APPENDIX 1 CODE

VIEWS

/\*/\* ---------------------------------------------------------------------------------------------------------- \*

\* Script File Name : vw\_TotalClassesPerMember \*

\* Programmer Name : Tejas Dwarkaram \*

\* Date : 2012.05.07 \*

\* Description : View total classes per member \*

\* ---------------------------------------------------------------------------------------------------------- \*/

\*/USE Suzi\_Yoga\_Studio

GO

/\*

CREATE VIEW vw\_TotalClassesPerMember

AS

SELECT FoKOL

GO\*/

/\* ---------------------------------------------------------------------------------------------------------- \*

\* Script File Name : vw\_ExerciseUsed \*

\* Programmer Name : Tejas Dwarkaram \*

\* Date : 2012.05.07 \*

\* Description : View the most commonly used \*

\* ---------------------------------------------------------------------------------------------------------- \*/

CREATE VIEW vw\_ExerciseUsed

AS

SELECT name, description, length\_Of\_Exercise, number\_Of\_Times\_To\_Be\_Done

FROM Exercises

GO

/\* ---------------------------------------------------------------------------------------------------------- \*

\* Script File Name : vw\_Exercises \*

\* Programmer Name : Tejas Dwarkaram \*

\* Date : 2012.05.07 \*

\* Description : View title, name, description, and time of exercises to be \*

\* ---------------------------------------------------------------------------------------------------------- \*/

CREATE VIEW vw\_Exercises

AS

SELECT title, name, description, number\_Of\_Times\_To\_Be\_Done

FROM Exercise\_Books JOIN Exercises

ON Exercise\_Books.book\_ID = Exercises.book\_ID

GO

/\* ---------------------------------------------------------------------------------------------------------- \*

\* Script File Name : vw\_ClassAttendance \*

\* Programmer Name : Tejas Dwarkaram \*

\* Date : 2012.05.07 \*

\* Description : View the class attendance of each \*

\* ---------------------------------------------------------------------------------------------------------- \*/

CREATE VIEW vw\_ClassAttendance

AS

SELECT TOP 100 members.class\_code, class\_sessions.day,class\_sessions.time\_Of\_Session, class\_sessions.studio\_Number

FROM members,class\_sessions

WHERE members.class\_code = class\_sessions.class\_code

ORDER BY members.class\_code DESC

GO

SELECT \*

FROM vw\_ClassAttendance

TRIGGERS

/\* ---------------------------------------------------------------------------------------------------------- \*

\* Script File Name : trg\_underage \*

\* Programmer Name : Tejas Dwarkaram \*

\* Date : 2012.05.10 \*

\* Description : creating a trigger to check the age of the new member \*

\* ----------------------------------------------------------------------------------------------------------- \*/

USE Suzi\_Yoga\_Studio

GO

CREATE TRIGGER young\_person

ON Members AFTER INSERT

AS IF UPDATE(date\_of\_birth)

BEGIN

DECLARE @age INT

DECLARE @new\_Birth DATE

SET @new\_Birth = (SELECT date\_of\_birth FROM inserted)

SET @age = (YEAR(GETDATE())-(YEAR(@new\_Birth)))

IF(@age<16)

BEGIN

RAISERROR('You are not of the appropriate age as yet.Only over 16 allowed.',16,10)

ROLLBACK TRANSACTION

END

END

GO

/\* ---------------------------------------------------------------------------------------------------------- \*

\* Script File Name : trg\_cancellation \*

\* Programmer Name : Tejas Dwarkaram \*

\* Date : 2012.05.10 \*

\* Description : creating a trigger to inform suzi of the members that need to be informed if there \*

\* is a cancellation \*

\* ----------------------------------------------------------------------------------------------------------- \*/

CREATE TRIGGER inform\_cancel

ON class\_sessions AFTER UPDATE

AS IF UPDATE(cancelled)

DECLARE @cancelYes VARCHAR(1)

DECLARE @classcode VARCHAR(6)

SET @cancelYes = (SELECT cancelled FROM inserted)

SET @classcode = (SELECT class\_code FROM inserted)

BEGIN

IF(@cancelYes = 'Y')

BEGIN

SELECT name, cell\_Number, email\_Address

INTO #tempTable

FROM members, class\_sessions

WHERE members.class\_code = @classcode

END

END

PRINT 'The following members need to be informed of the cancellation'

SELECT \*

FROM #tempTable

GO

PROCEDURES

/\* ---------------------------------------------------------------------------------------------------------- \*

