

PROJECT 3 Queens College Semester Course Database


NG1- Evnul Hossain , Hanlin
Wang, Haiim Salom
Lalehazarzadeh

Table of Contents

1	Group Meetings
2	Project Planner
3	Todo List
4	Nace Competencies
5	Stored Procedures
6	Queries



Group Meetings

- 5/5 - Discussed project specifications and how we should execute it
 - 5/7 - Assigned each team members jobs for this project
 - 5/15 - Helped any team members who is having trouble catching up
- 

Project Planner

Project Planner

Select a period to highlight at right. A legend describing the charting follows.

Period Highlight: 1

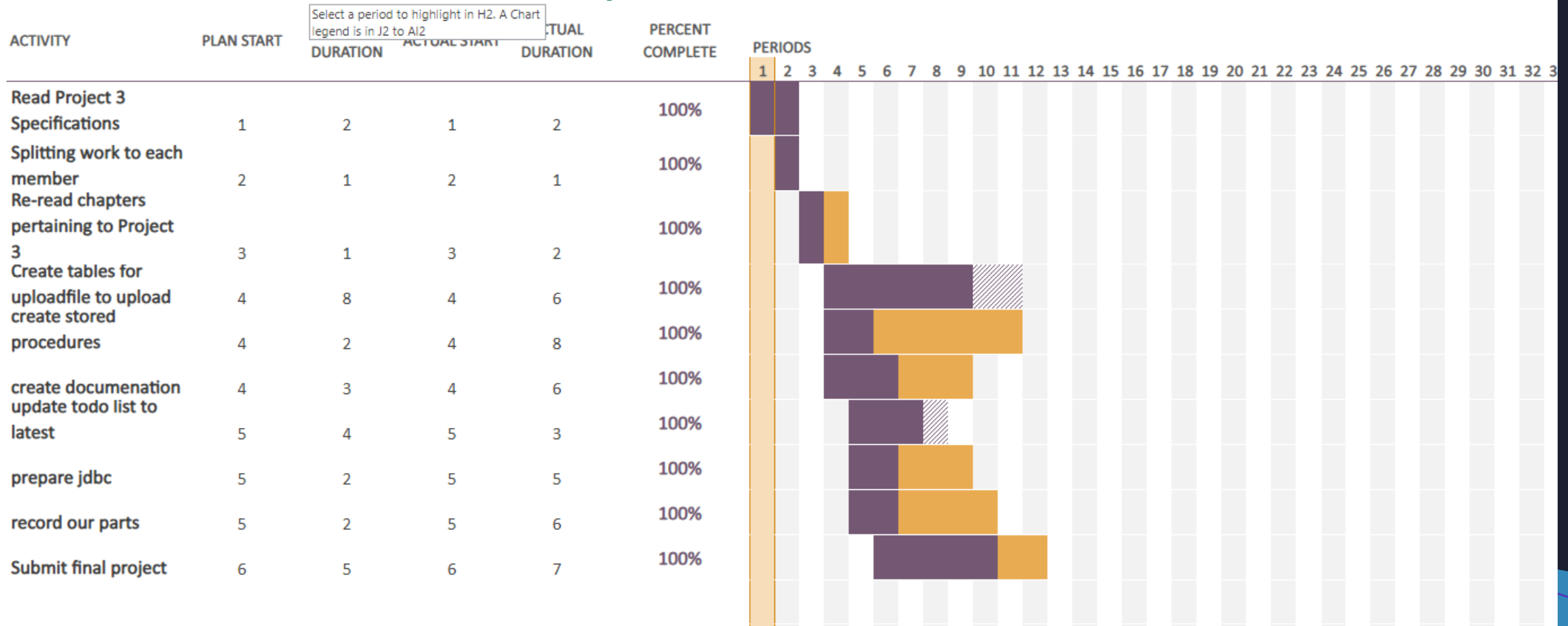
Plan Duration

Actual Start

% Complete

Actual (beyond plan)

% Complete (beyond plan)



Evnul Hossain To-Do List

To-do list

To be completed by:

Deadline: 15-May-24

Name: Evnul Hossain

Date: 05/5/2024

Project 1

% done	Phase	Start By	Original Due By	Revised Due By	Number Of Days	Revision Notes
100%	Group Meeting of Project 3 Requirements	5/4/2024	5/5/2024		1 day	Make sure everyone up to date
100%	Start final project by making tables	5/6/2024	5/7/2024		1day	Table looks like something you seen when registering for classes
100%	Create our stored procedures	5/6/2024	5/7/2024		1day	Should load data
100%	Finish and continue updating todo list	5/8/2024	5/10/2024		2day	
100%	Finish presentation	5/11/2024	5/12/2024		1day	make sure everyone presentation is good
100%	Create documentation with graphs and diagrams	5/12/2024	5/13/2024		1day	erd, sqldoc, etc
100%	jdbc recording	5/14/2024	5/15/2024		1day	
100%	Record my part presentation and wait for everyone else part	5/14/2024	5/15/2024		1day	
100%	SUBMIT FINAL PROJECT	5/15/2024	5/15/2024		1day	

NACE Competencies Leadership:

- Seek out resources to inform direction for teammates
- Plan, initiate due dates so teammates can finish their work
- Approach task with confidence

Hanlin Wang To-Do List

To-do list

To be completed by:

Deadline:

Name:Hanlin Wang

Date:5/1/2024

Project 1

% done	Phase	Start By	Original Due By	Revised Due By	Number Of Days	Revision Notes
100%	Restoring database	1-May	1-May	N/A	1	N/A
100%	Creating Schema	2-May	2-May	N/A	1	N/A
100%	Creating UDTs	3-May	3-May	N/A	1	N/A
100%	Creating Tables	4-May	4-May	N/A	1	N/A
100%	Creating Procedures	5-May	5-May	N/A	1	N/A
100%	LOADING DATA	6-May	6-May	N/A	1	N/A
100%	DEBUG	7-May	7-May	N/A	1	N/A
100%	DESIGNING PROPOSITIONS	8-May	8-May	N/A	1	N/A
100%	WRITING QUERIES	9-May	9-May	N/A	1	N/A
100%	RECORDING VIDEOS	10-May	10-May	N/A	1	N/A

NACE Competencies:

Critical Thinking/Problem Solving:

Analyze the need of this project, as well as the implementation.

Teamwork/Collaboration:

Having remote communications with group members via Discord.

Digital Technology:

A desktop with WIFI, also a from with internet services.

Haiim Salom Lalehazarzadeh To-Do List

- (insert image delete this text after) NACE Competencies

Evnu1 Hossain Stored Procedures / Workflow steps

```
USE [QueensClassSchedule]
GO
/***** Object:  StoredProcedure [Process].[usp_ShowWorkflowSteps]
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
-- =====
-- Author:      <Evnu1 Hossain>
-- Create date: <05/5/2024>
-- Description:  <To show each WorkFlowStep Process>
-- =====
ALTER PROCEDURE [Process].[usp_ShowWorkflowSteps]
AS
BEGIN
    -- For no additional results we do "SET NOCOUNT ON"
    SET NOCOUNT ON;
    SELECT *
    FROM [Process].[WorkFlowSteps];
END
```


