/12/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from SetUp.LGAs Table

CREATE PROC [SetUp].[SPLGAsDeletePermanently]
(
    @Code NVARCHAR(50)=NULL,
    @StatesCode NVARCHAR(50)=NULL,
    @CountriesCode NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[LGAs] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [SetUp].[LGAs]
            WHERE
```

```
                (
                    ([Code] = @Code) AND
                    ([StatesCode]=@StatesCode) AND
                    ([CountriesCode]=@CountriesCode)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPLGAsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Academics].[SPFinalSignatoriesSelect]    Script Date: 08↙
    /16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Academics.          ↙
    FinalSignatories Table

CREATE PROC [Academics].[SPFinalSignatoriesSelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Deleted BIT=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Academics].[FinalSignatories] WHERE (([Code] = @Code)   ↙
    AND ([UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted)))
            BEGIN
            SELECT * FROM [Academics].[FinalSignatories]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Academics].[FinalSignatories]
            WHERE
                (
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Academics].[SPFinalSignatoriesSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Academics].[SPFinalSignatoriesInsertUpdate]    Script   ↙
    Date: 08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/18/2011
--Last Updated        :   07/18/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for saving values into Academics.FinalSignatories↙
     Table

CREATE PROC [Academics].[SPFinalSignatoriesInsertUpdate]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @FacultyCode BIGINT=NULL,
    @DepartmentCode BIGINT=NULL,
    @CourseCode BIGINT=NULL,
    @ProgramCode BIGINT=NULL,
    @LevelCode INT=NULL,
    @StudentCode BIGINT=NULL,
    @MatricNo NVARCHAR(50)=NULL,
    @SessionCode BIGINT=NULL,
    @SemesterCode BIGINT=NULL,
    @Signed BIT=NULL,
    @SignedBy NVARCHAR(50)=NULL,
    @SignedOn DATETIME2(7)=NULL,
    @DesignationCode NVARCHAR(50)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Academics].[FinalSignatories] WHERE ([Code] = @Code)↙
     AND ([UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
                INSERT INTO [Academics].[FinalSignatories]
                (
                    [UniversityCode],
                    [FacultyCode],
                    [DepartmentCode],
                    [CourseCode],
                    [ProgramCode],
                    [LevelCode],
                    [StudentCode],
                    [MatricNo],
                    [SessionCode],
                    [SemesterCode],
                    [Signed],
                    [SignedBy],
                    [SignedOn],
                    [DesignationCode],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]

                )
                VALUES
                (
                    @UniversityCode,
                    @FacultyCode,
                    @DepartmentCode,
                    @CourseCode,
                    @ProgramCode,
                    @LevelCode,
                    @StudentCode,
                    @MatricNo,
                    @SessionCode,
                    @SemesterCode,
                    @Signed,
                    @SignedBy,
                    @SignedOn,
                    @DesignationCode,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted

                )
            END
        ELSE
            BEGIN
                UPDATE [Academics].[FinalSignatories]
                SET
                    [UniversityCode]=@UniversityCode,
                    [FacultyCode]=@FacultyCode,
                    [DepartmentCode]=@DepartmentCode,
                    [CourseCode]=@CourseCode,
                    [ProgramCode]=@ProgramCode,
                    [LevelCode]=@LevelCode,
                    [StudentCode]=@StudentCode,
                    [MatricNo]=@MatricNo,
                    [SessionCode]=@SessionCode,
                    [SemesterCode]=@SemesterCode,
                    [Signed]=@Signed,
                    [SignedBy]=@SignedBy,
                    [SignedOn]=@SignedOn,
                    [DesignationCode]=@DesignationCode,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted

                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Academics].[SPFinalSignatoriesInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Academics].[SPFinalSignatoriesDeletePermanently]      ↙
    Script Date: 08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Academics.      ↙
    FinalSignatories Table

CREATE PROC [Academics].[SPFinalSignatoriesDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Academics].[FinalSignatories] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Academics].[FinalSignatories]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Academics].[SPFinalSignatoriesDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Academics].[SPFinalSignatoriesDelete]    Script Date: 08↙
    /16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/18/2011
--Last Updated        :   07/18/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for deleting values temporarily from Academics.  ↙
    FinalSignatories Table

CREATE PROC [Academics].[SPFinalSignatoriesDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Academics].[FinalSignatories] WHERE ([Code] = @Code) AND↙
     (Deleted=@Deleted))
            BEGIN
            UPDATE [Academics].[FinalSignatories]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Academics].[SPFinalSignatoriesDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPDesignationsSelect]    Script Date: 08/16/2011↙
     01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for retrieving values from SetUp.Designations   ↙
    Table

CREATE PROC [SetUp].[SPDesignationsSelect]
(
    @Code BIGINT=NULL,
    @StaffCode NVARCHAR(50)=NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Designations] WHERE ([Code] = @Code) AND (      ↙
    [StaffCode] = @StaffCode) AND ([Deleted] = @Deleted))
            BEGIN
            SELECT * FROM [SetUp].[Designations]
            WHERE
                (
                ([Code] = @Code) AND
                ([StaffCode] = @StaffCode) AND
                ([Deleted] = @Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [SetUp].[Designations]
            WHERE
                (
                ([StaffCode] = @StaffCode) AND
                ([Deleted] = @Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPDesignationsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPDesignationsInsertUpdate]    Script Date: 08/ ↙
    16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into SetUp.Designations Table

CREATE PROC [SetUp].[SPDesignationsInsertUpdate]
(
    @Code BIGINT=NULL,
    @StaffCode NVARCHAR(50)=NULL,
    @ComDate DATETIME2(7)=NULL,
    @CessDate DATETIME2(7)=NULL,
    @DesignationCode NVARCHAR(50)=NULL,
    @TypeCode NVARCHAR(50)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL

)
AS
BEGIN TRY
BEGIN TRAN
```

```
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[Designations]  WHERE ([Code] = @Code) )
            BEGIN
                INSERT INTO [SetUp].[Designations]
                (
                    [StaffCode],
                    [ComDate],
                    [CessDate],
                    [DesignationCode],
                    [TypeCode],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]

                )
                VALUES
                (
                    @StaffCode,
                    @ComDate,
                    @CessDate,
                    @DesignationCode,
                    @TypeCode,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [SetUp].[Designations]
                SET
                    [StaffCode]=@StaffCode,
                    [ComDate]=@ComDate,
                    [CessDate]=@CessDate,
                    [DesignationCode]=@DesignationCode,
                    [TypeCode]=@TypeCode,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    ([Code] = @Code)
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPDesignationsInsertUpdate] TO PUBLIC
```

```
GO
/****** Object:  StoredProcedure [SetUp].[SPDesignationsDeletePermanently]    Script Date:↙
     08/16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/12/2011
--Last Updated        :   07/12/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for deleting values from SetUp.Designations Table

CREATE PROC [SetUp].[SPDesignationsDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Designations] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [SetUp].[Designations]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPDesignationsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPDesignationsDelete]    Script Date: 08/16/2011↙
     01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/27/2011
--Last Updated        :   07/27/2011
```

```
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from SetUp.     ↙
    Designations Table

CREATE PROC [SetUp].[SPDesignationsDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[Designations] WHERE (([Code] = @Code) AND   ↙
    (Deleted=@Deleted)))
            BEGIN
            UPDATE [SetUp].[Designations]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPDesignationsDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPDepartmentsSelect]    Script Date: 08/16/2011 ↙
    01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
```

```
--Description       :   Stored procedure for retrieving values from SetUp.Departments     ↙
    Table

CREATE PROC [SetUp].[SPDepartmentsSelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @FacultyCode BIGINT=NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Departments] WHERE ([Code] = @Code) AND (       ↙
    [UniversityCode] = @UniversityCode) AND ([FacultyCode] = @FacultyCode) AND ([Deleted] ↙
    = @Deleted))
            BEGIN
            SELECT * FROM [SetUp].[Departments]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([FacultyCode] = @FacultyCode) AND
                ([Deleted] = @Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [SetUp].[Departments]
            WHERE
                (
                ([FacultyCode] = @FacultyCode) AND
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPDepartmentsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPDepartmentsInsertUpdate]    Script Date: 08/16↙
    /2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
```

```
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into SetUp.Departments Table

CREATE PROC [SetUp].[SPDepartmentsInsertUpdate]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @FacultyCode BIGINT=NULL,
    @Acronym NVARCHAR(50)=NULL,
    @Description NVARCHAR(500)=NULL,
    @Notes NVARCHAR(1000)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[Departments]  WHERE ([Code] = @Code) AND (  ↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
                INSERT INTO [SetUp].[Departments]
                (
                    [UniversityCode],
                    [FacultyCode],
                    [Acronym],
                    [Description],
                    [Notes],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]

                )
                VALUES
                (
                    @UniversityCode,
                    @FacultyCode,
                    @Acronym,
                    @Description,
                    @Notes,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [SetUp].[Departments]
                SET
                    [UniversityCode]=@UniversityCode,
                    [FacultyCode]=@FacultyCode,
                    [Acronym]=@Acronym,
                    [Description]=@Description,
                    [Notes]=@Notes,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPDepartmentsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPDepartmentsDeletePermanently]    Script Date: ↙
    08/16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/08/2011
--Last Updated        :   07/08/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for deleting values from SetUp.Departments Table

CREATE PROC [SetUp].[SPDepartmentsDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Departments] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [SetUp].[Departments]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN
```

```
DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPDepartmentsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPDepartmentsDelete]    Script Date: 08/16/2011 ↙
    01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/27/2011
--Last Updated      :   07/27/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from SetUp.     ↙
    Departments Table

CREATE PROC [SetUp].[SPDepartmentsDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[Departments] WHERE (([Code] = @Code) AND    ↙
    (Deleted=@Deleted)))
            BEGIN
            UPDATE [SetUp].[Departments]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN
```

```
DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPDepartmentsDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPCoursesSelect]    Script Date: 08/16/2011 01: ↙
    24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from SetUp.Courses Table

CREATE PROC [SetUp].[SPCoursesSelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @FacultyCode BIGINT=NULL,
    @DepartmentCode BIGINT=NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Courses] WHERE ([Code] = @Code) AND (          ↙
    [UniversityCode] = @UniversityCode) AND ([FacultyCode]=@FacultyCode) AND (          ↙
    [DepartmentCode]=@DepartmentCode) AND ([Deleted] = @Deleted))
            BEGIN
            SELECT * FROM [SetUp].[Courses]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([FacultyCode]=@FacultyCode) AND
                ([DepartmentCode]=@DepartmentCode) AND
                ([Deleted] = @Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [SetUp].[Courses]
            WHERE
                (
                ([UniversityCode] = @UniversityCode) AND
                ([FacultyCode]=@FacultyCode) AND
                ([DepartmentCode]=@DepartmentCode) AND
                ([Deleted] = @Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPCoursesSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPCoursesInsertUpdate]    Script Date: 08/16/   ↙
    2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/08/2011
--Last Updated      :   07/08/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into SetUp.Courses Table

CREATE PROC [SetUp].[SPCoursesInsertUpdate]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @FacultyCode BIGINT=NULL,
    @DepartmentCode BIGINT=NULL,
    @Acronym NVARCHAR(50)=NULL,
    @Description NVARCHAR(300)=NULL,
    @Notes NVARCHAR(1000)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[Courses]  WHERE ([Code] = @Code) AND (       ↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
```

```
                INSERT INTO [SetUp].[Courses]
                (
                    [UniversityCode],
                    [FacultyCode],
                    [DepartmentCode],
                    [Acronym],
                    [Description],
                    [Notes],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]

                )
                VALUES
                (
                    @UniversityCode,
                    @FacultyCode,
                    @DepartmentCode,
                    @Acronym,
                    @Description,
                    @Notes,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [SetUp].[Courses]
                SET
                    [UniversityCode]=@UniversityCode,
                    [FacultyCode]=@FacultyCode,
                    [DepartmentCode]=@DepartmentCode,
                    [Acronym]=@Acronym,
                    [Description]=@Description,
                    [Notes]=@Notes,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPCoursesInsertUpdate] TO PUBLIC
GO
```

```sql
/****** Object:  StoredProcedure [SetUp].[SPCoursesDeletePermanently]    Script Date: 08/ ↙
    16/2011 01:24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author             :   Ademola Adebo
--Reviewer           :   Godwin Mathias
--Date Created       :   07/08/2011
--Last Updated       :   07/08/2011
--Last Updated By    :   Ademola Adebo
--Description        :   Stored procedure for deleting values from SetUp.Courses Table

CREATE PROC [SetUp].[SPCoursesDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [SetUp].[Courses] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [SetUp].[Courses]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPCoursesDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [SetUp].[SPCoursesDelete]    Script Date: 08/16/2011 01: ↙
    24:29 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author             :   Ademola Adebo
--Reviewer           :   Godwin Mathias
--Date Created       :   07/27/2011
--Last Updated       :   07/27/2011
--Last Updated By    :   Ademola Adebo
```

```
--Description       :   Stored procedure for deleting values temporarily from SetUp.  ↙
    Courses Table

CREATE PROC [SetUp].[SPCoursesDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [SetUp].[Courses] WHERE (([Code] = @Code) AND    ↙
    (Deleted=@Deleted)))
            BEGIN
            UPDATE [SetUp].[Courses]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [SetUp].[SPCoursesDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Academics].[SPSessionsSelect]    Script Date: 08/16/2011↙
     01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Academics.Sessions    ↙
```

```
    Table

CREATE PROC [Academics].[SPSessionsSelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Deleted BIT=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Academics].[Sessions] WHERE (([Code] = @Code) AND ([UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted)))
            BEGIN
            SELECT * FROM [Academics].[Sessions]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Academics].[Sessions]
            WHERE
                (
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Academics].[SPSessionsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Academics].[SPSessionsInsertUpdate]    Script Date: 08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
```

```sql
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Academics.Sessions Table

CREATE PROC [Academics].[SPSessionsInsertUpdate]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @CurrentYear INT=NULL,
    @NextYear INT=NULL,
    @StartDate DATETIME2(7)=NULL,
    @EndDate DATETIME2(7)=NULL,
    @SessionStatus NVARCHAR(50)=NULL,
    @Applicable BIT=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Academics].[Sessions] WHERE ([Code] = @Code) AND (   ↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
                INSERT INTO [Academics].[Sessions]
                (
                    [UniversityCode],
                    [CurrentYear],
                    [NextYear],
                    [StartDate],
                    [EndDate],
                    [SessionStatus],
                    [Applicable],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]

                )
                VALUES
                (
                    @UniversityCode,
                    @CurrentYear,
                    @NextYear,
                    @StartDate,
                    @EndDate,
                    @SessionStatus,
                    @Applicable,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted

                )
            END
        ELSE
            BEGIN
                UPDATE [Academics].[Sessions]
                SET
                    [UniversityCode]=@UniversityCode,
                    [CurrentYear]=@CurrentYear,
                    [NextYear]=@NextYear,
                    [StartDate]=@StartDate,
                    [EndDate]=@EndDate,
                    [SessionStatus]=@SessionStatus,
                    [Applicable]=@Applicable,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted

                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Academics].[SPSessionsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Academics].[SPSessionsDeletePermanently]    Script Date:↙
     08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Academics.Sessions Table

CREATE PROC [Academics].[SPSessionsDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Academics].[Sessions] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Academics].[Sessions]
            WHERE
                (
                    ([Code] = @Code)
                )
            END
```

```
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Academics].[SPSessionsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Academics].[SPSessionsDelete]    Script Date: 08/16/2011↙
     01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Academics.  ↙
    Sessions Table

CREATE PROC [Academics].[SPSessionsDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Academics].[Sessions] WHERE ([Code] = @Code) AND        ↙
    (Deleted=@Deleted))
            BEGIN
            UPDATE [Academics].[Sessions]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
```

```
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Academics].[SPSessionsDelete] TO PUBLIC
GO
/****** Object:  Table [Finance].[Transactions]    Script Date: 08/16/2011 01:24:32 ******↙
    /
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Finance].[Transactions](
    [Code] [bigint] NOT NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [SessionCode] [bigint] NULL,
    [SemesterCode] [nvarchar](50) NULL,
    [TransactionActivityCode] [nvarchar](50) NULL,
    [InvoiceTypeCode] [nvarchar](50) NULL,
    [LpoNo] [decimal](18, 0) NULL,
    [PoNo] [decimal](18, 0) NULL,
    [CategoryCode] [nvarchar](50) NULL,
    [Amount] [decimal](18, 0) NULL,
    [AmountPaid] [decimal](18, 0) NULL,
    [Balance] [decimal](18, 0) NULL,
    [Total] [decimal](18, 0) NULL,
    [CompanyCode] [nvarchar](50) NULL,
    [PaymentStatus] [nvarchar](50) NULL,
    [DateX] [datetime2](7) NULL,
    [AmountInWords] [nvarchar](500) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Invoices] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,           ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Registry].[Tickets]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
```

```
CREATE TABLE [Registry].[Tickets](
    [Code] [numeric](18, 0) NOT NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [ScreenCode] [nvarchar](50) NULL,
    [SessionCode] [bigint] NULL,
    [SemesterCode] [nvarchar](50) NULL,
    [PinCode] [numeric](18, 0) NOT NULL,
    [DateX] [datetime2](7) NULL,
    [StatusCode] [nvarchar](50) NULL,
    [NoOfUse] [int] NULL,
    [MaxNoUse] [int] NULL,
    [ExpiryDate] [datetime2](7) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Tickets] PRIMARY KEY CLUSTERED
(
    [Code] ASC,
    [PinCode] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,          ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Finance].[Vouchers]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Finance].[Vouchers](
    [Code] [bigint] IDENTITY(1,1) NOT NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [SessionCode] [bigint] NULL,
    [SemesterCode] [nvarchar](50) NULL,
    [AccountGroupCode] [nvarchar](50) NULL,
    [AccountSubCode] [int] NULL,
    [AccountTypeCode] [decimal](18, 0) NULL,
    [AccountCode] [decimal](18, 0) NULL,
    [VoucherTypeCode] [nvarchar](50) NULL,
    [TransactionTypeCode] [nvarchar](50) NULL,
    [EntityTypeCode] [nvarchar](50) NULL,
    [ParticularsCode] [nvarchar](50) NULL,
    [DateX] [datetime] NULL,
    [DebitAmount] [decimal](18, 2) NULL,
    [CreditAmount] [decimal](18, 2) NULL,
    [Narration] [nvarchar](256) NULL,
    [Approved] [bit] NULL,
    [ApproveBy] [nvarchar](50) NULL,
    [ApproveOn] [datetime2](7) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
    [Notes] [nvarchar](256) NULL,
 CONSTRAINT [PK_Vouchers] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,          ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
```

```
/****** Object:  StoredProcedure [dbo].[aspnet_Profile_DeleteProfiles]    Script Date: 08/↙
    16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER OFF
GO
CREATE PROCEDURE [dbo].[aspnet_Profile_DeleteProfiles]
    @ApplicationName        nvarchar(256),
    @UserNames              nvarchar(4000)
AS
BEGIN
    DECLARE @UserName     nvarchar(256)
    DECLARE @CurrentPos   int
    DECLARE @NextPos      int
    DECLARE @NumDeleted   int
    DECLARE @DeletedUser  int
    DECLARE @TranStarted  bit
    DECLARE @ErrorCode    int

    SET @ErrorCode = 0
    SET @CurrentPos = 1
    SET @NumDeleted = 0
    SET @TranStarted = 0

    IF( @@TRANCOUNT = 0 )
    BEGIN
        BEGIN TRANSACTION
        SET @TranStarted = 1
    END
    ELSE
        SET @TranStarted = 0

    WHILE (@CurrentPos <= LEN(@UserNames))
    BEGIN
        SELECT @NextPos = CHARINDEX(N',', @UserNames,  @CurrentPos)
        IF (@NextPos = 0 OR @NextPos IS NULL)
            SELECT @NextPos = LEN(@UserNames) + 1

        SELECT @UserName = SUBSTRING(@UserNames, @CurrentPos, @NextPos - @CurrentPos)
        SELECT @CurrentPos = @NextPos+1

        IF (LEN(@UserName) > 0)
        BEGIN
            SELECT @DeletedUser = 0
            EXEC dbo.aspnet_Users_DeleteUser @ApplicationName, @UserName, 4, @DeletedUser ↙
    OUTPUT
            IF( @@ERROR <> 0 )
            BEGIN
                SET @ErrorCode = -1
                GOTO Cleanup
            END
            IF (@DeletedUser <> 0)
                SELECT @NumDeleted = @NumDeleted + 1
        END
    END
    SELECT @NumDeleted
    IF (@TranStarted = 1)
    BEGIN
        SET @TranStarted = 0
        COMMIT TRANSACTION
    END
    SET @TranStarted = 0

    RETURN 0

Cleanup:
    IF (@TranStarted = 1 )
    BEGIN
        SET @TranStarted = 0
        ROLLBACK TRANSACTION
    END
    RETURN @ErrorCode
END
GO
/****** Object:  Table [Registry].[Admissions]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Registry].[Admissions](
    [Code] [bigint] NOT NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [AccountCode] [nvarchar](50) NULL,
    [FacultyCode] [bigint] NULL,
    [DepartmentCode] [bigint] NULL,
    [CourseCode] [bigint] NULL,
    [ProgramCode] [bigint] NULL,
    [AwardCode] [nvarchar](50) NULL,
    [LevelCode] [int] NULL,
    [EntryMode] [nvarchar](50) NULL,
    [StudyMode] [nvarchar](50) NULL,
    [SessionCode] [bigint] NULL,
    [Score] [int] NULL,
    [JambScore] [int] NULL,
    [CutOffMax] [int] NULL,
    [AdmissionStatus] [nvarchar](50) NULL,
    [StatusReason] [nvarchar](500) NULL,
    [TotalScore] [int] NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Admissions] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF, ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Finance].[FeeAllotment]    Script Date: 08/16/2011 01:24:32 ******↙
    /
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Finance].[FeeAllotment](
    [Code] [bigint] IDENTITY(1,1) NOT NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [FacultyCode] [bigint] NULL,
    [DepartmentCode] [bigint] NULL,
    [CourseCode] [bigint] NULL,
    [ProgramCode] [bigint] NULL,
    [LevelCode] [int] NULL,
    [FeeDefinitionCode] [nvarchar](50) NULL,
    [SessionCode] [bigint] NULL,
    [SemesterCode] [nvarchar](50) NULL,
    [EntryMode] [nvarchar](50) NULL,
    [StudyMode] [nvarchar](50) NULL,
    [Amount] [decimal](18, 2) NULL,
    [DiscountPercentage] [int] NULL,
    [DiscountedAmount]  AS (([DiscountPercentage]/(100))*[Amount]+[Amount]),
    [Approved] [bit] NULL,
    [ApproveBy] [nvarchar](50) NULL,
    [ApproveOn] [datetime2](7) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
    [Notes] [nvarchar](256) NULL,
 CONSTRAINT [PK_FeeAllotment] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,      ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Finance].[Fees]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Finance].[Fees](
    [Code] [bigint] NOT NULL,
    [MatricNo] [nvarchar](50) NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [FacultyCode] [bigint] NULL,
    [DepartmentCode] [bigint] NULL,
    [CourseCode] [bigint] NULL,
    [ProgramCode] [bigint] NULL,
    [LevelCode] [int] NULL,
    [SessionCode] [bigint] NULL,
    [SemesterCode] [nvarchar](50) NULL,
    [AccountTitle] [nvarchar](256) NULL,
    [AccountNo] [nvarchar](50) NULL,
    [BankCode] [nvarchar](50) NULL,
    [BankBranch] [nvarchar](256) NULL,
    [PaymentMode] [nvarchar](50) NULL,
    [DateX] [datetime2](7) NULL,
    [Amount] [decimal](18, 2) NULL,
    [Balance] [decimal](18, 2) NULL,
    [PaymentStatus] [nvarchar](50) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Fees] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,      ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
EXEC sys.sp_addextendedproperty @name=N'MS_Description', @value=N'Bank|Cash|Cheque|ATM' , ↙
    @level0type=N'SCHEMA',@level0name=N'Finance', @level1type=N'TABLE',@level1name=N'Fees'↙
    , @level2type=N'COLUMN',@level2name=N'PaymentMode'
GO
/****** Object:  Table [Academics].[SigningHierarchy]    Script Date: 08/16/2011 01:24:32 ↙
    ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
```

```
CREATE TABLE [Academics].[SigningHierarchy](
    [Code] [bigint] IDENTITY(1,1) NOT NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [DesignationCode] [nvarchar](50) NULL,
    [OrderCode] [nvarchar](50) NULL,
    [SessionCode] [bigint] NULL,
    [Applicable] [bit] NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_SigningHierarchy] PRIMARY KEY CLUSTERED 
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,       ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Finance].[Refunds]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Finance].[Refunds](
    [Code] [bigint] IDENTITY(1,1) NOT NULL,
    [MatricNo] [nvarchar](50) NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [FacultyCode] [bigint] NULL,
    [DepartmentCode] [bigint] NULL,
    [CourseCode] [bigint] NULL,
    [ProgramCode] [bigint] NULL,
    [LevelCode] [int] NULL,
    [FeesCode] [decimal](18, 0) NULL,
    [SessionCode] [bigint] NULL,
    [SemesterCode] [nvarchar](50) NULL,
    [Amount] [decimal](18, 2) NULL,
    [Refunded] [bit] NULL,
    [RefundedOn] [datetime2](7) NULL,
    [RefundedBy] [nvarchar](50) NULL,
    [Approved] [bit] NULL,
    [ApproveBy] [nvarchar](50) NULL,
    [ApproveOn] [datetime2](7) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_Refunds] PRIMARY KEY CLUSTERED 
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,       ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Finance].[PaymentPlans]    Script Date: 08/16/2011 01:24:32 ******↙
    /
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Finance].[PaymentPlans](
    [Code] [bigint] IDENTITY(1,1) NOT NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [SessionCode] [bigint] NULL,
    [SemesterCode] [nvarchar](50) NULL,
    [NoOfPaymentAllowed] [int] NULL,
    [DateX] [datetime2](7) NULL,
    [StatusCode] [nvarchar](50) NULL,
    [Applicable] [bit] NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_PaymentPlans] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,                ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Finance].[PaymentPlanDetails]    Script Date: 08/16/2011 01:24:32 ↙
    ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Finance].[PaymentPlanDetails](
    [Code] [bigint] NOT NULL,
    [PaymentPlanCode] [bigint] NULL,
    [Description] [nvarchar](256) NULL,
    [Amount] [decimal](18, 2) NULL,
    [DiscountedAmount] [decimal](18, 2) NULL,
    [DiscountedPercentage] [int] NULL,
    [Total] [decimal](18, 2) NULL,
    [CreatedOn] [datetime2](7) NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime2](7) NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_PaymentPlanDetails] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,                ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [Finance].[FeesDetail]    Script Date: 08/16/2011 01:24:32 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [Finance].[FeesDetail](
    [Code] [bigint] IDENTITY(1,1) NOT NULL,
    [MatricNo] [nvarchar](50) NULL,
    [UniversityCode] [nvarchar](50) NULL,
    [FacultyCode] [bigint] NULL,
    [DepartmentCode] [bigint] NULL,
    [CourseCode] [bigint] NULL,
    [ProgramCode] [bigint] NULL,
    [LevelCode] [int] NULL,
    [FeesCode] [bigint] NULL,
    [FeeDefinitionCode] [nvarchar](50) NULL,
    [FeeAllotmentCode] [bigint] NULL,
    [Amount] [decimal](18, 2) NULL,
    [ExtraDiscount] [int] NULL,
    [FinalAmount] [decimal](18, 2) NULL,
    [CreatedOn] [datetime] NULL,
    [CreatedBy] [nvarchar](50) NULL,
    [ModifiedOn] [datetime] NULL,
    [ModifiedBy] [nvarchar](50) NULL,
    [Deleted] [bit] NULL,
    [DeletedOn] [datetime2](7) NULL,
    [DeletedBy] [nvarchar](50) NULL,
 CONSTRAINT [PK_FeesDetail] PRIMARY KEY CLUSTERED
(
    [Code] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF,        ↙
    ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  StoredProcedure [Finance].[SPFeesDeletePermanently]    Script Date: 08/16↙
    /2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Finance.Fees Table

CREATE PROC [Finance].[SPFeesDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance].[Fees] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Finance].[Fees]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);
```

```
END CATCH

GRANT EXECUTE ON [Finance].[SPFeesDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPFeesDelete]    Script Date: 08/16/2011 01:24↙
    :28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author             :   Ademola Adebo
--Reviewer           :   Godwin Mathias
--Date Created       :   07/18/2011
--Last Updated       :   07/18/2011
--Last Updated By    :   Ademola Adebo
--Description        :   Stored procedure for deleting values temporarily from Finance.Fees↙
     Table

CREATE PROC [Finance].[SPFeesDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance].[Fees] WHERE ([Code] = @Code) AND (Deleted=@   ↙
    Deleted))
            BEGIN
            UPDATE [Finance].[Fees]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);
```

```
END CATCH

GRANT EXECUTE ON [Finance].[SPFeesDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPVouchersSelect]    Script Date: 08/16/2011  ↙
    01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/18/2011
--Last Updated        :   07/18/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for retrieving values from Finance.Vouchers Table