\* Script File Name : sp\_UpdateExerciseTimeUsed \*

\* Programmer Name : Tejas Dwarkaram \*

\* Date : 2012.05.03 \*

\* Description : Stored Procedure to updare the number of times a certain exercise must be used \*

\* ----------------------------------------------------------------------------------------------------------- \*/

USE Suzi\_Yoga\_Studio

GO

CREATE PROCEDURE sp\_UpdateExerciseTimeUsed(@book\_ID VARCHAR(30), @times INT)

AS

UPDATE Exercises

SET number\_Of\_Times\_To\_Be\_Done = @times

WHERE book\_ID = @book\_ID

GO

/\* ---------------------------------------------------------------------------------------------------------- \*

\* Script File Name : sp\_Report \*

\* Programmer Name : Tejas Dwarkaram \*

\* Date : 2012.05.07 \*

\* Description : Get the report for a certin class \*

\* ---------------------------------------------------------------------------------------------------------- \*/

CREATE PROCEDURE sp\_Report(@class\_code VARCHAR(6))

AS

DECLARE @day VARCHAR(10)

DECLARE @studio\_Number VARCHAR(10)

DECLARE @name VARCHAR(45)

DECLARE @cell VARCHAR(10)

DECLARE @time VARCHAR(15)

DECLARE @counter INT

DECLARE @totalMembers INT

DECLARE @auto VARCHAR (2)

DECLARE @memNum INT

DECLARE @mID INT

SET @day = (SELECT day FROM class\_sessions WHERE @class\_code = class\_sessions.class\_code )

SET @time = (SELECT time\_Of\_Session FROM class\_sessions WHERE @class\_code = class\_sessions.class\_code )

SET @studio\_Number = (SELECT studio\_Number FROM class\_sessions WHERE @class\_code = class\_sessions.class\_code )

PRINT'YOGA CLASS REPORT:'

PRINT'\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_'

PRINT'Class Code: ' + @class\_code

PRINT'Week Day: ' + @day + ' [Time: ' + @time + ']'

PRINT'Studio Number ' + @studio\_Number

PRINT''

PRINT'No. Member name Contact number'

PRINT'\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_'

--SELECT name, cell\_Number

--FROM members

--WHERE members.class\_code = @class\_code

SELECT @totalMembers = COUNT(members.class\_code)

FROM members

WHERE members.class\_code = @class\_code

SET @auto = 1

SET @counter = 0

SELECT @mID = (members.member\_Number)

FROM members

SELECT \*

FROM members

WHERE member\_Number = @mID

WHILE @counter<@totalMembers

BEGIN

SELECT @name = members.name, @cell = members.cell\_Number

FROM members

WHERE members.class\_code = @class\_code

Print @auto +' ' + @name + ' '+ @cell

SET @auto = @auto + 1

SET @counter = @counter + 1

END

GO

/\* ---------------------------------------------------------------------------------------------------------- \*

\* Script File Name : sp\_DeleteBook \*

\* Programmer Name : Tejas Dwarkaram \*

\* Date : 2012.04.26 \*

\* Description : Stored Procedure to delete a book record from the exercise books table \*

\* ----------------------------------------------------------------------------------------------------------- \*/

CREATE PROCEDURE sp\_DeleteBook(@book\_ID VARCHAR(30))

AS

IF(@book\_ID = (SELECT book\_ID FROM vw\_Exercises))

BEGIN

PRINT'The book that needs to be deleted cannot be deleted at this time.'

END

ELSE

BEGIN

DELETE FROM Exercise\_Books

WHERE book\_ID = @book\_ID

END

GO

/\* ---------------------------------------------------------------------------------------------------------- \*

\* Script File Name : sp\_AddNewExercise \*

\* Programmer Name : Tejas Dwarkaram \*

\* Date : 2012.05.03 \*

\* Description : Stored Procedure to add a new exercise record to the Exercises table \*

\* ----------------------------------------------------------------------------------------------------------- \*/

CREATE PROCEDURE sp\_AddNewExercise(@bookID VARCHAR(30), @name VARCHAR(100), @description VARCHAR(800), @length TIME, @timesDone INT)

AS

INSERT INTO Exercises

VALUES (@bookID,@name,@description,@length,@timesDone)

GO

INDEXES

/\* ---------------------------------------------------------------------------------------------------------- \*

\* Script File Name : ind\_name \*

\* Programmer Name : Tejas Dwarkaram \*

\* Date : 2012.05.07 \*

\* Description : Index the title column \*

\* ---------------------------------------------------------------------------------------------------------- \*/

USE Suzi\_Yoga\_Studio

GO

CREATE INDEX ind\_name

ON exercises(name)

GO

DATABASE(inserting)