Evnuul Hossain Stored Procedures / Load data from upload

```
ALTER PROCEDURE [dbo].[LoadDataFromUpload]
AS
BEGIN
    SET NOCOUNT ON;

    -- Load data into CourseManagement.Course table, avoiding duplicates
    INSERT INTO CourseManagement.Course (CourseCode, CourseName, Credits, InstructorId, UserAuthorizationKey, DateAdded, DateOfLastUpdate)
    SELECT DISTINCT
        [Code] AS CourseCode,
        [Description] AS CourseName,
        CASE
            WHEN CHARINDEX(',', [Course (hr, crd)]) > 0 AND CHARINDEX(')', [Course (hr, crd)]) > 0
            THEN TRY_CAST(SUBSTRING([Course (hr, crd)], CHARINDEX(',', [Course (hr, crd)]) + 1, CHARINDEX(')', [Course (hr, crd)]) - CHARINDEX(',', [Course (hr, crd)])) AS Credits
            ELSE 0 -- Set default value for Credits if extraction fails
        END AS Credits,
        HASHBYTES('MD5', [Instructor]) AS InstructorId, -- Assuming numeric InstructorId
        1 AS UserAuthorizationKey,
        GETDATE() AS DateAdded,
        GETDATE() AS DateOfLastUpdate
    FROM [QueensClassSchedule].[Uploadfile].[CurrentSemesterCourseOfferings]
    WHERE [Code] NOT IN (SELECT CourseCode FROM CourseManagement.Course);

    -- Load data into InstructorManagement.Instructor table with valid names
    INSERT INTO InstructorManagement.Instructor (LastName, FirstName, MaxCredits, UserAuthorizationKey, DateAdded, DateOfLastUpdate)
    SELECT DISTINCT
        PARSENAME([Instructor], 1) AS LastName,
        PARSENAME([Instructor], 2) AS FirstName,
        15 AS MaxCredits,
        1 AS UserAuthorizationKey,
        GETDATE() AS DateAdded,
        GETDATE() AS DateOfLastUpdate
    FROM [QueensClassSchedule].[Uploadfile].[CurrentSemesterCourseOfferings]
    WHERE [Instructor] IS NOT NULL
    AND [Instructor] != ''
    AND LEN(RTRIM(LTRIM([Instructor]))) > 0
    AND LEN(RTRIM(LTRIM(PARSENAME([Instructor], 2)))) > 0;

    -- Load data into LocationManagement.BuildingLocation table
    INSERT INTO LocationManagement.BuildingLocation (BuildingName, UserAuthorizationKey, DateAdded, DateOfLastUpdate)
    SELECT DISTINCT
        [Location] AS BuildingName,
        1 AS UserAuthorizationKey,
        GETDATE() AS DateAdded,
```

```
INSERT INTO DepartmentManagement.Department (DepartmentName, DepartmentCode, MaxEnrollment, UserAuthorizationKey, DateAdded, DateOfLastUpdate)
SELECT DISTINCT
    SUBSTRING([Description], 1, CHARINDEX(' - ', [Description]) - 1) AS DepartmentName,
    SUBSTRING([Description], CHARINDEX(' - ', [Description]) + 3, LEN([Description])) AS DepartmentCode,
    100 AS MaxEnrollment,
    1 AS UserAuthorizationKey,
    GETDATE() AS DateAdded,
    GETDATE() AS DateOfLastUpdate
FROM [QueensClassSchedule].[Uploadfile].[CurrentSemesterCourseOfferings]
WHERE [Description] LIKE '% - %' AND SUBSTRING([Description], 1, CHARINDEX(' - ', [Description]) - 1) NOT IN (SELECT DepartmentName FROM DepartmentManagement.Department);

-- Check and insert ModeOfInstruction for 'In-Person', 'Hybrid', and 'Online'
IF NOT EXISTS (SELECT 1 FROM InstructionManagement.ModeOfInstruction WHERE ModeName = 'In-Person')
BEGIN
    INSERT INTO InstructionManagement.ModeOfInstruction (ModeName, UserAuthorizationKey, DateAdded, DateOfLastUpdate)
    VALUES ('In-Person', 1, GETDATE(), GETDATE());
END

IF NOT EXISTS (SELECT 1 FROM InstructionManagement.ModeOfInstruction WHERE ModeName = 'Hybrid')
BEGIN
    INSERT INTO InstructionManagement.ModeOfInstruction (ModeName, UserAuthorizationKey, DateAdded, DateOfLastUpdate)
    VALUES ('Hybrid', 1, GETDATE(), GETDATE());
END

IF NOT EXISTS (SELECT 1 FROM InstructionManagement.ModeOfInstruction WHERE ModeName = 'Online')
BEGIN
    INSERT INTO InstructionManagement.ModeOfInstruction (ModeName, UserAuthorizationKey, DateAdded, DateOfLastUpdate)
    VALUES ('Online', 1, GETDATE(), GETDATE());
END

IF NOT EXISTS (SELECT 1 FROM InstructionManagement.ModeOfInstruction WHERE ModeName = 'Web-Enhanced')
BEGIN
    INSERT INTO InstructionManagement.ModeOfInstruction (ModeName, UserAuthorizationKey, DateAdded, DateOfLastUpdate)
    VALUES ('Web-Enhanced', 1, GETDATE(), GETDATE());
END

-- Load data into CourseManagement.Class table with valid CourseId and ModeId
INSERT INTO CourseManagement.Class (CourseId, Schedule, MaxEnrollment, ModeId, UserAuthorizationKey, DateAdded, DateOfLastUpdate)
SELECT DISTINCT
    cm.CourseId,
    c.[Time] AS Schedule,
    c.[Limit] AS MaxEnrollment,
    mi.ModeId, -- Insert ModeId from InstructionManagement.ModeOfInstruction
    1 AS UserAuthorizationKey,
    GETDATE() AS DateAdded,
    GETDATE() AS DateOfLastUpdate
FROM [QueensClassSchedule].[Uploadfile].[CurrentSemesterCourseOfferings] c
JOIN CourseManagement.Course cm ON cm.CourseCode = c.[Code]
JOIN InstructionManagement.ModeOfInstruction mi ON mi.ModeName = c.[Mode Of Instruction]
WHERE EXISTS (SELECT 1 FROM CourseManagement.Course WHERE CourseCode = c.[Code])
AND mi.ModeId IS NOT NULL;
```

