**CIS 631 - Database Management System Design**

**New Jersey Institute of Technology**

**Professor Vincent Oria**

**Course Registration System**

**Project Report**

**Yasser Farha**

**Rayon Myrie**

**Object and Purpose**

The main purpose for this project is to get familiar with structured storage, application, modification, and lastly maintenance of data. In this project, students will use DBMS, which is a system that aims to allow the definition, creation, querying, update, and administration of databases. Moreover, another purpose of this project is to understand the data modeling concepts that is applied in a real time scenario and to implement a fully functional database system that interacts with an end user interface.

**Problem Definition**

In this term project, we are asked to design a small database system, The Course Registration system for New Jersey Institute of Technology needs to be created with a back-end database and a front-end web interface. The selection of a database system for the back-end is optional. Also, the web server used can be chosen optionally from the various application. In addition, we need to write a number of application programs to access the database. The topic of the project is to design the database system that the New Jersey Tech., a university of excellence can use to manage students and courses.

**Summary of The System Requirements**

New Jersey Institute Technology is starting to move some of its applications into a database environment. We have been hired to set-up the database system to assist New Jersey Tech. achieve its goals. A requirements analysis that was conducted has identified a number of things about the operations and goals of New Jersey Tech.. We, as the systems analyst/designer, should feel free to add to these requirements in order to achieve a richer design. New Jersey Tech. keeps records about a number of items. These are the following:

1. First and foremost, it keeps track of its students who are uniquely identified by their Student Ids (different from the SSN). For each student, his/her SSN, address, high school, major (identified by a particular department) and year are other essential information.

2. Information about staff is also maintained. The name and SSN (which uniquely identify each staff) are essential information for each staff. The address, salary are other required information. 3. A particular type of staff is the faculty. In addition to the general staff information, the rank and course load (in terms of maximum number of courses that the particular faculty can teach) information are stored for each faculty.

4. A faculty may be assigned to multiple departments (joint appointments). This enables them to teach courses at multiple departments as long as the total number of courses is not higher than the course load.

5. Information about each department is maintained. Each department is identified by a unique codes. The department name, the location of the main office, latest annual budget 1 are the information that are stored. Each department has a chairperson who is a faculty member.

6. Physical room facilities of the university are identified. The identification is by means of a building code and room number. Other information include the audio-visual equipment and the capacity of the room.

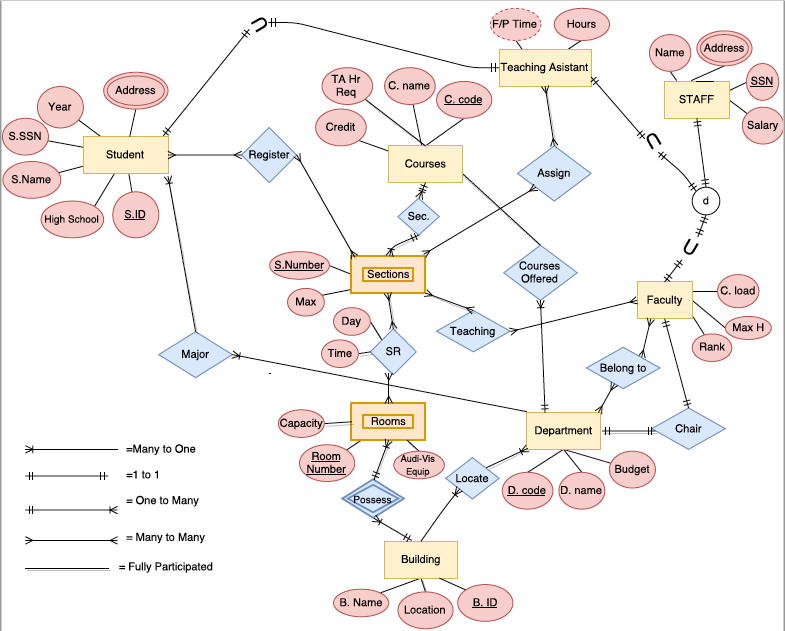
7. Departments offer a number of courses. Course information includes the course code, course name, course credit (using the nomenclature that is followed at NJIT). No two courses in the university can have the same code. For each course, the teaching assistant requirements (in terms of number of hours per week) are also stored.