CREATE PROC [Finance].[SPVouchersSelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Deleted BIT=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance].[Vouchers] WHERE (([Code] = @Code) AND (        ↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted)))
            BEGIN
            SELECT * FROM [Finance].[Vouchers]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Finance].[Vouchers]
            WHERE
                (
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()
```

```
RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPVouchersSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPVouchersInsertUpdate]    Script Date: 08/16/↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/18/2011
--Last Updated        :   07/18/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for saving values into Finance.Vouchers Table

CREATE PROC [Finance].[SPVouchersInsertUpdate]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @SessionCode BIGINT=NULL,
    @SemesterCode NVARCHAR(50)=NULL,
    @AccountGroupCode NVARCHAR(50)=NULL,
    @AccountSubCode INT=NULL,
    @AccountTypeCode DECIMAL(18,0)=NULL,
    @AccountCode DECIMAL(18,0)=NULL,
    @VoucherTypeCode NVARCHAR(50)=NULL,
    @TransactionTypeCode NVARCHAR(50)=NULL,
    @EntityTypeCode NVARCHAR(50)=NULL,
    @ParticularsCode NVARCHAR(50)=NULL,
    @DateX DATETIME=NULL,
    @DebitAmount DECIMAL(18,2)=NULL,
    @CreditAmount DECIMAL(18,2)=NULL,
    @Narration NVARCHAR(256)=NULL,
    @Approved BIT=NULL,
    @ApproveBy NVARCHAR(50)=NULL,
    @ApproveOn DATETIME2(7)=NULL,
    @Notes NVARCHAR(256)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Finance] .[Vouchers] WHERE ([Code] = @Code) AND (   ↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
                INSERT INTO [Finance] .[Vouchers]
                (
                    [UniversityCode],
                    [SessionCode],
                    [SemesterCode],
                    [AccountGroupCode],
                    [AccountSubCode],
                    [AccountTypeCode],
                    [AccountCode],
```

```sql
                    [VoucherTypeCode],
                    [TransactionTypeCode],
                    [EntityTypeCode],
                    [ParticularsCode],
                    [DateX],
                    [DebitAmount],
                    [CreditAmount],
                    [Narration],
                    [Approved],
                    [ApproveBy],
                    [ApproveOn],
                    [Notes],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]

                )
                VALUES
                (
                    @UniversityCode,
                    @SessionCode,
                    @SemesterCode,
                    @AccountGroupCode,
                    @AccountSubCode,
                    @AccountTypeCode,
                    @AccountCode,
                    @VoucherTypeCode,
                    @TransactionTypeCode,
                    @EntityTypeCode,
                    @ParticularsCode,
                    @DateX,
                    @DebitAmount,
                    @CreditAmount,
                    @Narration,
                    @Approved,
                    @ApproveBy,
                    @ApproveOn,
                    @Notes,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted

                )
            END
        ELSE
            BEGIN
                UPDATE [Finance] .[Vouchers]
                SET
                    [UniversityCode]=@UniversityCode,
                    [SessionCode]=@SessionCode,
                    [SemesterCode]=@SemesterCode,
                    [AccountGroupCode]=@AccountGroupCode,
                    [AccountSubCode]=@AccountSubCode,
                    [AccountTypeCode]=@AccountTypeCode,
                    [AccountCode]=@AccountCode,
                    [VoucherTypeCode]=@VoucherTypeCode,
                    [TransactionTypeCode]=@TransactionTypeCode,
                    [EntityTypeCode]=@EntityTypeCode,
                    [ParticularsCode]=@ParticularsCode,
                    [DateX]=@DateX,
                    [DebitAmount]=@DebitAmount,
                    [CreditAmount]=@CreditAmount,
                    [Narration]=@Narration,
                    [Approved]=@Approved,
                    [ApproveBy]=@ApproveBy,
                    [ApproveOn]=@ApproveOn,
                    [Notes]=@Notes,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted

                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPVouchersInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPVouchersDeletePermanently]    Script Date:  ↙
    08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Finance.Vouchers Table

CREATE PROC [Finance].[SPVouchersDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance].[Vouchers] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Finance].[Vouchers]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
```

```sql
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPVouchersDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPVouchersDelete]    Script Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Finance.Vouchers Table

CREATE PROC [Finance].[SPVouchersDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance].[Vouchers] WHERE ([Code] = @Code) AND (Deleted=@Deleted))
            BEGIN
            UPDATE [Finance].[Vouchers]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN
```

```
DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPVouchersDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPTransactionsSelect]    Script Date: 08/16/  ↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Finance.Transactions  ↙
    Table

CREATE PROC [Finance].[SPTransactionsSelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Deleted BIT=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance].[Transactions] WHERE (([Code] = @Code) AND (   ↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted)))
            BEGIN
            SELECT * FROM [Finance].[Transactions]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Finance].[Transactions]
            WHERE
                (
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
COMMIT TRAN
```

```
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPTransactionsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPTransactionsInsertUpdate]    Script Date: 08↙
    /16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Finance.Transactions Table

CREATE PROC [Finance].[SPTransactionsInsertUpdate]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @SessionCode BIGINT=NULL,
    @SemesterCode NVARCHAR(50)=NULL,
    @TransactionActivityCode NVARCHAR(50)=NULL,
    @InvoiceTypeCode NVARCHAR(50)=NULL,
    @LpoNo DECIMAL(18,0)=NULL,
    @PoNo DECIMAL(18,0)=NULL,
    @CategoryCode NVARCHAR(50)=NULL,
    @Amount DECIMAL(18,0)=NULL,
    @AmountPaid DECIMAL(18,0)=NULL,
    @Balance DECIMAL(18,0)=NULL,
    @Total DECIMAL(18,0)=NULL,
    @CompanyCode NVARCHAR(50)=NULL,
    @PaymentStatus NVARCHAR(50)=NULL,
    @DateX DATETIME2(7)=NULL,
    @AmountInWords NVARCHAR(500)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
```

```
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Finance] .[Transactions] WHERE ([Code] = @Code) AND ↙
    ([UniversityCode] = @UniversityCode))
            BEGIN
                INSERT INTO [Finance] .[Transactions]
                (
                    [UniversityCode],
                    [SessionCode],
                    [SemesterCode],
                    [TransactionActivityCode],
                    [InvoiceTypeCode],
                    [LpoNo],
                    [PoNo],
                    [CategoryCode],
                    [Amount],
                    [AmountPaid],
                    [Balance],
                    [Total],
                    [CompanyCode],
                    [PaymentStatus],
                    [DateX],
                    [AmountInWords],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]

                )
                VALUES
                (
                    @UniversityCode,
                    @SessionCode,
                    @SemesterCode,
                    @TransactionActivityCode,
                    @InvoiceTypeCode,
                    @LpoNo,
                    @PoNo,
                    @CategoryCode,
                    @Amount,
                    @AmountPaid,
                    @Balance,
                    @Total,
                    @CompanyCode,
                    @PaymentStatus,
                    @DateX,
                    @AmountInWords,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted

                )
            END
        ELSE
            BEGIN
                UPDATE [Finance] .[Transactions]
                SET
                    [UniversityCode]=@UniversityCode,
                    [SessionCode]=@SessionCode,
                    [SemesterCode]=@SemesterCode,
                    [TransactionActivityCode]=@TransactionActivityCode,
                    [InvoiceTypeCode]=@InvoiceTypeCode,
                    [LpoNo]=@LpoNo,
                    [PoNo]=@PoNo,
                    [CategoryCode]=@CategoryCode,
                    [Amount]=@Amount,
                    [AmountPaid]=@AmountPaid,
                    [Balance]=@Balance,
                    [Total]=@Total,
                    [CompanyCode]=@CompanyCode,
                    [PaymentStatus]=@PaymentStatus,
                    [DateX]=@DateX,
                    [AmountInWords]=@AmountInWords,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted

                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPTransactionsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPTransactionsDeletePermanently]    Script    ↙
    Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Finance.Transactions    ↙
    Table

CREATE PROC [Finance].[SPTransactionsDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance].[Transactions] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Finance].[Transactions]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPTransactionsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPTransactionsDelete]    Script Date: 08/16/ ↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Finance. ↙
    Transactions Table

CREATE PROC [Finance].[SPTransactionsDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance].[Transactions] WHERE ([Code] = @Code) AND ↙
    (Deleted=@Deleted))
            BEGIN
            UPDATE [Finance].[Transactions]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END
```

```
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPTransactionsDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Registry].[SPTicketsSelect]    Script Date: 08/16/2011  ↙
    01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Registry.Tickets Table

CREATE PROC [Registry].[SPTicketsSelect]
(
    @Code NUMERIC(18,0)=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @PinCode NUMERIC(18,0)=NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Registry].[Tickets] WHERE (([Code] = @Code) AND (       ↙
    [UniversityCode]=@UniversityCode) AND ([PinCode] = @PinCode) AND ([Deleted]=@         ↙
    Deleted)))
            BEGIN
            SELECT * FROM [Registry].[Tickets]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([PinCode] = @PinCode) AND
                ([Deleted]=@Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Registry].[Tickets]
            WHERE
                (
                ([UniversityCode]=@UniversityCode) AND
                ([PinCode]=@PinCode) AND
                ([Deleted]=@Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Registry].[SPTicketsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Registry].[SPTicketsInsertUpdate]    Script Date: 08/16/↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Registry.Tickets Table

CREATE PROC [Registry].[SPTicketsInsertUpdate]
(
    @Code NUMERIC(18,0)=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @ScreenCode NVARCHAR(50)=NULL,
    @SessionCode BIGINT=NULL,
    @SemesterCode NVARCHAR(50)=NULL,
    @PinCode NUMERIC(18,0)=NULL,
    @DateX DATETIME2(7)=NULL,
    @StatusCode NVARCHAR(50)=NULL,
    @NoOfUse INT=NULL,
    @MaxNoUse INT=NULL,
    @ExpiryDate DATETIME2(7)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
```

```
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Registry].[Tickets] WHERE ([Code] = @Code) AND (    ↙
    [UniversityCode]=@UniversityCode) AND ([PinCode] = @PinCode) AND ([Deleted]=@Deleted))
            BEGIN
                INSERT INTO [Registry].[Tickets]
                (
                    [UniversityCode],
                    [ScreenCode],
                    [SessionCode],
                    [SemesterCode],
                    [PinCode],
                    [DateX],
                    [StatusCode],
                    [NoOfUse],
                    [MaxNoUse],
                    [ExpiryDate],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    @UniversityCode,
                    @ScreenCode,
                    @SessionCode,
                    @SemesterCode,
                    @PinCode,
                    @DateX,
                    @StatusCode,
                    @NoOfUse,
                    @MaxNoUse,
                    @ExpiryDate,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [Registry].[Tickets]
                SET
                    [UniversityCode]=@UniversityCode,
                    [ScreenCode]=@ScreenCode,
                    [SessionCode]=@SessionCode,
                    [SemesterCode]=@SemesterCode,
                    [PinCode]=@PinCode,
                    [DateX]=@DateX,
                    [StatusCode]=@StatusCode,
                    [NoOfUse]=@NoOfUse,
                    [MaxNoUse]=@MaxNoUse,
                    [ExpiryDate]=@ExpiryDate,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode) AND (       ↙
    [PinCode] = @PinCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN
```

```
DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Registry].[SPTicketsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Registry].[SPTicketsDeletePermanently]    Script Date:  ↙
    08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author             :   Ademola Adebo
--Reviewer           :   Godwin Mathias
--Date Created       :   07/18/2011
--Last Updated       :   07/18/2011
--Last Updated By    :   Ademola Adebo
--Description        :   Stored procedure for deleting values from Registry.Tickets Table

CREATE PROC [Registry].[SPTicketsDeletePermanently]
(
    @Code NUMERIC(18,0)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Registry].[Tickets] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Registry].[Tickets]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Registry].[SPTicketsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Registry].[SPTicketsDelete]    Script Date: 08/16/2011  ↙
    01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Registry.  ↙
    Tickets Table

CREATE PROC [Registry].[SPTicketsDelete]
(
    @Code NUMERIC(18,0)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Registry].[Tickets] WHERE ([Code] = @Code) AND (Deleted=↙
    @Deleted))
            BEGIN
            UPDATE [Registry].[Tickets]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()
```

```
RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Registry].[SPTicketsDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Academics].[SPSigningHierarchySelect]    Script Date: 08↙
    /16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/18/2011
--Last Updated        :   07/18/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for retrieving values from Academics.          ↙
    SigningHierarchy Table

CREATE PROC [Academics].[SPSigningHierarchySelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Deleted BIT=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Academics].[SigningHierarchy] WHERE (([Code] = @Code)   ↙
    AND ([UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted)))
            BEGIN
            SELECT * FROM [Academics].[SigningHierarchy]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Academics].[SigningHierarchy]
            WHERE
                (
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Academics].[SPSigningHierarchySelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Academics].[SPSigningHierarchyInsertUpdate]    Script   ↙
    Date: 08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/18/2011
--Last Updated        :   07/18/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for saving values into Academics.SigningHierarchy↙
     Table

CREATE PROC [Academics].[SPSigningHierarchyInsertUpdate]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @DesignationCode NVARCHAR(50)=NULL,
    @OrderCode NVARCHAR(50)=NULL,
    @SessionCode BIGINT=NULL,
    @Applicable BIT=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Academics].[SigningHierarchy] WHERE ([Code] = @Code)↙
     AND ([UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
                INSERT INTO [Academics].[SigningHierarchy]
                (
                    [UniversityCode],
                    [DesignationCode],
                    [OrderCode],
                    [SessionCode],
                    [Applicable],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]

                )
                VALUES
                (
                    @UniversityCode,
                    @DesignationCode,
                    @OrderCode,
                    @SessionCode,
                    @Applicable,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted

                )
            END
        ELSE
            BEGIN
                UPDATE [Academics].[SigningHierarchy]
                SET
                    [UniversityCode]=@UniversityCode,
                    [DesignationCode]=@DesignationCode,
                    [OrderCode]=@OrderCode,
                    [SessionCode]=@SessionCode,
                    [Applicable]=@Applicable,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted

                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Academics].[SPSigningHierarchyInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Academics].[SPSigningHierarchyDeletePermanently]        ↙
    Script Date: 08/16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Academics.             ↙
    SigningHierarchy Table

CREATE PROC [Academics].[SPSigningHierarchyDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
```

```
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Academics].[SigningHierarchy] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Academics].[SigningHierarchy]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Academics].[SPSigningHierarchyDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Academics].[SPSigningHierarchyDelete]    Script Date: 08↙
    /16/2011 01:24:27 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/18/2011
--Last Updated        :   07/18/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for deleting values temporarily from Academics.  ↙
    SigningHierarchy Table

