**Database Design**

CREATE TABLE tbl\_Student

(

SSO INT PRIMARY KEY,

StudentName VARCHAR(50),

Gender CHAR(1) CHECK (Gender = 'M' OR Gender = 'F'),

Degree VARCHAR(10),

Email VARCHAR(50),

ContactNumber BIGINT,

DOB Date,

Passwd VARCHAR(50),

)

DROP TABLE tbl\_Student

INSERT INTO tbl\_Student VALUES(12426434,'Sumanth Gunda','M','Masters','sgnw8@mail.umkc.edu',8166823758,'1993-07-30','UMKC@Sum@nth4032')

INSERT INTO tbl\_Student VALUES(16186322,'Goutham Donthu','M','Masters','gd6d9@mail.umkc.edu',8166823463,'03-27-1992','gou')

SELECT \* FROM tbl\_Student

CREATE TABLE tbl\_Instructor

(

InstructorId INT PRIMARY KEY,

InstructorName VARCHAR(50),

Gender CHAR(1) CHECK (Gender = 'M' OR Gender = 'F'),

Email VARCHAR(50),

ContactNumber BIGINT,

DOB Date,

Passwd VARCHAR(50),

)

SELECT \* FROM tbl\_Instructor

DROP TABLE tbl\_Instructor

INSERT INTO tbl\_Instructor VALUES(111,'BaekYoungChoi','F','choiby@mail.umkc.edu',9999999999,'1983-07-30','choi')

CREATE PROCEDURE usp\_Login(@Email VARCHAR(50),@Passwd VARCHAR(50))

AS

BEGIN

BEGIN TRY

IF EXISTS(SELECT Email FROM tbl\_Student WHERE Email = @Email)

BEGIN

IF EXISTS(SELECT Email,Passwd FROM tbl\_Student WHERE Email = @Email AND Passwd = @Passwd)

BEGIN

RETURN 1

END

ELSE

BEGIN

RETURN 2

END

END

ELSE IF EXISTS(SELECT Email FROM tbl\_Instructor WHERE Email = @Email)

BEGIN

IF EXISTS(SELECT Email,Passwd FROM tbl\_Instructor WHERE Email = @Email AND Passwd = @Passwd)

BEGIN

RETURN 3

END

ELSE

BEGIN

RETURN 4

END

END

ELSE

BEGIN

RETURN 5

END

END TRY

BEGIN CATCH

RETURN -99

END CATCH

END

CREATE PROCEDURE usp\_LoginService(@Name VARCHAR(50),@Passwd VARCHAR(50))

AS

BEGIN

BEGIN TRY

IF EXISTS(SELECT StudentName FROM tbl\_Student WHERE StudentName = @Name)

BEGIN

IF EXISTS(SELECT StudentName,Passwd FROM tbl\_Student WHERE StudentName = @Name AND Passwd = @Passwd)

BEGIN

RETURN 1

END

ELSE

BEGIN

RETURN 2

END

END

ELSE IF EXISTS(SELECT InstructorName FROM tbl\_Instructor WHERE InstructorName = @Name)

BEGIN

IF EXISTS(SELECT InstructorName,Passwd FROM tbl\_Instructor WHERE InstructorName = @Name AND Passwd = @Passwd)

BEGIN

RETURN 3

END

ELSE

BEGIN

RETURN 4

END

END

ELSE

BEGIN

RETURN 5

END

END TRY

BEGIN CATCH

RETURN -99

END CATCH

END

CREATE TABLE tbl\_SetAppointments

(

InstructorId INT CONSTRAINT srkey foreign key references tbl\_Instructor(InstructorId),

AppointmentDate DATE,

FromTime TIME(0),

ToTime TIME(0),

AppointmentDuration INT,

MaxAppointments INT,

PRIMARY KEY(InstructorId,AppointmentDate,FromTime,ToTime)

)

CREATE TABLE tbl\_AppointmentRequests

(

AppointmentId INT IDENTITY(1,1) PRIMARY KEY,

SSO INT CONSTRAINT srkey1 foreign key references tbl\_Student(SSO),

AppointmentDate DATE,

InstructorName VARCHAR(50),

AppointmentTime VARCHAR(50),

AppointmentType VARCHAR(50),

Description VARCHAR(200),

AppointmentStatus VARCHAR(20)

)

SELECT \* FROM tbl\_AppointmentRequests

DROP TABLE tbl\_AppointmentRequests

INSERT INTO tbl\_SetAppointments VALUES(111,'03-08-2015','05-08-2015','1:00','2:00',10,4)