8. Sections of courses are identified by a number. Note that there cannot exist a section unless it is related to an existing course. For the sections, New Jersey Tech. keeps information about the room, the weekdays (1 or 2) and the time the lectures are held and the maximum enrollment.

9. Each section is taught by one faculty member. Faculty members can teach multiple sections of a course or sections of different courses.

10. Students register for a number of course sections but to only one section of the same course. 11. For each course a number of teaching assistants (TA) can be assigned. Teaching assistants are students who have been hired as staff. Full-time TAs are allowed to work a maximum of 20 hours a week while half-time TAs can work 12 hours a week.

**Entity-Relation Diagram**



**Database Design Schema**

CREATE TABLE [dbo].[Register] (

[RId] INT IDENTITY (1, 1) NOT NULL,

[S\_ID] INT NOT NULL,

[S.Number] INT NOT NULL,

PRIMARY KEY CLUSTERED ([RId] ASC),

CONSTRAINT [UQ\_ID\_Reg] UNIQUE NONCLUSTERED ([S\_ID] ASC, [S.Number] ASC),

FOREIGN KEY ([S.Number]) REFERENCES [dbo].[Section] ([S.Number])

);

CREATE TABLE [dbo].[Rooms] (

[Room\_ID] VARCHAR (10) NOT NULL,

[Capacity] INT NOT NULL,

[Audio\_VIS\_Equip] VARCHAR (10) NOT NULL,

[B.ID] INT NOT NULL,

PRIMARY KEY CLUSTERED ([Room\_ID] ASC)

FOREIGN KEY ([B.ID]) REFERENCES Building ([B.ID])

);

CREATE TABLE [dbo].[Major] (

[M\_ID] INT IDENTITY (1, 1) NOT NULL,

[Major] NCHAR (20) NULL,

[D.Code] VARCHAR (6) NULL,

PRIMARY KEY CLUSTERED ([M\_ID] ASC)

);

CREATE TABLE [dbo].[Faculty] (

[Fac\_SSN] INT NOT NULL,

[C.load] INT NOT NULL,

[Max.H] INT NOT NULL,

[Rank] INT NOT NULL,

PRIMARY KEY CLUSTERED ([Fac\_SSN] ASC),

FOREIGN KEY ([Fac\_SSN]) REFERENCES [dbo].[STAFF] ([SSN])

);

CREATE TABLE [dbo].[Department ] (

[D.Code] VARCHAR (6) NOT NULL,

[D.Name] VARCHAR (50) NOT NULL,

PRIMARY KEY CLUSTERED ([D.Code] ASC)

);

CREATE TABLE [dbo].[Courses] (

[CR\_ID] VARCHAR (10) NOT NULL,

[C.Name] VARCHAR (40) NOT NULL,

[Credits] INT NOT NULL,

[TA\_hours] INT NOT NULL,

[D.code] VARCHAR (6) NOT NULL,

PRIMARY KEY CLUSTERED ([CR\_ID] ASC),

FOREIGN KEY ([D.code]) REFERENCES [dbo].[Department ] ([D.Code])

);

CREATE TABLE [dbo].[Building] (

[B.ID] INT NOT NULL,

[B.Name] NCHAR (20) NOT NULL,

[Location] NCHAR (20) NULL,

PRIMARY KEY CLUSTERED ([B.ID] ASC)

);

CREATE TABLE [dbo].[Section] (

[S.Number] INT NOT NULL,

[Max\_Cap] INT NOT NULL,

[CR\_ID] VARCHAR (10) NOT NULL,

[TIMEID] INT NOT NULL,

[Room\_ID] VARCHAR (10) NOT NULL,

[Fac\_SSN] INT NOT NULL,

PRIMARY KEY CLUSTERED ([S.Number] ASC),

FOREIGN KEY ([CR\_ID]) REFERENCES [dbo].[Courses] ([CR\_ID]),

FOREIGN KEY ([Fac\_SSN]) REFERENCES [dbo].[STAFF] ([SSN])

);