CREATE PROC [Academics].[SPSigningHierarchyDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Academics].[SigningHierarchy] WHERE ([Code] = @Code) AND↙
     (Deleted=@Deleted))
            BEGIN
            UPDATE [Academics].[SigningHierarchy]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Academics].[SPSigningHierarchyDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPRefundsSelect]    Script Date: 08/16/2011 01↙
    :24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Finance.Refunds Table

CREATE PROC [Finance].[SPRefundsSelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Deleted BIT=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance].[Refunds] WHERE (([Code] = @Code) AND (       ↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted)))
            BEGIN
            SELECT * FROM [Finance].[Refunds]
            WHERE
```

```
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Finance].[Refunds]
            WHERE
                (
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPRefundsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPRefundsInsertUpdate]    Script Date: 08/16/ ↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Finance.Refunds Table

CREATE PROC [Finance].[SPRefundsInsertUpdate]
(
    @Code BIGINT=NULL,
    @MatricNo NVARCHAR(50)=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @FacultyCode BIGINT=NULL,
    @DepartmentCode BIGINT=NULL,
    @CourseCode BIGINT=NULL,
    @ProgramCode BIGINT=NULL,
    @LevelCode INT=NULL,
    @FeesCode DECIMAL(18,0)=NULL,
    @SessionCode BIGINT=NULL,
    @SemesterCode NVARCHAR(50)=NULL,
    @Amount DECIMAL(18,2)=NULL,
    @Refunded BIT=NULL,
    @RefundedOn DATETIME2(7)=NULL,
    @RefundedBy NVARCHAR(50)=NULL,
    @Approved BIT=NULL,
    @ApproveBy NVARCHAR(50)=NULL,
    @ApproveOn DATETIME2(7)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Finance] .[Refunds] WHERE ([Code] = @Code) AND (     ↙
    [UniversityCode] = @UniversityCode))
            BEGIN
                INSERT INTO [Finance] .[Refunds]
                (
                    [MatricNo],
                    [UniversityCode],
                    [FacultyCode],
                    [DepartmentCode],
                    [CourseCode],
                    [ProgramCode],
                    [LevelCode],
                    [FeesCode],
                    [SessionCode],
                    [SemesterCode],
                    [Amount],
                    [Refunded],
                    [RefundedOn],
                    [RefundedBy],
                    [Approved],
                    [ApproveBy],
                    [ApproveOn],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]

                )
                VALUES
                (
                    @MatricNo,
                    @UniversityCode,
                    @FacultyCode,
                    @DepartmentCode,
                    @CourseCode,
                    @ProgramCode,
                    @LevelCode,
                    @FeesCode,
                    @SessionCode,
                    @SemesterCode,
                    @Amount,
                    @Refunded,
                    @RefundedOn,
                    @RefundedBy,
                    @Approved,
                    @ApproveBy,
                    @ApproveOn,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted

                )
            END
        ELSE
            BEGIN
                UPDATE [Finance] .[Refunds]
                SET
                    [MatricNo]=@MatricNo,
                    [UniversityCode]=@UniversityCode,
                    [FacultyCode]=@FacultyCode,
                    [DepartmentCode]=@DepartmentCode,
                    [CourseCode]=@CourseCode,
                    [ProgramCode]=@ProgramCode,
                    [LevelCode]=@LevelCode,
                    [FeesCode]=@FeesCode,
                    [SessionCode]=@SessionCode,
                    [SemesterCode]=@SemesterCode,
                    [Amount]=@Amount,
                    [Refunded]=@Refunded,
                    [RefundedOn]=@RefundedOn,
                    [RefundedBy]=@RefundedBy,
                    [Approved]=@Approved,
                    [ApproveBy]=@ApproveBy,
                    [ApproveOn]=@ApproveOn,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted

                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPRefundsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPRefundsDeletePermanently]    Script Date: 08↙
    /16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
```

```sql
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Finance.Refunds Table

CREATE PROC [Finance].[SPRefundsDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance].[Refunds] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Finance].[Refunds]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPRefundsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPRefundsDelete]    Script Date: 08/16/2011 01↙
    :24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Finance.    ↙
    Refunds Table

CREATE PROC [Finance].[SPRefundsDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
```

```
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance].[Refunds] WHERE ([Code] = @Code) AND (Deleted=@↙
    Deleted))
            BEGIN
            UPDATE [Finance].[Refunds]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPRefundsDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Registry].[SPAdmissionsSelect]    Script Date: 08/16/   ↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Registry.Admissions   ↙
    Table

CREATE PROC [Registry].[SPAdmissionsSelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
```

```sql
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Registry].[Admissions] WHERE (([Code] = @Code) AND (    ↙
    [UniversityCode]=@UniversityCode) AND ([Deleted]=@Deleted)))
            BEGIN
            SELECT * FROM [Registry].[Admissions]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted]=@Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Registry].[Admissions]
            WHERE
                (
                ([UniversityCode]=@UniversityCode) AND
                ([Deleted]=@Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Registry].[SPAdmissionsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Registry].[SPAdmissionsInsertUpdate]    Script Date: 08/↙
    16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Registry.Admissions Table

CREATE PROC [Registry].[SPAdmissionsInsertUpdate]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @AccountCode NVARCHAR(50)=NULL,
    @FacultyCode BIGINT=NULL,
    @DepartmentCode BIGINT=NULL,
    @CourseCode BIGINT=NULL,
    @ProgramCode BIGINT=NULL,
    @AwardCode NVARCHAR(50)=NULL,
    @LevelCode INT=NULL,
    @EntryMode NVARCHAR(50)=NULL,
    @StudyMode NVARCHAR(50)=NULL,
    @SessionCode BIGINT=NULL,
    @Score INT=NULL,
    @JambScore INT=NULL,
    @CutOffMax INT=NULL,
    @AdmissionStatus NVARCHAR(50)=NULL,
    @StatusReason NVARCHAR(500)=NULL,
    @TotalScore INT=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Registry].[Admissions] WHERE ([Code] = @Code) AND ( ↙
    [UniversityCode]=@UniversityCode) AND ([Deleted]=@Deleted))
            BEGIN
                INSERT INTO [Registry].[Admissions]
                (
                    [UniversityCode],
                    [AccountCode],
                    [FacultyCode],
                    [DepartmentCode],
                    [CourseCode],
                    [ProgramCode],
                    [AwardCode],
                    [LevelCode],
                    [EntryMode],
                    [StudyMode],
                    [SessionCode],
                    [Score],
                    [JambScore],
                    [CutOffMax],
                    [AdmissionStatus],
                    [StatusReason],
                    [TotalScore],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    @UniversityCode,
                    @AccountCode,
                    @FacultyCode,
                    @DepartmentCode,
                    @CourseCode,
                    @ProgramCode,
                    @AwardCode,
                    @LevelCode,
                    @EntryMode,
                    @StudyMode,
                    @SessionCode,
                    @Score,
                    @JambScore,
                    @CutOffMax,
                    @AdmissionStatus,
                    @StatusReason,
                    @TotalScore,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [Registry].[Admissions]
                SET
                    [UniversityCode]=@UniversityCode,
                    [AccountCode]=@AccountCode,
                    [FacultyCode]=@FacultyCode,
                    [DepartmentCode]=@DepartmentCode,
                    [CourseCode]=@CourseCode,
                    [ProgramCode]=@ProgramCode,
                    [AwardCode]=@AwardCode,
                    [LevelCode]=@LevelCode,
                    [EntryMode]=@EntryMode,
                    [StudyMode]=@StudyMode,
                    [SessionCode]=@SessionCode,
                    [Score]=@Score,
                    [JambScore]=@JambScore,
                    [CutOffMax]=@CutOffMax,
                    [AdmissionStatus]=@AdmissionStatus,
                    [StatusReason]=@StatusReason,
                    [TotalScore]=@TotalScore,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Registry].[SPAdmissionsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Registry].[SPAdmissionsDeletePermanently]    Script Date↙
    : 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
```

```
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Registry.Admissions    ↙
    Table

CREATE PROC [Registry].[SPAdmissionsDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Registry].[Admissions] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Registry].[Admissions]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Registry].[SPAdmissionsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Registry].[SPAdmissionsDelete]    Script Date: 08/16/   ↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Registry.   ↙
    Admissions Table

CREATE PROC [Registry].[SPAdmissionsDelete]
```

```
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Registry].[Admissions] WHERE (([Code] = @Code) AND      ↙
    (Deleted=@Deleted)))
            BEGIN
            UPDATE [Registry].[Admissions]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Registry].[SPAdmissionsDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPFeesSelect]    Script Date: 08/16/2011 01:24↙
    :28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author             :   Ademola Adebo
--Reviewer           :   Godwin Mathias
--Date Created       :   07/18/2011
--Last Updated       :   07/18/2011
--Last Updated By    :   Ademola Adebo
--Description        :   Stored procedure for retrieving values from Finance.Fees Table

CREATE PROC [Finance].[SPFeesSelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance].[Fees] WHERE (([Code] = @Code) AND (          ↙
    [UniversityCode]=@UniversityCode) AND ([Deleted]=@Deleted)))
            BEGIN
            SELECT * FROM [Finance].[Fees]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted]=@Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Finance].[Fees]
            WHERE
                (
                ([UniversityCode]=@UniversityCode) AND
                ([Deleted]=@Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPFeesSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPFeesInsertUpdate]    Script Date: 08/16/2011↙
     01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/18/2011
--Last Updated        :   07/18/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for saving values into Finance.Fees Table

CREATE PROC [Finance].[SPFeesInsertUpdate]
```

```sql
(
    @Code BIGINT=NULL,
    @MatricNo NVARCHAR(50)=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @FacultyCode BIGINT=NULL,
    @DepartmentCode BIGINT=NULL,
    @CourseCode BIGINT=NULL,
    @ProgramCode BIGINT=NULL,
    @LevelCode INT=NULL,
    @SessionCode BIGINT=NULL,
    @SemesterCode NVARCHAR(50)=NULL,
    @AccountTitle NVARCHAR(256)=NULL,
    @AccountNo NVARCHAR(50)=NULL,
    @BankCode NVARCHAR(50)=NULL,
    @BankBranch NVARCHAR(256)=NULL,
    @PaymentMode NVARCHAR(50)=NULL,
    @DateX DATETIME2(7)=NULL,
    @Amount DECIMAL(18,0)=NULL,
    @Balance DECIMAL(18,2)=NULL,
    @PaymentStatus NVARCHAR(50)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Finance] .[Fees] WHERE ([Code] = @Code) AND (     ↙
    [UniversityCode]=@UniversityCode) AND ([Deleted]=@Deleted))
            BEGIN
                INSERT INTO [Finance] .[Fees]
                (
                    [MatricNo],
                    [UniversityCode],
                    [FacultyCode],
                    [DepartmentCode],
                    [CourseCode],
                    [ProgramCode],
                    [LevelCode],
                    [SessionCode],
                    [SemesterCode],
                    [Amount],
                    [AccountTitle],
                    [AccountNo],
                    [BankCode],
                    [BankBranch],
                    [PaymentMode],
                    [DateX],
                    [Balance],
                    [PaymentStatus],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    @MatricNo,
                    @UniversityCode,
                    @FacultyCode,
                    @DepartmentCode,
                    @CourseCode,
                    @ProgramCode,
                    @LevelCode,
                    @SessionCode,
                    @SemesterCode,
                    @Amount,
                    @AccountTitle,
                    @AccountNo,
                    @BankCode,
                    @BankBranch,
                    @PaymentMode,
                    @DateX,
                    @Balance,
                    @PaymentStatus,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [Finance] .[Fees]
                SET
                    [MatricNo]=@MatricNo,
                    [UniversityCode]=@UniversityCode,
                    [FacultyCode]=@FacultyCode,
                    [DepartmentCode]=@DepartmentCode,
                    [CourseCode]=@CourseCode,
                    [ProgramCode]=@ProgramCode,
                    [LevelCode]=@LevelCode,
                    [SessionCode]=@SessionCode,
                    [SemesterCode]=@SemesterCode,
                    [Amount]=@Amount,
                    [AccountTitle]=@AccountTitle,
                    [AccountNo]=@AccountNo,
                    [BankCode]=@BankCode,
                    [BankBranch]=@BankBranch,
                    [PaymentMode]=@PaymentMode,
                    [DateX]=@DateX,
                    [Balance]=@Balance,
                    [PaymentStatus]=@PaymentStatus,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);
```

```
END CATCH

GRANT EXECUTE ON [Finance].[SPFeesInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPFeeAllotmentSelect]    Script Date: 08/16/ ↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Finance.FeeAllotment ↙
    Table

CREATE PROC [Finance].[SPFeeAllotmentSelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Deleted BIT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance].[FeeAllotment] WHERE (([Code] = @Code) AND ( ↙
    [UniversityCode]=@UniversityCode) AND ([Deleted]=@Deleted)))
            BEGIN
            SELECT * FROM [Finance].[FeeAllotment]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted]=@Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Finance].[FeeAllotment]
            WHERE
                (
                ([UniversityCode]=@UniversityCode) AND
                ([Deleted]=@Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()
```

```
RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPFeeAllotmentSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPFeeAllotmentInsertUpdate]    Script Date: 08↙
    /16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/18/2011
--Last Updated        :   07/18/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for saving values into Finance.FeeAllotment Table

CREATE PROC [Finance].[SPFeeAllotmentInsertUpdate]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @FacultyCode BIGINT=NULL,
    @DepartmentCode BIGINT=NULL,
    @CourseCode BIGINT=NULL,
    @ProgramCode BIGINT=NULL,
    @LevelCode INT=NULL,
    @EntryMode NVARCHAR(50)=NULL,
    @StudyMode NVARCHAR(50)=NULL,
    @SessionCode BIGINT=NULL,
    @SemesterCode NVARCHAR(50)=NULL,
    @FeeDefinitionCode NVARCHAR(50)=NULL,
    @Amount DECIMAL(18,0)=NULL,
    @DiscountPercentage INT=NULL,
    @Approved BIT=NULL,
    @ApproveBy NVARCHAR(50)=NULL,
    @ApproveOn DATETIME2(7)=NULL,
    @Notes NVARCHAR(256)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Finance] .[FeeAllotment] WHERE ([Code] = @Code) AND ↙
    ([UniversityCode]=@UniversityCode) AND ([Deleted]=@Deleted))
            BEGIN
                INSERT INTO [Finance] .[FeeAllotment]
                (
                    [UniversityCode],
                    [FacultyCode],
                    [DepartmentCode],
                    [CourseCode],
                    [ProgramCode],
                    [LevelCode],
                    [EntryMode],
                    [StudyMode],
                    [SessionCode],
                    [SemesterCode],
                    [FeeDefinitionCode],
                    [Amount],
                    [DiscountPercentage],
                    [Approved],
                    [ApproveBy],
                    [ApproveOn],
                    [Notes],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]
                )
                VALUES
                (
                    @UniversityCode,
                    @FacultyCode,
                    @DepartmentCode,
                    @CourseCode,
                    @ProgramCode,
                    @LevelCode,
                    @EntryMode,
                    @StudyMode,
                    @SessionCode,
                    @SemesterCode,
                    @FeeDefinitionCode,
                    @Amount,
                    @DiscountPercentage,
                    @Approved,
                    @ApproveBy,
                    @ApproveOn,
                    @Notes,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted
                )
            END
        ELSE
            BEGIN
                UPDATE [Finance] .[FeeAllotment]
                SET
                    [UniversityCode]=@UniversityCode,
                    [FacultyCode]=@FacultyCode,
                    [DepartmentCode]=@DepartmentCode,
                    [CourseCode]=@CourseCode,
                    [ProgramCode]=@ProgramCode,
                    [LevelCode]=@LevelCode,
                    [EntryMode]=@EntryMode,
                    [StudyMode]=@StudyMode,
                    [SessionCode]=@SessionCode,
                    [SemesterCode]=@SemesterCode,
                    [FeeDefinitionCode]=@FeeDefinitionCode,
                    [Amount]=@Amount,
                    [DiscountPercentage]=@DiscountPercentage,
                    [Approved]=@Approved,
                    [ApproveBy]=@ApproveBy,
                    [ApproveOn]=@ApproveOn,
                    [Notes]=@Notes,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted
                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
```