/\* ---------------------------------------------------------------------------------------------------------- \*

\* Script File Name : insert\_all\_of\_the\_sample\_data \*

\* Programmer Name : Tejas Dwarkaram \*

\* Date : 2012.04.26 \*

\* Description : Inserting rows of data into the tables \*

\* ----------------------------------------------------------------------------------------------------------- \*/

USE Suzi\_Yoga\_Studio

GO

INSERT INTO class\_sessions

VALUES('md02',1,'Monday',3,'03:30:00','N'),

('st40',3,'Wednesday',2,'16:00:00','N'),

('md01',2,'Tuesday',1,'09:30:00','N'),

('f32g',1,'Friday',3,'03:30:00','N')

GO

INSERT INTO members(name,surname,date\_of\_birth,cell\_Number,email\_Address,class\_code,studio\_number,sessions\_Per\_Week)

VALUES('Tejas', 'Dwarkaram', '1993.11.22', '0829424982', 'tejas\_dwarkaram@eml.cc','md02', 3, 3),

('Brandon', 'Rossouw', '1989.05.01', '0835734756', 'brandon.rossouw@gmail.com','f32g', 2, 5),

('Dimitri', 'Gonsalves', '1968.02.28', '0836785985', 'DimzG@mitrimail.com','md02', 3, 2),

('Devin', 'Botha', '1990.03.25', '0715454250', 'deviB@vinmail.com', 'st40', 1, 4),

('Albert', 'Rust', '1989.12.03', '0713254459', 'rust@telkomsa.com', 'st40', 4, 2),

('Nelio', 'Lucas', '1896.01.23', '0829654786', 'nelioL@rbgmail.com','f32g', 3, 6),

('Mamba', 'Samba', '1990.09.03', '0840021454', 'sambam@mamail.com','st40', 1, 3)

GO

INSERT INTO exercise\_Books

VALUES('ks01','Yoga For Life','Bending'),

('sd01','Breathing New Light','Breathing Techniques'),

('ks02','Mantra Healing','Self Realisation'),

('tj98','Meditation through the Universe','Meditation')

GO

INSERT INTO exercises

VALUES('tj98','Rainbow Stretching','This entails the usage of the lower back, in order to stretch the lower

back area. This helps with improving posture, and can help to relieve the syptoms of hunchback.'

,'01:00:00',25),

('ks02','Flamingo Meditation','This meditation routine calms the mind, body and soul. It is done in a

flamingo stance i.e. Standing on one foot at a time, and raising ones hands above their head,

in the form of praying.','00:03:00',4),

('ks02','Aathma Relief','Aathma Relief is done whilst laying flat on ones body, joining ones feet together.

By raising ur feet a little, without splitting them , you allow the flow of blood throughout the system.',

'01:00:00',30),

('tj98','Splitting Bananas','Involving splits in yoga practice allows for improved flexability of the body.

By spontaneously doing splits in 4 directions, the chances of arthirtis is lowered.',

'02:00:00',6),

('sd01', 'Aum Relaxation','This is a meditation technique that soothes the body and allows the body to

communicate together as one.','00:30:00',5),

('ks01','Breath of life','This exercise, if done correctly, opens the artiries and allows for intensiv

deep breathing. This deep breathing helps with circulation of air to the brain, as well increases the flow

of blood to the heart','01:00:00',15),

('sd01','Lotus Flower','Lotus is a sacred indian flower. The purpose of this exercise is to loosen joints

and enhance flexability which can strengthen the muscles. Including biceps, triceps and calf muscles.',

'00:45:00',10)

GO

DATABASe(create)

/\* ---------------------------------------------------------------------------------------------------------- \*

\* Script File Name : Create\_All\_Base \*

\* Programmer Name : Tejas Dwarkaram \*

\* Date : 2012.04.26 \*

\* Description : Creating the database \*

\* ---------------------------------------------------------------------------------------------------------- \*/

USE master

GO

IF EXISTS(SELECT name FROM master.dbo.sysdatabases

WHERE name = 'Suzi\_Yoga\_Studio')

DROP DATABASE Suzi\_Yoga\_Studio

GO

CREATE DATABASE Suzi\_Yoga\_Studio

ON PRIMARY

(

NAME = 'Suzi\_Yoga\_Studio\_Data',

FILENAME = 'C:\Users\Student\Downloads\Suzi\_Yoga\_Studio080512\Main\_Database\Suzi\_Yoga\_Studio.mdf',

SIZE = 10,

MAXSIZE = UNLIMITED,

FILEGROWTH = 2

)