CREATE TABLE [dbo].[STAFF] (

[SSN] INT NOT NULL,

[FNAME] VARCHAR (20) NOT NULL,

[LNAME] VARCHAR (20) NOT NULL,

[STR1] VARCHAR (40) NOT NULL,

[CITY] VARCHAR (20) NOT NULL,

[STATE] VARCHAR (2) NOT NULL,

[ZIP] VARCHAR (7) NOT NULL,

[Salary] DECIMAL (18) NOT NULL,

[Staff\_Type] VARCHAR (10) NOT NULL,

[Passwd] TEXT NOT NULL,

PRIMARY KEY CLUSTERED ([SSN] ASC)

);

CREATE TABLE [dbo].[Student] (

[S\_ID] INT IDENTITY (1, 1) NOT NULL,

[FNAME] VARCHAR (20) NOT NULL,

[LNAME] VARCHAR (20) NOT NULL,

[S\_SSN] INT NOT NULL,

[STR1] VARCHAR (40) NOT NULL,

[CITY] VARCHAR (20) NOT NULL,

[ST\_CD] VARCHAR (2) NOT NULL,

[PSTL] VARCHAR (7) NOT NULL,

[H.School] VARCHAR (30) NOT NULL,

[Year] INT NOT NULL,

[Password] TEXT NOT NULL,

[M\_ID] INT NULL,

PRIMARY KEY CLUSTERED ([S\_ID] ASC),

FOREIGN KEY ([M\_ID]) REFERENCES [dbo].[Major] ([M\_ID])

);

CREATE TABLE [dbo].[Teaching\_Assistant] (

[TA\_SSN] INT NOT NULL,

[F\_p\_time] CHAR (1) NULL,

[W\_hours] INT NOT NULL,

PRIMARY KEY CLUSTERED ([TA\_SSN] ASC),

FOREIGN KEY ([TA\_SSN]) REFERENCES [dbo].[STAFF] ([SSN])

);

CREATE TABLE [dbo].[Time] (

[TIMEID] INT NOT NULL,

[Period] VARCHAR (40) NULL,

[Day] VARCHAR (20) NULL,

PRIMARY KEY CLUSTERED ([TIMEID] ASC)

);

Views Created:

CREATE VIEW courselist

AS SELECT s.CR\_ID AS [Course ID],

c.[C.Name] AS [Course Name],

s.[S.Number] AS Section,

t.Period AS [Time Period],

t.Day AS Days, { fn CONCAT(st.FNAME, st.LNAME) } AS Instructor,

s.Max\_Cap AS Capacity,

(SELECT COUNT(S\_ID) AS Expr1 FROM Register AS rg WHERE (s.[S.Number] = [S.Number])) AS [Students In class]

FROM Courses AS c RIGHT OUTER JOIN

Section AS s ON c.CR\_ID = s.CR\_ID LEFT OUTER JOIN

STAFF AS st ON s.Fac\_SSN = st.SSN FULL OUTER JOIN

Time AS t ON s.TIMEID = t.TIMEID

GROUP BY c.[C.Name], s.[S.Number], t.Period, t.Day, { fn CONCAT(st.FNAME, st.LNAME) }, s.Max\_Cap, s.CR\_ID

CREATE VIEW Student\_course\_count

AS SELECT S\_ID, COUNT([S.Number])as 'No of Classes' FROM Register

group by S\_ID

CREATE VIEW Student\_Courselist

AS SELECT s.CR\_ID AS [Course ID],

c.[C.Name] AS Course,

s.[S.Number] AS Section,

t.Period AS [Time Period],

t.Day, s.Max\_Cap AS Capacity,

{ fn CONCAT(st.FNAME, st.LNAME) } AS Instructor,

rg.S\_ID, rg.RId

FROM Time AS t INNER JOIN

Section AS s ON t.TIMEID = s.TIMEID INNER JOIN

STAFF AS st ON s.Fac\_SSN = st.SSN INNER JOIN

Register AS rg ON s.[S.Number] = rg.[S.Number] LEFT OUTER JOIN

Courses AS c ON s.CR\_ID = c.CR\_ID

Stored Procedures:

CREATE PROCEDURE sectionvalid

@Action Varchar (10),

@S\_ID int,

@S\_number int

as

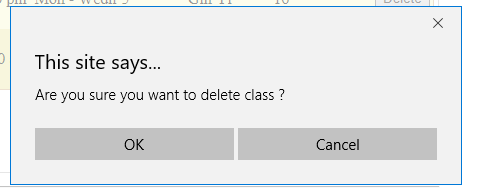
Begin

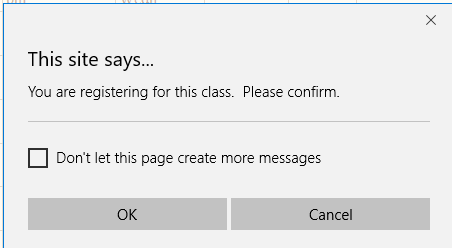
  SET NOCOUNT ON;

If @Action='Insert'

**Warning**

When registering students will receive 2 warnings, a delete confirmation and an Insert confirmation. These warning are shown in a pop up box. If the user presses cancel, the transaction will not get loaded to the database. If the user presses ok and insert or delete transaction will get loaded to the database.

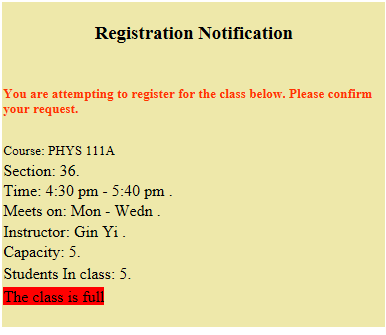




**Errors**

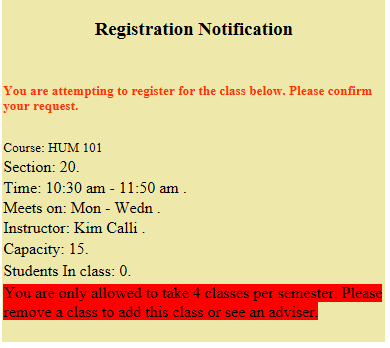
There are 3 types of errors a student can receive while trying to register for a class. These errors are shown in the student registration box, in a red message

1- Class if Full Error: This error occurs when a student tries to register for a class and the room has meet its full capacity.



2- Already Enrolled: If a student tries to enroll in a section or course that they are already enrolled in they will receive the following message.



3- Class Limit: Students are only allowed to register for 4 class a semester, if they have reached their enrollment limit and try to register for another class the will get the following message  
  


**Major Challenges:**

* The creation of the end user interface application (website), using C#
* Data insertion (taken from NJIT courses)
* Connecting the SQL database to xHTML interface as a grid to create views that the user can select
* Inserting rows selected from gridview back into the database. This took us 8 hours to figure out
* After creating assertions in the database it caused our front-end to crash, because the errors weren’t reported to the user. We have to find a way to notify the user with error warning before the went to the database, by creating user validation warnings and errors
* Learning curve: this was the first time we designed an application using Visual Studio, we had to learn how to create the application by doing research/trial and error. In addition, we were also learning about stored procedures and assertions in CS631, most of the functionality’s were complete, so we had to find a way to go back and implement the new items we learnt in class.

**Functionalities a student can perform in the system:**

Students can register in a course. Constraints here include:

* Maximum courses registered for a semester is 4
* Course previously not registered
* Courses capacity
* Time conflicts with currently registered courses

**Functionalities an Administrator can implement in the system:**

The administrator can do the following:

* Can view current schedule for a student and course
* Can edit seats left - to increase or decrease the capacity of the course

**Database Server & Web Interface Application**

This project was developed using ASP.net and C# and Microsoft SQL Server. We use Visual Studio to write and test the code. We chose to use Visual Studio because it had a built in web Server which runs locally on our machine and it made testing our web application easier. Using the Local IIS is the closest to what a live site would run under. It also has a one click deploy to Azure, which makes transferring our application into a life environment fast and easy. In addition, since Visual studio comes built in with Microsoft SQL Server it made it easy for us to install the program and start code immediately. We didn’t need to set up any server connections or install any other tools separately it was just one program. It this made testing easy because we both were in different locations and the application was saved on the cloud.