```
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPFeeAllotmentInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPFeeAllotmentDeletePermanently]    Script    ↙
    Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Finance.FeeAllotment    ↙
    Table

CREATE PROC [Finance].[SPFeeAllotmentDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance].[FeeAllotment] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Finance].[FeeAllotment]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPFeeAllotmentDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPFeeAllotmentDelete]    Script Date: 08/16/ ↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Finance. ↙
    FeeAllotment Table

CREATE PROC [Finance].[SPFeeAllotmentDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance].[FeeAllotment] WHERE ([Code] = @Code) AND ↙
    (Deleted=@Deleted))
            BEGIN
            UPDATE [Finance].[FeeAllotment]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPFeeAllotmentDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPPaymentPlansSelect]    Script Date: 08/16/ ↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Finance.PaymentPlans ↙
    Table

CREATE PROC [Finance].[SPPaymentPlansSelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Deleted BIT=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance].[PaymentPlans] WHERE (([Code] = @Code) AND ( ↙
    [UniversityCode] = @UniversityCode) AND ([Deleted] = @Deleted)))
            BEGIN
            SELECT * FROM [Finance].[PaymentPlans]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Finance].[PaymentPlans]
            WHERE
                (
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);
```

```
SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPPaymentPlansSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPPaymentPlansInsertUpdate]    Script Date: 08↙
    /16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Finance.PaymentPlans Table

CREATE PROC [Finance].[SPPaymentPlansInsertUpdate]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @SessionCode BIGINT=NULL,
    @SemesterCode NVARCHAR(50)=NULL,
    @NoOfPaymentAllowed INT=NULL,
    @DateX DATETIME2(7)=NULL,
    @StatusCode NVARCHAR(50)=NULL,
    @Applicable BIT=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Finance] .[PaymentPlans] WHERE ([Code] = @Code) AND ↙
    ([UniversityCode] = @UniversityCode))
            BEGIN
                INSERT INTO [Finance] .[PaymentPlans]
                (
                    [UniversityCode],
                    [SessionCode],
                    [SemesterCode],
                    [NoOfPaymentAllowed],
                    [DateX],
                    [StatusCode],
                    [Applicable],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]

                )
                VALUES
                (
                    @UniversityCode,
                    @SessionCode,
                    @SemesterCode,
                    @NoOfPaymentAllowed,
                    @DateX,
                    @StatusCode,
                    @Applicable,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted

                )
            END
        ELSE
            BEGIN
                UPDATE [Finance] .[PaymentPlans]
                SET
                    [UniversityCode]=@UniversityCode,
                    [SessionCode]=@SessionCode,
                    [SemesterCode]=@SemesterCode,
                    [NoOfPaymentAllowed]=@NoOfPaymentAllowed,
                    [DateX]=@DateX,
                    [StatusCode]=@StatusCode,
                    [Applicable]=@Applicable,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted

                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPPaymentPlanDetailsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPPaymentPlansDeletePermanently]    Script    ↙
    Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
```

```
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Finance.PaymentPlans    ↙
    Table

CREATE PROC [Finance].[SPPaymentPlansDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance].[PaymentPlans] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Finance].[PaymentPlans]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPPaymentPlansDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPPaymentPlansDelete]    Script Date: 08/16/ ↙
    2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Finance.    ↙
    PaymentPlans Table

CREATE PROC [Finance].[SPPaymentPlansDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance].[PaymentPlans] WHERE ([Code] = @Code) AND (Deleted=@Deleted))
            BEGIN
            UPDATE [Finance].[PaymentPlans]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPPaymentPlansDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPPaymentPlanDetailsSelect]    Script Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Finance.PaymentPlanDetails Table

CREATE PROC [Finance].[SPPaymentPlanDetailsSelect]
(
    @Code BIGINT=NULL,
    @PaymentPlanCode BIGINT=NULL

)
```

```sql
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance].[PaymentPlanDetails] WHERE (([Code] = @Code)   ↙
    AND ([PaymentPlanCode] = @PaymentPlanCode)))
            BEGIN
            SELECT * FROM [Finance].[PaymentPlanDetails]
            WHERE
                (
                ([Code] = @Code) AND
                ([PaymentPlanCode] = @PaymentPlanCode)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Finance].[PaymentPlanDetails]
            WHERE
                (
                ([PaymentPlanCode] = @PaymentPlanCode)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPPaymentPlanDetailsSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPPaymentPlanDetailsInsertUpdate]    Script   ↙
    Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Finance.PaymentPlanDetails↙
     Table

CREATE PROC [Finance].[SPPaymentPlanDetailsInsertUpdate]
(
    @Code BIGINT=NULL,
    @PaymentPlanCode BIGINT=NULL,
    @Description NVARCHAR(256)=NULL,
    @Amount DECIMAL(18,2)=NULL,
    @DiscountedAmount DECIMAL(18,2)=NULL,
    @DiscountedPercentage INT=NULL,
    @Total DECIMAL(18,2)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Finance] .[PaymentPlanDetails] WHERE ([Code] = @    ↙
    Code))
            BEGIN
                INSERT INTO [Finance] .[PaymentPlanDetails]
                (
                    [PaymentPlanCode],
                    [Description],
                    [Amount],
                    [DiscountedAmount],
                    [DiscountedPercentage],
                    [Total],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]

                )
                VALUES
                (
                    @PaymentPlanCode,
                    @Description,
                    @Amount,
                    @DiscountedAmount,
                    @DiscountedPercentage,
                    @Total,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted

                )
            END
        ELSE
            BEGIN
                UPDATE [Finance] .[PaymentPlanDetails]
                SET
                    [PaymentPlanCode]=@PaymentPlanCode,
                    [Description]=@Description,
                    [Amount]=@Amount,
                    [DiscountedAmount]=@DiscountedAmount,
                    [DiscountedPercentage]=@DiscountedPercentage,
                    [Total]=@Total,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted

                WHERE
                    (([Code] = @Code))
            END
COMMIT TRAN
END TRY
```

```
BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPPaymentPlanDetailsInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPPaymentPlanDetailsDeletePermanently]         ↙
    Script Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Finance.                 ↙
    PaymentPlanDetails Table

CREATE PROC [Finance].[SPPaymentPlanDetailsDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance].[PaymentPlanDetails] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Finance].[PaymentPlanDetails]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
```

```
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPPaymentPlanDetailsDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPPaymentPlanDetailsDelete]    Script Date: 08↙
    /16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values temporarily from Finance.    ↙
    PaymentPlanDetails Table

CREATE PROC [Finance].[SPPaymentPlanDetailsDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance].[PaymentPlanDetails] WHERE ([Code] = @Code) AND↙
     (Deleted=@Deleted))
            BEGIN
            UPDATE [Finance].[PaymentPlanDetails]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
```

```sql
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPPaymentPlanDetailsDelete] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPFeesDetailSelect]    Script Date: 08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for retrieving values from Finance.FeesDetail Table

CREATE PROC [Finance].[SPFeesDetailSelect]
(
    @Code BIGINT=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @Deleted BIT=NULL

)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance].[FeesDetail] WHERE (([Code] = @Code) AND ([UniversityCode]=@UniversityCode) AND ([Deleted] = @Deleted)))
            BEGIN
            SELECT * FROM [Finance].[FeesDetail]
            WHERE
                (
                ([Code] = @Code) AND
                ([UniversityCode] = @UniversityCode) AND
                ([Deleted] = @Deleted)
                )
            END
        ELSE
            BEGIN
            SELECT * FROM [Finance].[FeesDetail]
            WHERE
                (
                ([UniversityCode]=@UniversityCode) AND
                ([Deleted]=@Deleted)
                )
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPFeesDetailSelect] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPFeesDetailInsertUpdate]    Script Date: 08/ ↙
    16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for saving values into Finance.FeesDetail Table

CREATE PROC [Finance].[SPFeesDetailInsertUpdate]
(
    @Code BIGINT=NULL,
    @MatricNo NVARCHAR(50)=NULL,
    @UniversityCode NVARCHAR(50)=NULL,
    @FacultyCode BIGINT=NULL,
    @DepartmentCode BIGINT=NULL,
    @CourseCode BIGINT=NULL,
    @ProgramCode BIGINT=NULL,
    @LevelCode INT=NULL,
    @FeesCode BIGINT=NULL,
    @FeeDefinitionCode NVARCHAR(50)=NULL,
    @FeeAllotmentCode BIGINT=NULL,
    @Amount DECIMAL(18,2)=NULL,
    @ExtraDiscount INT=NULL,
    @FinalAmount DECIMAL(18,2)=NULL,
    @CreatedOn DATETIME2(7)=NULL,
    @CreatedBy NVARCHAR(50)=NULL,
    @ModifiedOn DATETIME2(7)=NULL,
    @ModifiedBy NVARCHAR(50)=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF NOT EXISTS (SELECT * FROM [Finance] .[FeesDetail] WHERE ([Code] = @Code) AND ( ↙
    [UniversityCode]=@UniversityCode) AND ([Deleted] = @Deleted))
            BEGIN
                INSERT INTO [Finance] .[FeesDetail]
                (
                    [MatricNo],
                    [UniversityCode],
                    [FacultyCode],
                    [DepartmentCode],
                    [CourseCode],
                    [ProgramCode],
                    [LevelCode],
                    [FeesCode],
                    [FeeDefinitionCode],
                    [FeeAllotmentCode],
                    [Amount],
                    [ExtraDiscount],
                    [FinalAmount],
                    [CreatedOn],
                    [CreatedBy],
                    [Deleted]

                )
                VALUES
                (
                    @MatricNo,
                    @UniversityCode,
                    @FacultyCode,
                    @DepartmentCode,
                    @CourseCode,
                    @ProgramCode,
                    @LevelCode,
                    @FeesCode,
                    @FeeDefinitionCode,
                    @FeeAllotmentCode,
                    @Amount,
                    @ExtraDiscount,
                    @FinalAmount,
                    @CreatedOn,
                    @CreatedBy,
                    @Deleted

                )
            END
        ELSE
            BEGIN
                UPDATE [Finance] .[FeesDetail]
                SET
                    [MatricNo]=@MatricNo,
                    [UniversityCode]=@UniversityCode,
                    [FacultyCode]=@FacultyCode,
                    [DepartmentCode]=@DepartmentCode,
                    [CourseCode]=@CourseCode,
                    [ProgramCode]=@ProgramCode,
                    [LevelCode]=@LevelCode,
                    [FeesCode]=@FeesCode,
                    [FeeDefinitionCode]=@FeeDefinitionCode,
                    [FeeAllotmentCode]=@FeeAllotmentCode,
                    [Amount]=@Amount,
                    [ExtraDiscount]=@ExtraDiscount,
                    [FinalAmount]=@FinalAmount,
                    [ModifiedOn]=@ModifiedOn,
                    [ModifiedBy]=@ModifiedBy,
                    [Deleted]=@Deleted

                WHERE
                    (([Code] = @Code) AND ([UniversityCode] = @UniversityCode) AND (      ↙
    [Deleted] = @Deleted))
            END
COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPFeesDetailInsertUpdate] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPFeesDetailDeletePermanently]    Script Date:↙
     08/16/2011 01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author            :   Ademola Adebo
--Reviewer          :   Godwin Mathias
--Date Created      :   07/18/2011
--Last Updated      :   07/18/2011
--Last Updated By   :   Ademola Adebo
--Description       :   Stored procedure for deleting values from Finance.FeesDetail Table

CREATE PROC [Finance].[SPFeesDetailDeletePermanently]
(
    @Code BIGINT=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance].[FeesDetail] WHERE (([Code] = @Code)))
            BEGIN
            DELETE FROM [Finance].[FeesDetail]
            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()
```

```
RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);

END CATCH

GRANT EXECUTE ON [Finance].[SPFeesDetailDeletePermanently] TO PUBLIC
GO
/****** Object:  StoredProcedure [Finance].[SPFeesDetailDelete]    Script Date: 08/16/2011↙
     01:24:28 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
--Author              :   Ademola Adebo
--Reviewer            :   Godwin Mathias
--Date Created        :   07/18/2011
--Last Updated        :   07/18/2011
--Last Updated By     :   Ademola Adebo
--Description         :   Stored procedure for deleting values temporarily from Finance.    ↙
    FeesDetail Table

CREATE PROC [Finance].[SPFeesDetailDelete]
(
    @Code BIGINT=NULL,
    @Deleted BIT=NULL,
    @DeletedOn DATETIME2(7)=NULL,
    @DeletedBy NVARCHAR(50)=NULL
)
AS
BEGIN TRY
BEGIN TRAN
SET NOCOUNT ON;

        IF EXISTS (SELECT * FROM [Finance].[FeesDetail] WHERE ([Code] = @Code) AND        ↙
    (Deleted=@Deleted))
            BEGIN
            UPDATE [Finance].[FeesDetail]
            SET
                [Deleted]=@Deleted,
                [DeletedOn]=@DeletedOn,
                [DeletedBy]=@DeletedBy

            WHERE
                (
                    ([Code] = @Code)
                )
            END

COMMIT TRAN
END TRY

BEGIN CATCH
ROLLBACK TRAN

DECLARE @ErrorNumber_INT INT;
DECLARE @ErrorSeverity_INT INT;
DECLARE @ErrorProcedure_VC VARCHAR(200);
DECLARE @ErrorLine_INT INT;
DECLARE @ErrorMessage_NVC NVARCHAR(4000);