LOG ON

(

NAME = 'Suzi\_Yoga\_Studio\_Log',

FILENAME = 'C:\Users\Student\Downloads\Suzi\_Yoga\_Studio080512\Main\_Database\Suzi\_Yoga\_Studio\_Log.ldf',

SIZE =10,

MAXSIZE = UNLIMITED,

FILEGROWTH = 2

)

GO

/\* ---------------------------------------------------------------------------------------------------------- \*

\* Script File Name : inserting\_tables \*

\* Programmer Name : Tejas Dwarkaram \*

\* Date : 2012.04.26 \*

\* Description : Inserting the tables \*

\* ---------------------------------------------------------------------------------------------------------- \*/

USE Suzi\_Yoga\_Studio

GO

CREATE TABLE class\_sessions

(

class\_code VARCHAR(6) NOT NULL,

week INT NOT NULL,

day VARCHAR(10) NOT NULL,

studio\_Number INT NOT NULL,

time\_Of\_Session TIME NOT NULL,

cancelled VARCHAR(1) DEFAULT 'N',

CHECK(studio\_Number IN(1,2,3,4)),

CONSTRAINT prim\_c\_Code PRIMARY KEY(class\_code)

)

GO

CREATE TABLE members

(

member\_Number INT NOT NULL IDENTITY(1, 1),

name VARCHAR(45) NOT NULL,

surname VARCHAR(45) NOT NULL,

date\_of\_birth DATE NOT NULL,

cell\_Number VARCHAR(10) DEFAULT '0000000000',

email\_Address VARCHAR(90) NULL,

class\_code VARCHAR(6) NOT NULL,

studio\_number INT NOT NULL,

sessions\_Per\_Week INT NULL,

CONSTRAINT prim\_memNum PRIMARY KEY(member\_Number),

CONSTRAINT uniq\_Email UNIQUE(email\_Address),

CONSTRAINT for\_Code FOREIGN KEY(class\_code) REFERENCES class\_sessions(class\_code)

)

GO

CREATE TABLE exercise\_Books

(

book\_ID VARCHAR(30) NOT NULL,

title VARCHAR(90) NOT NULL,

exercise\_Type VARCHAR(90) NOT NULL,

CONSTRAINT prim\_bookID PRIMARY KEY(book\_ID)

)

GO

CREATE TABLE exercises

(

book\_ID VARCHAR(30) NOT NULL,

name VARCHAR(100) NOT NULL,

description VARCHAR(800) NOT NULL,

length\_Of\_Exercise TIME NOT NULL,

number\_Of\_Times\_To\_Be\_Done INT NOT NULL,

CONSTRAINT for\_BookID FOREIGN KEY(book\_ID) REFERENCES Exercise\_Books(book\_ID)

)

GO

APPENDIX 2 USER\_DOCUMENTATION

Table of Contents

[Using SQL Management Studio 15](#_Toc324400595)

[Using the created views 15](#_Toc324400596)

[To view vw\_Exercises 16](#_Toc324400597)

[To view vw\_ClassAttendance 16](#_Toc324400598)

[To view vw\_ExerciseUsed 16](#_Toc324400599)

[To view vw\_TotalClassesPerMember 16](#_Toc324400600)

[Using the created Stored Procedures 17](#_Toc324400601)

[To use sp\_AddNewExercise 18](#_Toc324400602)

[To use sp\_UpdateExerciseTimesUsed 19](#_Toc324400603)

[To use sp\_DeleteBook 20](#_Toc324400604)

[To use sp\_Report 21](#_Toc324400605)

[Author Notes 22](#_Toc324400606)

[Name : 22](#_Toc324400607)

[Surname : 22](#_Toc324400608)

[Date : 22](#_Toc324400609)

[Project Notes 22](#_Toc324400610)

[Purpose 22](#_Toc324400611)

[Description 22](#_Toc324400612)

# Using SQL Management Studio

To begin SQL Management Studio, double click the desktop shortcut

(or if not found on desktop, locate the ‘Microsoft SQL Server 2008’ folder in the start menu)

Once opened connect to the ‘(local)’ server, and then you should be presented with the following layout

To use any of the codes that are going to follow, click on “New Query”(usually found in the top left corner).

Please Note.

To execute a query press “F5”

And to end a query press “Alt” + “Break/Pause” together.

# Using the created views

A view is used to view certain values of any table. What is represented is a kind of filter that contains data that is specified to be shown.

To use a view

A SELECT statement, is used on which we ‘SELECT’ everything from the ‘view container’

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

The following needs to be typed in a ‘New Query Editor Window’.