Evnu! Hossain Stored Procedures / Track Workflow

```
USE [QueensClassSchedule]
GO
/***** Object:  StoredProcedure [Process].[usp_TrackWorkflow]    Script Date: 5/18/2024 12:04:48 AM *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
-- =====
-- Author:      <Evnu! Hossain>
-- Create date: <05/5/2024>
-- Description:  <To show track each WorkflowStep within the database when processing>
-- =====
-- Create the usp_TrackWorkflow stored procedure
ALTER PROCEDURE [Process].[usp_TrackWorkflow]
    @StartTime DATETIME2,
    @WorkflowDescription NVARCHAR(100),
    @WorkflowStepTableRowCount INT,
    @UserAuthorizationKey INT
AS
BEGIN
    SET NOCOUNT ON;

    -- Insert tracking data into the WorkflowSteps table
    INSERT INTO Process.WorkflowSteps (WorkflowStepKey, WorkflowStepDescription, WorkflowStepTableRowCount, StartingDateTime, UserAuthorizationKey)
    VALUES (
        (SELECT ISNULL(MAX(WorkflowStepKey), 0) + 1 FROM Process.WorkflowSteps),
        @WorkflowDescription,
        @WorkflowStepTableRowCount,
        @StartTime,
        @UserAuthorizationKey
    );

    PRINT 'Workflow step tracked successfully.';
END;
```

Evnul Hossain Query : Find c++ classes

```
-- Find all professors that teaches C++ including the Schedule
```

```
SELECT DISTINCT
    c.CourseName,
    I.LastName,
    I.FirstName,
    cl.Schedule
FROM
    InstructorManagement.Instructor AS I
JOIN
    CourseManagement.Course AS C ON I.InstructorID = C.InstructorID
Join CourseManagement.Class as cl on cl.CourseId = c.CourseId
WHERE
    C.CourseName LIKE '%C++%';
```

Evnuul Hossain Query : Find online classes

```
-- Find all classes where the course is online

SELECT DISTINCT
    c.ClassId,
    c.CourseId,
    cr.CourseName,
    c.Schedule,
    c.MaxEnrollment,
    mi.ModeName,
    i.FullName
FROM
    CourseManagement.Class AS c
JOIN
    CourseManagement.Course AS cr ON c.CourseId = cr.CourseId
JOIN
    InstructionManagement.ModeOfInstruction AS mi ON c.ModeId = mi.ModeId
Join InstructorManagement.Instructor as i on cr.InstructorId = i.InstructorId
WHERE
    c.ModeId = 3; --3 is the id for online classes if you look at the modeofinstruction table
```