SELECT
        @ErrorMessage_NVC = ERROR_MESSAGE(),
        @ErrorSeverity_INT = ERROR_SEVERITY(),
        @ErrorNumber_INT = ERROR_NUMBER(),
        @ErrorProcedure_VC = ERROR_PROCEDURE(),
        @ErrorLine_INT = ERROR_LINE()

RAISERROR(@ErrorMessage_NVC,@ErrorSeverity_INT,1);
```

```
END CATCH

GRANT EXECUTE ON [Finance].[SPFeesDetailDelete] TO PUBLIC
GO
/****** Object:  Default [DF__aspnet_Ap__Appli__7B905C75]    Script Date: 08/16/2011 01:24↙
    :32 ******/
ALTER TABLE [dbo].[aspnet_Applications] ADD  CONSTRAINT [DF__aspnet_Ap__Appli__7B905C75]  ↙
    DEFAULT (newid()) FOR [ApplicationId]
GO
/****** Object:  Default [DF__aspnet_Me__Passw__108B795B]    Script Date: 08/16/2011 01:24↙
    :32 ******/
ALTER TABLE [dbo].[aspnet_Membership] ADD  CONSTRAINT [DF__aspnet_Me__Passw__108B795B]    ↙
    DEFAULT ((0)) FOR [PasswordFormat]
GO
/****** Object:  Default [DF__aspnet_Pa__PathI__412EB0B6]    Script Date: 08/16/2011 01:24↙
    :32 ******/
ALTER TABLE [dbo].[aspnet_Paths] ADD  CONSTRAINT [DF__aspnet_Pa__PathI__412EB0B6]  DEFAULT↙
     (newid()) FOR [PathId]
GO
/****** Object:  Default [DF__aspnet_Perso__Id__48CFD27E]    Script Date: 08/16/2011 01:24↙
    :32 ******/
ALTER TABLE [dbo].[aspnet_PersonalizationPerUser] ADD  CONSTRAINT                         ↙
    [DF__aspnet_Perso__Id__48CFD27E]  DEFAULT (newid()) FOR [Id]
GO
/****** Object:  Default [DF__aspnet_Ro__RoleI__2E1BDC42]    Script Date: 08/16/2011 01:24↙
    :32 ******/
ALTER TABLE [dbo].[aspnet_Roles] ADD  CONSTRAINT [DF__aspnet_Ro__RoleI__2E1BDC42]  DEFAULT↙
     (newid()) FOR [RoleId]
GO
/****** Object:  Default [DF__aspnet_Us__UserI__7F60ED59]    Script Date: 08/16/2011 01:24↙
    :32 ******/
ALTER TABLE [dbo].[aspnet_Users] ADD  CONSTRAINT [DF__aspnet_Us__UserI__7F60ED59]  DEFAULT↙
     (newid()) FOR [UserId]
GO
/****** Object:  Default [DF__aspnet_Us__Mobil__00551192]    Script Date: 08/16/2011 01:24↙
    :32 ******/
ALTER TABLE [dbo].[aspnet_Users] ADD  CONSTRAINT [DF__aspnet_Us__Mobil__00551192]  DEFAULT↙
     (NULL) FOR [MobileAlias]
GO
/****** Object:  Default [DF__aspnet_Us__IsAno__014935CB]    Script Date: 08/16/2011 01:24↙
    :32 ******/
ALTER TABLE [dbo].[aspnet_Users] ADD  CONSTRAINT [DF__aspnet_Us__IsAno__014935CB]  DEFAULT↙
     ((0)) FOR [IsAnonymous]
GO
/****** Object:  ForeignKey [FK_FinalSignatories_Students]    Script Date: 08/16/2011 01: ↙
    24:32 ******/
ALTER TABLE [Academics].[FinalSignatories]  WITH CHECK ADD  CONSTRAINT                    ↙
    [FK_FinalSignatories_Students] FOREIGN KEY([StudentCode], [MatricNo])
REFERENCES [Registry].[Students] ([Code], [MatricNo])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [Academics].[FinalSignatories] CHECK CONSTRAINT [FK_FinalSignatories_Students]
GO
/****** Object:  ForeignKey [FK_Registrations_Students]    Script Date: 08/16/2011 01:24: ↙
    32 ******/
ALTER TABLE [Academics].[Registrations]  WITH CHECK ADD  CONSTRAINT                       ↙
    [FK_Registrations_Students] FOREIGN KEY([StudentCode], [MatricNo])
REFERENCES [Registry].[Students] ([Code], [MatricNo])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [Academics].[Registrations] CHECK CONSTRAINT [FK_Registrations_Students]
GO
/****** Object:  ForeignKey [FK_Sessions_Universities]    Script Date: 08/16/2011 01:24:32↙
     ******/
```

```
ALTER TABLE [Academics].[Sessions]  WITH CHECK ADD  CONSTRAINT [FK_Sessions_Universities] ↙
    FOREIGN KEY([UniversityCode])
REFERENCES [SetUp].[Universities] ([Code])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [Academics].[Sessions] CHECK CONSTRAINT [FK_Sessions_Universities]
GO
/****** Object:  ForeignKey [FK_SigningHierarchy_Sessions]    Script Date: 08/16/2011 01: ↙
    24:32 ******/
ALTER TABLE [Academics].[SigningHierarchy]  WITH CHECK ADD  CONSTRAINT                     ↙
    [FK_SigningHierarchy_Sessions] FOREIGN KEY([SessionCode])
REFERENCES [Academics].[Sessions] ([Code])
GO
ALTER TABLE [Academics].[SigningHierarchy] CHECK CONSTRAINT [FK_SigningHierarchy_Sessions]
GO
/****** Object:  ForeignKey [FK_SigningHierarchy_Universities]    Script Date: 08/16/2011 ↙
    01:24:32 ******/
ALTER TABLE [Academics].[SigningHierarchy]  WITH CHECK ADD  CONSTRAINT                     ↙
    [FK_SigningHierarchy_Universities] FOREIGN KEY([UniversityCode])
REFERENCES [SetUp].[Universities] ([Code])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [Academics].[SigningHierarchy] CHECK CONSTRAINT                                ↙
    [FK_SigningHierarchy_Universities]
GO
/****** Object:  ForeignKey [FK_Submissions_Students]    Script Date: 08/16/2011 01:24:32 ↙
    ******/
ALTER TABLE [Academics].[Submissions]  WITH CHECK ADD  CONSTRAINT                          ↙
    [FK_Submissions_Students] FOREIGN KEY([StudentCode], [MatricNo])
REFERENCES [Registry].[Students] ([Code], [MatricNo])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [Academics].[Submissions] CHECK CONSTRAINT [FK_Submissions_Students]
GO
/****** Object:  ForeignKey [FK_Attendance_Students]    Script Date: 08/16/2011 01:24:32 *↙
    *****/
ALTER TABLE [Assessment].[Attendance]  WITH CHECK ADD  CONSTRAINT [FK_Attendance_Students]↙
     FOREIGN KEY([StudentCode], [MatricNo])
REFERENCES [Registry].[Students] ([Code], [MatricNo])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [Assessment].[Attendance] CHECK CONSTRAINT [FK_Attendance_Students]
GO
/****** Object:  ForeignKey [FK_Records_Students]    Script Date: 08/16/2011 01:24:32 ****↙
    **/
ALTER TABLE [Assessment].[Records]  WITH CHECK ADD  CONSTRAINT [FK_Records_Students]      ↙
    FOREIGN KEY([StudentCode], [MatricNo])
REFERENCES [Registry].[Students] ([Code], [MatricNo])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [Assessment].[Records] CHECK CONSTRAINT [FK_Records_Students]
GO
/****** Object:  ForeignKey [FK__aspnet_Me__Appli__395884C4]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_Membership]  WITH CHECK ADD FOREIGN KEY([ApplicationId])
REFERENCES [dbo].[aspnet_Applications] ([ApplicationId])
GO
/****** Object:  ForeignKey [FK__aspnet_Me__Appli__3A4CA8FD]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_Membership]  WITH CHECK ADD FOREIGN KEY([ApplicationId])
REFERENCES [dbo].[aspnet_Applications] ([ApplicationId])
GO
```

```
/****** Object:  ForeignKey [FK__aspnet_Me__Appli__3B40CD36]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_Membership]  WITH CHECK ADD FOREIGN KEY([ApplicationId])
REFERENCES [dbo].[aspnet_Applications] ([ApplicationId])
GO
/****** Object:  ForeignKey [FK__aspnet_Me__Appli__3C34F16F]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_Membership]  WITH CHECK ADD FOREIGN KEY([ApplicationId])
REFERENCES [dbo].[aspnet_Applications] ([ApplicationId])
GO
/****** Object:  ForeignKey [FK__aspnet_Me__UserI__3D2915A8]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_Membership]  WITH CHECK ADD FOREIGN KEY([UserId])
REFERENCES [dbo].[aspnet_Users] ([UserId])
GO
/****** Object:  ForeignKey [FK__aspnet_Me__UserI__3E1D39E1]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_Membership]  WITH CHECK ADD FOREIGN KEY([UserId])
REFERENCES [dbo].[aspnet_Users] ([UserId])
GO
/****** Object:  ForeignKey [FK__aspnet_Me__UserI__3F115E1A]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_Membership]  WITH CHECK ADD FOREIGN KEY([UserId])
REFERENCES [dbo].[aspnet_Users] ([UserId])
GO
/****** Object:  ForeignKey [FK__aspnet_Me__UserI__40058253]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_Membership]  WITH CHECK ADD FOREIGN KEY([UserId])
REFERENCES [dbo].[aspnet_Users] ([UserId])
GO
/****** Object:  ForeignKey [FK__aspnet_Pa__Appli__40F9A68C]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_Paths]  WITH CHECK ADD FOREIGN KEY([ApplicationId])
REFERENCES [dbo].[aspnet_Applications] ([ApplicationId])
GO
/****** Object:  ForeignKey [FK__aspnet_Pa__Appli__41EDCAC5]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_Paths]  WITH CHECK ADD FOREIGN KEY([ApplicationId])
REFERENCES [dbo].[aspnet_Applications] ([ApplicationId])
GO
/****** Object:  ForeignKey [FK__aspnet_Pa__Appli__42E1EEFE]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_Paths]  WITH CHECK ADD FOREIGN KEY([ApplicationId])
REFERENCES [dbo].[aspnet_Applications] ([ApplicationId])
GO
/****** Object:  ForeignKey [FK__aspnet_Pa__Appli__43D61337]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_Paths]  WITH CHECK ADD FOREIGN KEY([ApplicationId])
REFERENCES [dbo].[aspnet_Applications] ([ApplicationId])
GO
/****** Object:  ForeignKey [FK__aspnet_Pe__PathI__44CA3770]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_PersonalizationAllUsers]  WITH CHECK ADD FOREIGN KEY([PathId])
REFERENCES [dbo].[aspnet_Paths] ([PathId])
GO
/****** Object:  ForeignKey [FK__aspnet_Pe__PathI__45BE5BA9]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_PersonalizationAllUsers]  WITH CHECK ADD FOREIGN KEY([PathId])
REFERENCES [dbo].[aspnet_Paths] ([PathId])
GO
/****** Object:  ForeignKey [FK__aspnet_Pe__PathI__46B27FE2]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_PersonalizationAllUsers]  WITH CHECK ADD FOREIGN KEY([PathId])
REFERENCES [dbo].[aspnet_Paths] ([PathId])
GO
/****** Object:  ForeignKey [FK__aspnet_Pe__PathI__47A6A41B]    Script Date: 08/16/2011 01↙
    :24:32 ******/
```

```
ALTER TABLE [dbo].[aspnet_PersonalizationAllUsers]  WITH CHECK ADD FOREIGN KEY([PathId])
REFERENCES [dbo].[aspnet_Paths] ([PathId])
GO
/****** Object:  ForeignKey [FK__aspnet_Pe__PathI__489AC854]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_PersonalizationPerUser]  WITH CHECK ADD FOREIGN KEY([PathId])
REFERENCES [dbo].[aspnet_Paths] ([PathId])
GO
/****** Object:  ForeignKey [FK__aspnet_Pe__PathI__498EEC8D]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_PersonalizationPerUser]  WITH CHECK ADD FOREIGN KEY([PathId])
REFERENCES [dbo].[aspnet_Paths] ([PathId])
GO
/****** Object:  ForeignKey [FK__aspnet_Pe__PathI__4A8310C6]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_PersonalizationPerUser]  WITH CHECK ADD FOREIGN KEY([PathId])
REFERENCES [dbo].[aspnet_Paths] ([PathId])
GO
/****** Object:  ForeignKey [FK__aspnet_Pe__PathI__4B7734FF]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_PersonalizationPerUser]  WITH CHECK ADD FOREIGN KEY([PathId])
REFERENCES [dbo].[aspnet_Paths] ([PathId])
GO
/****** Object:  ForeignKey [FK__aspnet_Pe__UserI__4C6B5938]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_PersonalizationPerUser]  WITH CHECK ADD FOREIGN KEY([UserId])
REFERENCES [dbo].[aspnet_Users] ([UserId])
GO
/****** Object:  ForeignKey [FK__aspnet_Pe__UserI__4D5F7D71]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_PersonalizationPerUser]  WITH CHECK ADD FOREIGN KEY([UserId])
REFERENCES [dbo].[aspnet_Users] ([UserId])
GO
/****** Object:  ForeignKey [FK__aspnet_Pe__UserI__4E53A1AA]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_PersonalizationPerUser]  WITH CHECK ADD FOREIGN KEY([UserId])
REFERENCES [dbo].[aspnet_Users] ([UserId])
GO
/****** Object:  ForeignKey [FK__aspnet_Pe__UserI__4F47C5E3]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_PersonalizationPerUser]  WITH CHECK ADD FOREIGN KEY([UserId])
REFERENCES [dbo].[aspnet_Users] ([UserId])
GO
/****** Object:  ForeignKey [FK__aspnet_Pr__UserI__503BEA1C]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_Profile]  WITH CHECK ADD FOREIGN KEY([UserId])
REFERENCES [dbo].[aspnet_Users] ([UserId])
GO
/****** Object:  ForeignKey [FK__aspnet_Pr__UserI__51300E55]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_Profile]  WITH CHECK ADD FOREIGN KEY([UserId])
REFERENCES [dbo].[aspnet_Users] ([UserId])
GO
/****** Object:  ForeignKey [FK__aspnet_Pr__UserI__5224328E]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_Profile]  WITH CHECK ADD FOREIGN KEY([UserId])
REFERENCES [dbo].[aspnet_Users] ([UserId])
GO
/****** Object:  ForeignKey [FK__aspnet_Pr__UserI__531856C7]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_Profile]  WITH CHECK ADD FOREIGN KEY([UserId])
REFERENCES [dbo].[aspnet_Users] ([UserId])
GO
/****** Object:  ForeignKey [FK__aspnet_Ro__Appli__540C7B00]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_Roles]  WITH CHECK ADD FOREIGN KEY([ApplicationId])
REFERENCES [dbo].[aspnet_Applications] ([ApplicationId])
```