SELECT \* FROM tbl\_SetAppointments

DROP TABLE tbl\_SetAppointments

CREATE PROCEDURE usp\_GetInstructorId(@Email VARCHAR(50),@InstructorId INT OUT)

AS

BEGIN

SET @InstructorId = 1

BEGIN TRY

SET @InstructorId = (SELECT InstructorId FROm tbl\_Instructor WHERE Email = @Email)

RETURN 1

END TRY

BEGIN CATCH

RETURN -1

END CATCH

END

CREATE PROCEDURE usp\_GetStudentId(@Email VARCHAR(50),@SSO INT OUT)

AS

BEGIN

SET @SSO = 1

BEGIN TRY

SET @SSO = (SELECT SSO FROm tbl\_Student WHERE Email = @Email)

RETURN 1

END TRY

BEGIN CATCH

RETURN -1

END CATCH

END

DROP PROCEDURE usp\_GetStudentId

CREATE PROCEDURE usp\_GetInstructorIdByname(@InstructorName VARCHAR(50),@InstructorId INT OUT)

AS

BEGIN

SET @InstructorId = 1

BEGIN TRY

SET @InstructorId = (SELECT InstructorId FROM tbl\_Instructor WHERE InstructorName = @InstructorName)

RETURN 1

END TRY

BEGIN CATCH

RETURN -1

END CATCH

END

ALTER PROCEDURE usp\_GetInstructorNameByEmail(@Email VARCHAR(50),@InstructorName VARCHAR(50) OUT)

AS

BEGIN

SET @InstructorName = 'Instructor'

BEGIN TRY

SET @InstructorName = (SELECT InstructorName FROM tbl\_Instructor WHERE Email = @Email)

RETURN 1

END TRY

BEGIN CATCH

RETURN -1

END CATCH

END

ALTER PROCEDURE usp\_SetAppointments(@InstructorId INT,@FromDate DATE,@ToDate DATE,@FromTime VARCHAR(20),@ToTime VARCHAR(20),@AppointmentDuration INT,@MaxAppointments INT)

AS

BEGIN

DECLARE @AppointmentDate DATE

DECLARE @Name VARCHAR(10)

SET @AppointmentDate = @FromDate

BEGIN TRY

WHILE(@AppointmentDate <= @ToDate)

BEGIN

SET @Name = (SELECT DATENAME(weekday,@AppointmentDate))

IF (@Name = 'Sunday' OR @Name = 'Saturday')

BEGIN

SET @AppointmentDate = (SELECT CAST((SELECT DATEADD(DAY,1,@AppointmentDate)) AS DATE))

END

ELSE

BEGIN

INSERT INTO tbl\_SetAppointments(InstructorId,AppointmentDate,FromTime,ToTime,AppointmentDuration,MaxAppointments) VALUES(@InstructorId,@AppointmentDate,@FromTime,@ToTime,@AppointmentDuration,@MaxAppointments)

SET @AppointmentDate = (SELECT CAST((SELECT DATEADD(DAY,1,@AppointmentDate)) AS DATE))

END

END

RETURN 1

END TRY

BEGIN CATCH

RETURN -99

END CATCH

END

SELECT DATENAME(weekday,'2015-03-07')

SELECT CAST((SELECT DATEADD(DAY,1,'2015-03-01')) AS DATE)

DROP PROCEDURE usp\_SetAppointments

SELECT AppointmentDate FROM tbl\_SetAppointments WHERE AppointmentDate = '2015-03-04' AND FromTime = '2:00' AND ToTime = '3:00'

SELECT t.SSO,t.StudentName,a.AppointmentDate,a.AppointmentTime,a.AppointmentType,a.Description,a.AppointmentStatus FROM tbl\_Student t INNER JOIN tbl\_AppointmentRequests a ON t.SSO = a.SSO AND a.InstructorName = 'BaekYoungChoi'

CREATE PROCEDURE AppRejAppointments(@AppointmentId INT,@StatusUpdate VARCHAR(50))

AS

BEGIN

BEGIN TRY

IF(@StatusUpdate = 'Approved')

BEGIN

UPDATE tbl\_AppointmentRequests SET AppointmentStatus = @StatusUpdate WHERE AppointmentId = @AppointmentId

RETURN 1

END

ELSE IF(@StatusUpdate = 'Rejected')

BEGIN

UPDATE tbl\_AppointmentRequests SET AppointmentStatus = @StatusUpdate WHERE AppointmentId = @AppointmentId

RETURN 2

END

END TRY

BEGIN CATCH

RETURN -99

END CATCH

END