Evnuul Hossain Query : Find building locations and state their full names

```
SELECT
    RL.RoomNumber,
    RL.BuildingId,
    CASE BL.BuildingName
        WHEN 'KG' THEN 'King Hall'
        WHEN 'KY' THEN 'Kiely Hall'
        WHEN 'PH' THEN 'Powermaker Hall'
        WHEN 'SU' THEN 'Student Union'
        WHEN 'SB' THEN 'Science Building'
        WHEN 'QH' THEN 'Queens Hall'
        WHEN 'HH' THEN 'Honor Hall'
        WHEN 'IB' THEN 'I Building'
        WHEN 'JH' THEN 'Jefferson Hall'
        WHEN 'GB' THEN 'G Building'
        WHEN 'RE' THEN 'Remsen Hall'
        WHEN 'RZ' THEN 'Razran Hall'
        WHEN 'RA' THEN 'Rathaus Hall'
        WHEN 'GC' THEN 'Gertz Center'
        WHEN 'FH' THEN 'Frese Hall'
        WHEN 'MU' THEN 'Music Hall'
        WHEN 'RO' THEN 'Rosenthal Library'
        WHEN 'FG' THEN 'FitzGerald Gym'
        WHEN 'KP' THEN 'Klapper Hall'
        WHEN 'GT' THEN 'Goldstein Theater'
        WHEN 'CH' THEN 'Colwin Hall'
        WHEN 'DY' THEN 'Delany Hall'
        ELSE BL.BuildingName
    END AS FullBuildingName
FROM
    LocationManagement.RoomLocation AS RL
JOIN
    LocationManagement.BuildingLocation AS BL ON RL.BuildingId = BL.BuildingId;
```

Evnul Hossain Query : Course time duration

```
SELECT DISTINCT
    c.CourseId,
    co.courseName as [Course Name],
    i.fullname as Instructor,
    -- Convert duration to hours and minutes
    CAST(DATEDIFF(MINUTE,
        CAST(REPLACE(SUBSTRING(c.Schedule, 1, CHARINDEX('-', c.Schedule) - 1), ' ', '' ) AS TIME),
        CAST(REPLACE(SUBSTRING(c.Schedule, CHARINDEX('-', c.Schedule) + 1, LEN(c.Schedule)), ' ', '' ) AS TIME)
    ) / 60 AS VARCHAR) + ' hours ' +
    CAST(DATEDIFF(MINUTE,
        CAST(REPLACE(SUBSTRING(c.Schedule, 1, CHARINDEX('-', c.Schedule) - 1), ' ', '' ) AS TIME),
        CAST(REPLACE(SUBSTRING(c.Schedule, CHARINDEX('-', c.Schedule) + 1, LEN(c.Schedule)), ' ', '' ) AS TIME)
    ) % 60 AS VARCHAR) + ' minutes' AS Duration

FROM [CourseManagement].[Class] as c
JOIN CourseManagement.Course AS co ON c.CourseId = co.CourseId
JOIN InstructorManagement.Instructor as i on i.InstructorId = co.InstructorId
```

Haiim Lalehazarzadeh Building Location

```
USE [QueensClassSchedule]
GO

/***** Object: Table [LocationManagement].[BuildingLocation]    Script Date: 5/18/2024 12:42:33 AM *****/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [LocationManagement].[BuildingLocation](
    [BuildingId] [int] IDENTITY(1,1) NOT NULL,
    [BuildingName] [nvarchar](100) NOT NULL,
    [UserAuthorizationKey] [Udt].[SurrogateKeyInt] NOT NULL,
    [DateAdded] [Udt].[DateAdded] NOT NULL,
    [DateOfLastUpdate] [Udt].[DateOfLastUpdate] NOT NULL,
    PRIMARY KEY CLUSTERED
    (
        [BuildingId] ASC
    )WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, ON
    ) ON [PRIMARY]
GO

ALTER TABLE [LocationManagement].[BuildingLocation] ADD DEFAULT (sysdatetime()) FOR [DateAdded]
GO

ALTER TABLE [LocationManagement].[BuildingLocation] ADD DEFAULT (sysdatetime()) FOR [DateOfLastUpdate]
GO
```