```
GO
/****** Object:  ForeignKey [FK__aspnet_Ro__Appli__55009F39]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_Roles]  WITH CHECK ADD FOREIGN KEY([ApplicationId])
REFERENCES [dbo].[aspnet_Applications] ([ApplicationId])
GO
/****** Object:  ForeignKey [FK__aspnet_Ro__Appli__55F4C372]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_Roles]  WITH CHECK ADD FOREIGN KEY([ApplicationId])
REFERENCES [dbo].[aspnet_Applications] ([ApplicationId])
GO
/****** Object:  ForeignKey [FK__aspnet_Ro__Appli__56E8E7AB]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_Roles]  WITH CHECK ADD FOREIGN KEY([ApplicationId])
REFERENCES [dbo].[aspnet_Applications] ([ApplicationId])
GO
/****** Object:  ForeignKey [FK__aspnet_Us__Appli__57DD0BE4]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_Users]  WITH CHECK ADD FOREIGN KEY([ApplicationId])
REFERENCES [dbo].[aspnet_Applications] ([ApplicationId])
GO
/****** Object:  ForeignKey [FK__aspnet_Us__Appli__58D1301D]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_Users]  WITH CHECK ADD FOREIGN KEY([ApplicationId])
REFERENCES [dbo].[aspnet_Applications] ([ApplicationId])
GO
/****** Object:  ForeignKey [FK__aspnet_Us__Appli__59C55456]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_Users]  WITH CHECK ADD FOREIGN KEY([ApplicationId])
REFERENCES [dbo].[aspnet_Applications] ([ApplicationId])
GO
/****** Object:  ForeignKey [FK__aspnet_Us__Appli__5AB9788F]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_Users]  WITH CHECK ADD FOREIGN KEY([ApplicationId])
REFERENCES [dbo].[aspnet_Applications] ([ApplicationId])
GO
/****** Object:  ForeignKey [FK_aspnet_UsersInBranches_aspnet_Users]    Script Date: 08/16↙
    /2011 01:24:32 ******/
ALTER TABLE [dbo].[aspnet_UsersInBranches]  WITH CHECK ADD  CONSTRAINT                    ↙
    [FK_aspnet_UsersInBranches_aspnet_Users] FOREIGN KEY([UserId])
REFERENCES [dbo].[aspnet_Users] ([UserId])
GO
ALTER TABLE [dbo].[aspnet_UsersInBranches] CHECK CONSTRAINT                               ↙
    [FK_aspnet_UsersInBranches_aspnet_Users]
GO
/****** Object:  ForeignKey [FK__aspnet_Us__RoleI__2CF37936]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_UsersInRoles]  WITH CHECK ADD  CONSTRAINT                       ↙
    [FK__aspnet_Us__RoleI__2CF37936] FOREIGN KEY([RoleId])
REFERENCES [dbo].[aspnet_Roles] ([RoleId])
GO
ALTER TABLE [dbo].[aspnet_UsersInRoles] CHECK CONSTRAINT [FK__aspnet_Us__RoleI__2CF37936]
GO
/****** Object:  ForeignKey [FK__aspnet_Us__RoleI__50FBCF82]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_UsersInRoles]  WITH CHECK ADD  CONSTRAINT                       ↙
    [FK__aspnet_Us__RoleI__50FBCF82] FOREIGN KEY([RoleId])
REFERENCES [dbo].[aspnet_Roles] ([RoleId])
GO
ALTER TABLE [dbo].[aspnet_UsersInRoles] CHECK CONSTRAINT [FK__aspnet_Us__RoleI__50FBCF82]
GO
/****** Object:  ForeignKey [FK__aspnet_Us__RoleI__5AEF4E10]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_UsersInRoles]  WITH CHECK ADD  CONSTRAINT                       ↙
    [FK__aspnet_Us__RoleI__5AEF4E10] FOREIGN KEY([RoleId])
REFERENCES [dbo].[aspnet_Roles] ([RoleId])
GO
```

```
ALTER TABLE [dbo].[aspnet_UsersInRoles] CHECK CONSTRAINT [FK__aspnet_Us__RoleI__5AEF4E10]
GO
/****** Object:  ForeignKey [FK__aspnet_Us__RoleI__7EF7A45C]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_UsersInRoles]  WITH CHECK ADD  CONSTRAINT                      ↙
    [FK__aspnet_Us__RoleI__7EF7A45C] FOREIGN KEY([RoleId])
REFERENCES [dbo].[aspnet_Roles] ([RoleId])
GO
ALTER TABLE [dbo].[aspnet_UsersInRoles] CHECK CONSTRAINT [FK__aspnet_Us__RoleI__7EF7A45C]
GO
/****** Object:  ForeignKey [FK__aspnet_Us__UserI__2DE79D6F]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_UsersInRoles]  WITH CHECK ADD  CONSTRAINT                      ↙
    [FK__aspnet_Us__UserI__2DE79D6F] FOREIGN KEY([UserId])
REFERENCES [dbo].[aspnet_Users] ([UserId])
GO
ALTER TABLE [dbo].[aspnet_UsersInRoles] CHECK CONSTRAINT [FK__aspnet_Us__UserI__2DE79D6F]
GO
/****** Object:  ForeignKey [FK__aspnet_Us__UserI__51EFF3BB]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_UsersInRoles]  WITH CHECK ADD  CONSTRAINT                      ↙
    [FK__aspnet_Us__UserI__51EFF3BB] FOREIGN KEY([UserId])
REFERENCES [dbo].[aspnet_Users] ([UserId])
GO
ALTER TABLE [dbo].[aspnet_UsersInRoles] CHECK CONSTRAINT [FK__aspnet_Us__UserI__51EFF3BB]
GO
/****** Object:  ForeignKey [FK__aspnet_Us__UserI__5BE37249]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_UsersInRoles]  WITH CHECK ADD  CONSTRAINT                      ↙
    [FK__aspnet_Us__UserI__5BE37249] FOREIGN KEY([UserId])
REFERENCES [dbo].[aspnet_Users] ([UserId])
GO
ALTER TABLE [dbo].[aspnet_UsersInRoles] CHECK CONSTRAINT [FK__aspnet_Us__UserI__5BE37249]
GO
/****** Object:  ForeignKey [FK__aspnet_Us__UserI__7FEBC895]    Script Date: 08/16/2011 01↙
    :24:32 ******/
ALTER TABLE [dbo].[aspnet_UsersInRoles]  WITH CHECK ADD  CONSTRAINT                      ↙
    [FK__aspnet_Us__UserI__7FEBC895] FOREIGN KEY([UserId])
REFERENCES [dbo].[aspnet_Users] ([UserId])
GO
ALTER TABLE [dbo].[aspnet_UsersInRoles] CHECK CONSTRAINT [FK__aspnet_Us__UserI__7FEBC895]
GO
/****** Object:  ForeignKey [FK_Accounts_Universities]    Script Date: 08/16/2011 01:24:32↙
     ******/
ALTER TABLE [Finance].[Accounts]  WITH CHECK ADD  CONSTRAINT [FK_Accounts_Universities]  ↙
    FOREIGN KEY([UniversityCode])
REFERENCES [SetUp].[Universities] ([Code])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [Finance].[Accounts] CHECK CONSTRAINT [FK_Accounts_Universities]
GO
/****** Object:  ForeignKey [FK_AccountSub_Parameters]    Script Date: 08/16/2011 01:24:32↙
     ******/
ALTER TABLE [Finance].[AccountSub]  WITH CHECK ADD  CONSTRAINT [FK_AccountSub_Parameters] ↙
    FOREIGN KEY([ParameterCode])
REFERENCES [SetUp].[Parameters] ([Code])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [Finance].[AccountSub] CHECK CONSTRAINT [FK_AccountSub_Parameters]
GO
/****** Object:  ForeignKey [FK_AccountSub_Universities]    Script Date: 08/16/2011 01:24:↙
    32 ******/
ALTER TABLE [Finance].[AccountSub]  WITH CHECK ADD  CONSTRAINT                           ↙
    [FK_AccountSub_Universities] FOREIGN KEY([UniversityCode])
REFERENCES [SetUp].[Universities] ([Code])
```

```
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [Finance].[AccountSub] CHECK CONSTRAINT [FK_AccountSub_Universities]
GO
/****** Object:  ForeignKey [FK_AccountTypes_Universities]    Script Date: 08/16/2011 01: ↙
    24:32 ******/
ALTER TABLE [Finance].[AccountTypes]  WITH CHECK ADD  CONSTRAINT                          ↙
    [FK_AccountTypes_Universities] FOREIGN KEY([UniversityCode])
REFERENCES [SetUp].[Universities] ([Code])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [Finance].[AccountTypes] CHECK CONSTRAINT [FK_AccountTypes_Universities]
GO
/****** Object:  ForeignKey [FK_FeeAllotment_Sessions]    Script Date: 08/16/2011 01:24:32↙
     ******/
ALTER TABLE [Finance].[FeeAllotment]  WITH CHECK ADD  CONSTRAINT                          ↙
    [FK_FeeAllotment_Sessions] FOREIGN KEY([SessionCode])
REFERENCES [Academics].[Sessions] ([Code])
GO
ALTER TABLE [Finance].[FeeAllotment] CHECK CONSTRAINT [FK_FeeAllotment_Sessions]
GO
/****** Object:  ForeignKey [FK_FeeAllotment_Universities]    Script Date: 08/16/2011 01: ↙
    24:32 ******/
ALTER TABLE [Finance].[FeeAllotment]  WITH CHECK ADD  CONSTRAINT                          ↙
    [FK_FeeAllotment_Universities] FOREIGN KEY([UniversityCode])
REFERENCES [SetUp].[Universities] ([Code])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [Finance].[FeeAllotment] CHECK CONSTRAINT [FK_FeeAllotment_Universities]
GO
/****** Object:  ForeignKey [FK_FeeDefinition_Universities]    Script Date: 08/16/2011 01:↙
    24:32 ******/
ALTER TABLE [Finance].[FeeDefinition]  WITH CHECK ADD  CONSTRAINT                         ↙
    [FK_FeeDefinition_Universities] FOREIGN KEY([UniversityCode])
REFERENCES [SetUp].[Universities] ([Code])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [Finance].[FeeDefinition] CHECK CONSTRAINT [FK_FeeDefinition_Universities]
GO
/****** Object:  ForeignKey [FK_Fees_Sessions]    Script Date: 08/16/2011 01:24:32 ******/
ALTER TABLE [Finance].[Fees]  WITH CHECK ADD  CONSTRAINT [FK_Fees_Sessions] FOREIGN KEY(   ↙
    [SessionCode])
REFERENCES [Academics].[Sessions] ([Code])
GO
ALTER TABLE [Finance].[Fees] CHECK CONSTRAINT [FK_Fees_Sessions]
GO
/****** Object:  ForeignKey [FK_Fees_Universities]    Script Date: 08/16/2011 01:24:32 ***↙
    ***/
ALTER TABLE [Finance].[Fees]  WITH CHECK ADD  CONSTRAINT [FK_Fees_Universities] FOREIGN   ↙
    KEY([UniversityCode])
REFERENCES [SetUp].[Universities] ([Code])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [Finance].[Fees] CHECK CONSTRAINT [FK_Fees_Universities]
GO
/****** Object:  ForeignKey [FK_FeesDetail_Fees]    Script Date: 08/16/2011 01:24:32 *****↙
    */
ALTER TABLE [Finance].[FeesDetail]  WITH CHECK ADD  CONSTRAINT [FK_FeesDetail_Fees]       ↙
    FOREIGN KEY([FeesCode])
REFERENCES [Finance].[Fees] ([Code])
ON UPDATE CASCADE
ON DELETE CASCADE
```

```
GO
ALTER TABLE [Finance].[FeesDetail] CHECK CONSTRAINT [FK_FeesDetail_Fees]
GO
/****** Object:  ForeignKey [FK_PaymentPlanDetails_PaymentPlans]    Script Date: 08/16/ ↙
    2011 01:24:32 ******/
ALTER TABLE [Finance].[PaymentPlanDetails]  WITH CHECK ADD  CONSTRAINT                ↙
    [FK_PaymentPlanDetails_PaymentPlans] FOREIGN KEY([PaymentPlanCode])
REFERENCES [Finance].[PaymentPlans] ([Code])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [Finance].[PaymentPlanDetails] CHECK CONSTRAINT                           ↙
    [FK_PaymentPlanDetails_PaymentPlans]
GO
/****** Object:  ForeignKey [FK_PaymentPlans_Sessions]    Script Date: 08/16/2011 01:24:32↙
     ******/
ALTER TABLE [Finance].[PaymentPlans]  WITH CHECK ADD  CONSTRAINT                      ↙
    [FK_PaymentPlans_Sessions] FOREIGN KEY([SessionCode])
REFERENCES [Academics].[Sessions] ([Code])
GO
ALTER TABLE [Finance].[PaymentPlans] CHECK CONSTRAINT [FK_PaymentPlans_Sessions]
GO
/****** Object:  ForeignKey [FK_PaymentPlans_Universities]    Script Date: 08/16/2011 01: ↙
    24:32 ******/
ALTER TABLE [Finance].[PaymentPlans]  WITH CHECK ADD  CONSTRAINT                      ↙
    [FK_PaymentPlans_Universities] FOREIGN KEY([UniversityCode])
REFERENCES [SetUp].[Universities] ([Code])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [Finance].[PaymentPlans] CHECK CONSTRAINT [FK_PaymentPlans_Universities]
GO
/****** Object:  ForeignKey [FK_Refunds_Sessions]    Script Date: 08/16/2011 01:24:32 ****↙
    **/
ALTER TABLE [Finance].[Refunds]  WITH CHECK ADD  CONSTRAINT [FK_Refunds_Sessions] FOREIGN ↙
    KEY([SessionCode])
REFERENCES [Academics].[Sessions] ([Code])
GO
ALTER TABLE [Finance].[Refunds] CHECK CONSTRAINT [FK_Refunds_Sessions]
GO
/****** Object:  ForeignKey [FK_Refunds_Universities]    Script Date: 08/16/2011 01:24:32 ↙
    ******/
ALTER TABLE [Finance].[Refunds]  WITH CHECK ADD  CONSTRAINT [FK_Refunds_Universities]     ↙
    FOREIGN KEY([UniversityCode])
REFERENCES [SetUp].[Universities] ([Code])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [Finance].[Refunds] CHECK CONSTRAINT [FK_Refunds_Universities]
GO
/****** Object:  ForeignKey [FK_Transactions_Sessions]    Script Date: 08/16/2011 01:24:32↙
     ******/
ALTER TABLE [Finance].[Transactions]  WITH CHECK ADD  CONSTRAINT                      ↙
    [FK_Transactions_Sessions] FOREIGN KEY([SessionCode])
REFERENCES [Academics].[Sessions] ([Code])
GO
ALTER TABLE [Finance].[Transactions] CHECK CONSTRAINT [FK_Transactions_Sessions]
GO
/****** Object:  ForeignKey [FK_Transactions_Universities]    Script Date: 08/16/2011 01: ↙
    24:32 ******/
ALTER TABLE [Finance].[Transactions]  WITH CHECK ADD  CONSTRAINT                      ↙
    [FK_Transactions_Universities] FOREIGN KEY([UniversityCode])
REFERENCES [SetUp].[Universities] ([Code])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [Finance].[Transactions] CHECK CONSTRAINT [FK_Transactions_Universities]
```