## To view vw\_Exercises

USE Suzi\_Yoga\_Studio  
GO

SELECT \*  
FROM vw\_Exercises

## To view vw\_ClassAttendance

USE Suzi\_Yoga\_Studio  
GO

SELECT \*  
FROM vw\_ClasAttendance

## To view vw\_ExerciseUsed

USE Suzi\_Yoga\_Studio  
GO

SELECT \*  
FROM vw\_ExerciseUsed

## To view vw\_TotalClassesPerMember

USE Suzi\_Yoga\_Studio  
GO

SELECT \*  
FROM vw\_TotalClassesPerMember

# 

# Using the created Stored Procedures

Stored procedures are also referred to as precompiled statements. This means that after one compilation, it is ready to be reused over and over again.

To use a stored procedure

We use the ‘EXEC’ or ‘EXECUTE’ statement to run a stored procedure. Most stored procedures have certain variables that must be provided for the procedure to function.

Eg.

A stored procedure for adding 2 numbers, will require 2 numbers. Variables are shown with “@” in the front of the variable name.

The format of a stored procedure is usually as follows;

<..procedure\_name..>(@variable1, @variable2)

For example.

addNumbers(@num1, @num2)

So if we were to want to add 5 and 3, the code will look like this

USE Suzi\_Yoga\_Studio  
GO

EXECUTE addNumbers @num1 = 3, @num2 = 5  
GO

And upon execution(pressF5)

The answer ‘8’ will be printed.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

The following needs to be typed in a ‘New Query Editor Window’.

## To use sp\_AddNewExercise

Variables required

@bookID which is the ID of the book

@name which is the name of the book

@description which is the description of the book

@length which is how long the exercise must be done for

@timesdone which is how many times the exercise must be done

What to type in order to execute the procedure

USE Suzi\_Yoga\_Studio  
GO

EXEC sp\_AddNewExercise @bookID=<..‘bookID’..>, @name=<..‘name’..>, @description=<..‘description’..>, @length=<..‘length’..>,

@timesdone=<..‘timestobedone’..>

GO

After execution a new record will be added to the Exercises table.

The following needs to be typed in a ‘New Query Editor Window’.

## To use sp\_UpdateExerciseTimesUsed

Variables required

@bookID which is the ID of the book

@times which is how many times the exercise must be done

What to type in order to execute the procedure

USE Suzi\_Yoga\_Studio  
GO

EXEC sp\_UpdateExerciseTimeUsed @bookID=<..‘bookID’..>, @timesdone=<..‘timestobedone’..>

GO

After execution the number of times that the exercise will need to be done will be updated where the bookID is the value that you input.

The following needs to be typed in a ‘New Query Editor Window’.

## To use sp\_DeleteBook

Variables required

@bookID which is the ID of the book

What to type in order to execute the procedure

USE Suzi\_Yoga\_Studio  
GO

EXEC sp\_DeleteBook @bookID=<..‘bookID’..>

GO

After execution the procedure will first check whether or not the record exists as part of the Exercises view, if it does not, the record where the bookID is of the value provided, else the record will not be deleted.

The following needs to be typed in a ‘New Query Editor Window’.

## To use sp\_Report

Variables required

@class\_code which is the code of the class you want check

What to type in order to execute the procedure

USE Suzi\_Yoga\_Studio  
GO

EXEC sp\_Report @class\_code = <..‘code\_of\_the\_class’..>

GO

After execution the procedure will print a report containing details about the class, including members and their cell numbers that are part of that class.

# Author Notes

Name : **Tejas**

Surname : **Dwarkaram**

Date : **2012/05/10**

# Project Notes

## Purpose

The purpose of this project is to create a database as a student project. The database is created under scenario of a Yoga Studio.

## Description

The database is for a Yoga Studio, named Suzi’s Yoga Studio. This database will house information about the members, classes and exercises of Suzi’s Yoga Studio. There are 4 tables that are utilized in this database. By making use of triggers, certain aspects of the tables are secured. Views have been created to cater for different criteria’s. Stored procedures have been designed to carry out certain tasks easier, like adding records to the “Exercises” table.

# ER Diagram

MEMBERS

#\* member\_Number  
 \* name  
 \* surname  
 \* date\_of\_birth  
 \* cell\_Number  
o email\_Address  
 \* studio\_number  
o sessions\_Per\_Week

CLASS\_SESSIONS

#\* class\_code  
 \* week  
 \* day  
 \* time\_Of\_Session  
 \* cancelled

EXERCISE\_BOOKS

#\* book\_ID  
 \* title  
 \* exercise\_Type

EXERCISES

\* name  
\* description  
\* length\_Of\_Exercises  
\* number\_Of\_Times\_To\_be\_Done