Haiim Lalehazarzadeh Department

```
USE [QueensClassSchedule]
GO

/***** Object: Table [DepartmentManagement].[Department]    Script Date: 5/18/2024 12:47:14 AM *****/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [DepartmentManagement].[Department](
    [DepartmentId] [int] IDENTITY(1,1) NOT NULL,
    [DepartmentName] [nvarchar](100) NOT NULL,
    [DepartmentCode] [nvarchar](20) NOT NULL,
    [MaxEnrollment] [int] NULL,
    [UserAuthorizationKey] [Udt].[SurrogateKeyInt] NOT NULL,
    [DateAdded] [Udt].[DateAdded] NOT NULL,
    [DateOfLastUpdate] [Udt].[DateOfLastUpdate] NOT NULL,
    PRIMARY KEY CLUSTERED
    (
        [DepartmentId] ASC
    )WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, ON
    UNIQUE NONCLUSTERED
    (
        [DepartmentCode] ASC
    )WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, ON
    ) ON [PRIMARY]
GO

ALTER TABLE [DepartmentManagement].[Department] ADD DEFAULT (sysdatetime()) FOR [DateAdded]
GO

ALTER TABLE [DepartmentManagement].[Department] ADD DEFAULT (sysdatetime()) FOR [DateOfLastUpdate]
GO

ALTER TABLE [DepartmentManagement].[Department] WITH CHECK ADD CHECK (([MaxEnrollment]>=(0)))
GO
```



Haiim Lalehazarzadeh Instructor

```

/***** Object:  Table [InstructorManagement].[Instructor]      Script Date: 5/18/2024 12:46:53 AM *****/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [InstructorManagement].[Instructor](
    [InstructorId] [int] IDENTITY(1,1) NOT NULL,
    [MaxCredits] [int] NULL,
    [UserAuthorizationKey] [Udt].[SurrogateKeyInt] NOT NULL,
    [DateAdded] [Udt].[DateAdded] NOT NULL,
    [DateOfLastUpdate] [Udt].[DateOfLastUpdate] NOT NULL,
    [FullName] [varchar](255) NULL,
    [FirstName] [varchar](255) NULL,
    [LastName] [varchar](255) NULL,
    PRIMARY KEY CLUSTERED
    (
        [InstructorId] 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

ALTER TABLE [InstructorManagement].[Instructor] ADD DEFAULT (sysdatetime()) FOR [DateAdded]
GO

ALTER TABLE [InstructorManagement].[Instructor] ADD DEFAULT (sysdatetime()) FOR [DateOfLastUpdate]
GO

ALTER TABLE [InstructorManagement].[Instructor] ADD DEFAULT (NULL) FOR [FullName]
GO

ALTER TABLE [InstructorManagement].[Instructor] ADD DEFAULT (NULL) FOR [FirstName]
GO

ALTER TABLE [InstructorManagement].[Instructor] ADD DEFAULT (NULL) FOR [LastName]
GO

ALTER TABLE [InstructorManagement].[Instructor] WITH CHECK ADD CHECK ((([MaxCredits]>=(0))))
GO

```

Haiim Lalehazarzadeh Room Location

```
QLQuery33.sql - Io...sSchedule (sa (86))  SQLQuery32.sql - Io...sSchedule (sa (53))  FindOnlineClass_Qu...sSchedule (sa (79))
USE [QueensClassSchedule]
GO

/***** Object: Table [LocationManagement].[RoomLocation]    Script Date: 5/18/2024 12:46:30 AM *****/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [LocationManagement].[RoomLocation](
    [RoomId] [int] IDENTITY(1,1) NOT NULL,
    [BuildingId] [int] NULL,
    [RoomNumber] [nvarchar](20) NOT NULL,
    [Capacity] [int] NULL,
    [UserAuthorizationKey] [Udt].[SurrogateKeyInt] NOT NULL,
    [DateAdded] [Udt].[DateAdded] NOT NULL,
    [DateOfLastUpdate] [Udt].[DateOfLastUpdate] NOT NULL,
    PRIMARY KEY CLUSTERED
    (
        [RoomId] ASC
    )WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS
    ) ON [PRIMARY]
GO

ALTER TABLE [LocationManagement].[RoomLocation] ADD DEFAULT (sysdatetime()) FOR [DateAdded]
GO

ALTER TABLE [LocationManagement].[RoomLocation] ADD DEFAULT (sysdatetime()) FOR [DateOfLastUpdate]
GO

ALTER TABLE [LocationManagement].[RoomLocation] WITH CHECK ADD CHECK ((([Capacity]>=(0))))
GO
```