```
GO
/****** Object:  ForeignKey [FK_Vouchers_Sessions]    Script Date: 08/16/2011 01:24:32 ***↙
    ***/
ALTER TABLE [Finance].[Vouchers]  WITH CHECK ADD  CONSTRAINT [FK_Vouchers_Sessions]      ↙
    FOREIGN KEY([SessionCode])
REFERENCES [Academics].[Sessions] ([Code])
GO
ALTER TABLE [Finance].[Vouchers] CHECK CONSTRAINT [FK_Vouchers_Sessions]
GO
/****** Object:  ForeignKey [FK_Vouchers_Universities]    Script Date: 08/16/2011 01:24:32↙
     ******/
ALTER TABLE [Finance].[Vouchers]  WITH CHECK ADD  CONSTRAINT [FK_Vouchers_Universities]  ↙
    FOREIGN KEY([UniversityCode])
REFERENCES [SetUp].[Universities] ([Code])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [Finance].[Vouchers] CHECK CONSTRAINT [FK_Vouchers_Universities]
GO
/****** Object:  ForeignKey [FK_BioDataReligion]    Script Date: 08/16/2011 01:24:32 *****↙
    */
ALTER TABLE [Personals].[Religions]  WITH CHECK ADD  CONSTRAINT [FK_BioDataReligion]      ↙
    FOREIGN KEY([BioDataCode])
REFERENCES [Personals].[BioDatas] ([Code])
GO
ALTER TABLE [Personals].[Religions] CHECK CONSTRAINT [FK_BioDataReligion]
GO
/****** Object:  ForeignKey [FK_Admissions_Sessions]    Script Date: 08/16/2011 01:24:32 *↙
    *****/
ALTER TABLE [Registry].[Admissions]  WITH CHECK ADD  CONSTRAINT [FK_Admissions_Sessions]  ↙
    FOREIGN KEY([SessionCode])
REFERENCES [Academics].[Sessions] ([Code])
GO
ALTER TABLE [Registry].[Admissions] CHECK CONSTRAINT [FK_Admissions_Sessions]
GO
/****** Object:  ForeignKey [FK_Admissions_Universities]    Script Date: 08/16/2011 01:24:↙
    32 ******/
ALTER TABLE [Registry].[Admissions]  WITH CHECK ADD  CONSTRAINT                           ↙
    [FK_Admissions_Universities] FOREIGN KEY([UniversityCode])
REFERENCES [SetUp].[Universities] ([Code])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [Registry].[Admissions] CHECK CONSTRAINT [FK_Admissions_Universities]
GO
/****** Object:  ForeignKey [FK_QualifiedForExams_Universities]    Script Date: 08/16/2011↙
     01:24:32 ******/
ALTER TABLE [Registry].[Exams]  WITH CHECK ADD  CONSTRAINT                                ↙
    [FK_QualifiedForExams_Universities] FOREIGN KEY([UniversityCode])
REFERENCES [SetUp].[Universities] ([Code])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [Registry].[Exams] CHECK CONSTRAINT [FK_QualifiedForExams_Universities]
GO
/****** Object:  ForeignKey [FK_Students_Universities]    Script Date: 08/16/2011 01:24:32↙
     ******/
ALTER TABLE [Registry].[Students]  WITH CHECK ADD  CONSTRAINT [FK_Students_Universities]  ↙
    FOREIGN KEY([UniversityCode])
REFERENCES [SetUp].[Universities] ([Code])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [Registry].[Students] CHECK CONSTRAINT [FK_Students_Universities]
GO
/****** Object:  ForeignKey [FK_Tickets_Sessions]    Script Date: 08/16/2011 01:24:32 ****↙
    **/
```

```sql
ALTER TABLE [Registry].[Tickets]  WITH CHECK ADD  CONSTRAINT [FK_Tickets_Sessions] FOREIGN KEY([SessionCode])
REFERENCES [Academics].[Sessions] ([Code])
GO
ALTER TABLE [Registry].[Tickets] CHECK CONSTRAINT [FK_Tickets_Sessions]
GO
/****** Object:  ForeignKey [FK_Tickets_Universities]    Script Date: 08/16/2011 01:24:32 ******/
ALTER TABLE [Registry].[Tickets]  WITH CHECK ADD  CONSTRAINT [FK_Tickets_Universities] FOREIGN KEY([UniversityCode])
REFERENCES [SetUp].[Universities] ([Code])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [Registry].[Tickets] CHECK CONSTRAINT [FK_Tickets_Universities]
GO
/****** Object:  ForeignKey [FK_Approvals_Universities]    Script Date: 08/16/2011 01:24:32 ******/
ALTER TABLE [SetUp].[Approvals]  WITH CHECK ADD  CONSTRAINT [FK_Approvals_Universities] FOREIGN KEY([UniversityCode])
REFERENCES [SetUp].[Universities] ([Code])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [SetUp].[Approvals] CHECK CONSTRAINT [FK_Approvals_Universities]
GO
/****** Object:  ForeignKey [FK_CourseNumbering_Universities]    Script Date: 08/16/2011 01:24:32 ******/
ALTER TABLE [SetUp].[CourseNumbering]  WITH CHECK ADD  CONSTRAINT [FK_CourseNumbering_Universities] FOREIGN KEY([UniversityCode])
REFERENCES [SetUp].[Universities] ([Code])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [SetUp].[CourseNumbering] CHECK CONSTRAINT [FK_CourseNumbering_Universities]
GO
/****** Object:  ForeignKey [FK_Courses_Departments]    Script Date: 08/16/2011 01:24:32 ******/
ALTER TABLE [SetUp].[Courses]  WITH CHECK ADD  CONSTRAINT [FK_Courses_Departments] FOREIGN KEY([DepartmentCode])
REFERENCES [SetUp].[Departments] ([Code])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [SetUp].[Courses] CHECK CONSTRAINT [FK_Courses_Departments]
GO
/****** Object:  ForeignKey [FK_Departments_Faculties]    Script Date: 08/16/2011 01:24:32 ******/
ALTER TABLE [SetUp].[Departments]  WITH CHECK ADD  CONSTRAINT [FK_Departments_Faculties] FOREIGN KEY([FacultyCode])
REFERENCES [SetUp].[Faculties] ([Code])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [SetUp].[Departments] CHECK CONSTRAINT [FK_Departments_Faculties]
GO
/****** Object:  ForeignKey [FK_ParametersDescriptions]    Script Date: 08/16/2011 01:24:32 ******/
ALTER TABLE [SetUp].[Descriptions]  WITH CHECK ADD  CONSTRAINT [FK_ParametersDescriptions] FOREIGN KEY([ParametersCode])
REFERENCES [SetUp].[Parameters] ([Code])
GO
ALTER TABLE [SetUp].[Descriptions] CHECK CONSTRAINT [FK_ParametersDescriptions]
GO
/****** Object:  ForeignKey [FK_Designations_Staff]    Script Date: 08/16/2011 01:24:32 ******/
ALTER TABLE [SetUp].[Designations]  WITH CHECK ADD  CONSTRAINT [FK_Designations_Staff]
```

```
    FOREIGN KEY([StaffCode])
REFERENCES [SetUp].[Staff] ([Code])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [SetUp].[Designations] CHECK CONSTRAINT [FK_Designations_Staff]
GO
/****** Object:  ForeignKey [FK_Durations_Programs]    Script Date: 08/16/2011 01:24:32 **↙
    ****/
ALTER TABLE [SetUp].[Durations]  WITH CHECK ADD  CONSTRAINT [FK_Durations_Programs]       ↙
    FOREIGN KEY([ProgramCode])
REFERENCES [SetUp].[Programs] ([Code])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [SetUp].[Durations] CHECK CONSTRAINT [FK_Durations_Programs]
GO
/****** Object:  ForeignKey [FK_Faculties_Universities]    Script Date: 08/16/2011 01:24: ↙
    32 ******/
ALTER TABLE [SetUp].[Faculties]  WITH CHECK ADD  CONSTRAINT [FK_Faculties_Universities]   ↙
    FOREIGN KEY([UniversityCode])
REFERENCES [SetUp].[Universities] ([Code])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [SetUp].[Faculties] CHECK CONSTRAINT [FK_Faculties_Universities]
GO
/****** Object:  ForeignKey [FK_GradingSystem_Universities]    Script Date: 08/16/2011 01:↙
    24:32 ******/
ALTER TABLE [SetUp].[GradingSystem]  WITH CHECK ADD  CONSTRAINT                           ↙
    [FK_GradingSystem_Universities] FOREIGN KEY([UniversityCode])
REFERENCES [SetUp].[Universities] ([Code])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [SetUp].[GradingSystem] CHECK CONSTRAINT [FK_GradingSystem_Universities]
GO
/****** Object:  ForeignKey [FK_CountriesLGA]    Script Date: 08/16/2011 01:24:32 ******/
ALTER TABLE [SetUp].[LGAs]  WITH CHECK ADD  CONSTRAINT [FK_CountriesLGA] FOREIGN KEY(     ↙
    [CountriesCode])
REFERENCES [SetUp].[Countries] ([Code])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [SetUp].[LGAs] CHECK CONSTRAINT [FK_CountriesLGA]
GO
/****** Object:  ForeignKey [FK_StatesLGA]    Script Date: 08/16/2011 01:24:32 ******/
ALTER TABLE [SetUp].[LGAs]  WITH CHECK ADD  CONSTRAINT [FK_StatesLGA] FOREIGN KEY(        ↙
    [StatesCode])
REFERENCES [SetUp].[States] ([Code])
GO
ALTER TABLE [SetUp].[LGAs] CHECK CONSTRAINT [FK_StatesLGA]
GO
/****** Object:  ForeignKey [FK_Modules_Universities]    Script Date: 08/16/2011 01:24:32 ↙
    ******/
ALTER TABLE [SetUp].[Modules]  WITH CHECK ADD  CONSTRAINT [FK_Modules_Universities]       ↙
    FOREIGN KEY([UniversityCode])
REFERENCES [SetUp].[Universities] ([Code])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [SetUp].[Modules] CHECK CONSTRAINT [FK_Modules_Universities]
GO
/****** Object:  ForeignKey [FK_Programs_Courses]    Script Date: 08/16/2011 01:24:32 ****↙
    **/
ALTER TABLE [SetUp].[Programs]  WITH CHECK ADD  CONSTRAINT [FK_Programs_Courses] FOREIGN  ↙
    KEY([CourseCode])
```

```
REFERENCES [SetUp].[Courses] ([Code])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [SetUp].[Programs] CHECK CONSTRAINT [FK_Programs_Courses]
GO
/****** Object:  ForeignKey [FK_Screens_Screens]    Script Date: 08/16/2011 01:24:32 *****↙
    */
ALTER TABLE [SetUp].[Screens]  WITH CHECK ADD  CONSTRAINT [FK_Screens_Screens] FOREIGN KEY↙
    ([ModuleCode])
REFERENCES [SetUp].[Modules] ([Code])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [SetUp].[Screens] CHECK CONSTRAINT [FK_Screens_Screens]
GO
/****** Object:  ForeignKey [FK_Staff_BioDatas]    Script Date: 08/16/2011 01:24:32 ******↙
    /
ALTER TABLE [SetUp].[Staff]  WITH CHECK ADD  CONSTRAINT [FK_Staff_BioDatas] FOREIGN KEY(   ↙
    [AccountCode])
REFERENCES [Personals].[BioDatas] ([Code])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [SetUp].[Staff] CHECK CONSTRAINT [FK_Staff_BioDatas]
GO
/****** Object:  ForeignKey [FK_Staff_Universities]    Script Date: 08/16/2011 01:24:32 **↙
    ****/
ALTER TABLE [SetUp].[Staff]  WITH CHECK ADD  CONSTRAINT [FK_Staff_Universities] FOREIGN   ↙
    KEY([UniversityCode])
REFERENCES [SetUp].[Universities] ([Code])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [SetUp].[Staff] CHECK CONSTRAINT [FK_Staff_Universities]
GO
/****** Object:  ForeignKey [FK_CountriesStates]    Script Date: 08/16/2011 01:24:32 *****↙
    */
ALTER TABLE [SetUp].[States]  WITH CHECK ADD  CONSTRAINT [FK_CountriesStates] FOREIGN KEY(↙
    [CountriesCode])
REFERENCES [SetUp].[Countries] ([Code])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [SetUp].[States] CHECK CONSTRAINT [FK_CountriesStates]
GO
/****** Object:  ForeignKey [FK_SubCourses_CourseNumbering]    Script Date: 08/16/2011 01:↙
    24:32 ******/
ALTER TABLE [SetUp].[SubCourses]  WITH CHECK ADD  CONSTRAINT                              ↙
    [FK_SubCourses_CourseNumbering] FOREIGN KEY([LevelCode])
REFERENCES [SetUp].[CourseNumbering] ([Code])
GO
ALTER TABLE [SetUp].[SubCourses] CHECK CONSTRAINT [FK_SubCourses_CourseNumbering]
GO
/****** Object:  ForeignKey [FK_SubjectRequirementDetails_SubjectRequirements]    Script  ↙
    Date: 08/16/2011 01:24:32 ******/
ALTER TABLE [SetUp].[SubjectRequirementDetails]  WITH CHECK ADD  CONSTRAINT               ↙
    [FK_SubjectRequirementDetails_SubjectRequirements] FOREIGN KEY(                       ↙
    [SubjectRequirementCode])
REFERENCES [SetUp].[SubjectRequirements] ([Code])
ON UPDATE CASCADE
ON DELETE CASCADE
GO
ALTER TABLE [SetUp].[SubjectRequirementDetails] CHECK CONSTRAINT                          ↙
    [FK_SubjectRequirementDetails_SubjectRequirements]
GO
```