Design Software:

Microsoft Visual Studio Community 2017 Version 15.2 (26430.16) Release

Microsoft .NET Framework

Version 4.7.02046

Azure portal - ( we originally started to work and test product on azure. However we were working with a free licience and found defects with the connections, limits on the size of the database. At the last minute, we had to bring back the database to our local machine to present.)

Installed Version: Community

Visual C# 2017

Microsoft Visual C# 2017

Microsoft SQL Server

Version 13.00.1601

Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=" v5-rm\_FY\_st\v5-rm\_FY\_st\App\_Data\stbase1.mdf";Integrated Security=True;MultipleActiveResultSets=True

Hardware

|  |  |
| --- | --- |
| System Model | Latitude E5510 |
| Processor | Intel(R) Core(TM) i7 CPU M 620 @ 2.67GHz, 2667 Mhz, 2 Core(s), 4 Logical Processor(s) |
| Hard drive | 120GB |
|  |  |

**References**

Izik Ben-Gan, Dejan Sarka, Ron Talmage.- Querying Microsoft SQL Server 2012, ISBN:978-0-7356-6605-4

R. Elmasri & S. B. Navathe.- Fundamentals of Database Systems. Addison Wesley, 7th edition, 2015, ISBN:978-0133970777

Francis, Kauffman, Llibre, Sussman and Ullman. - Active Server Pages 2.0 ISBN:1-861001-34-7

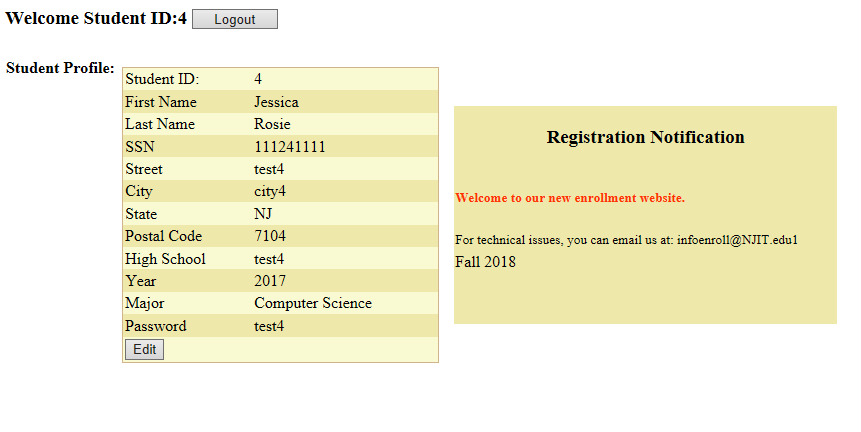
Judith S. Bowman, Sandra L. Emerson, Marcy Darnovsky The Practical SQL Handbook using Structured Query Language 3rd Edition ISBN:0-201-44787-8

Microsoft Developer Network: <https://msdn.microsoft.com/en-us/library/dd566231.aspx>

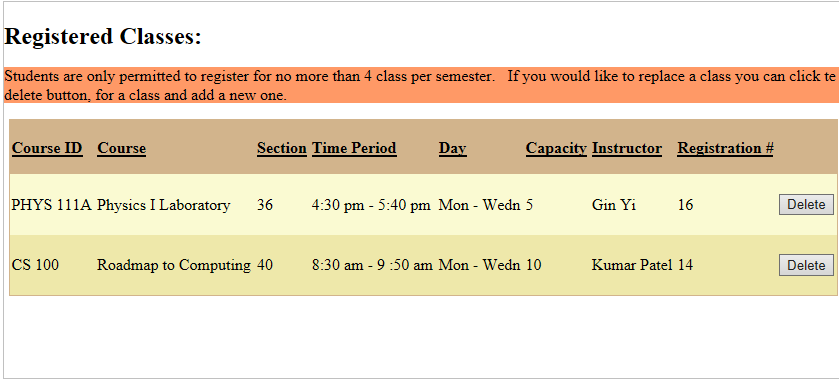
Code Project: <https://www.codeproject.com>

**Appendix**

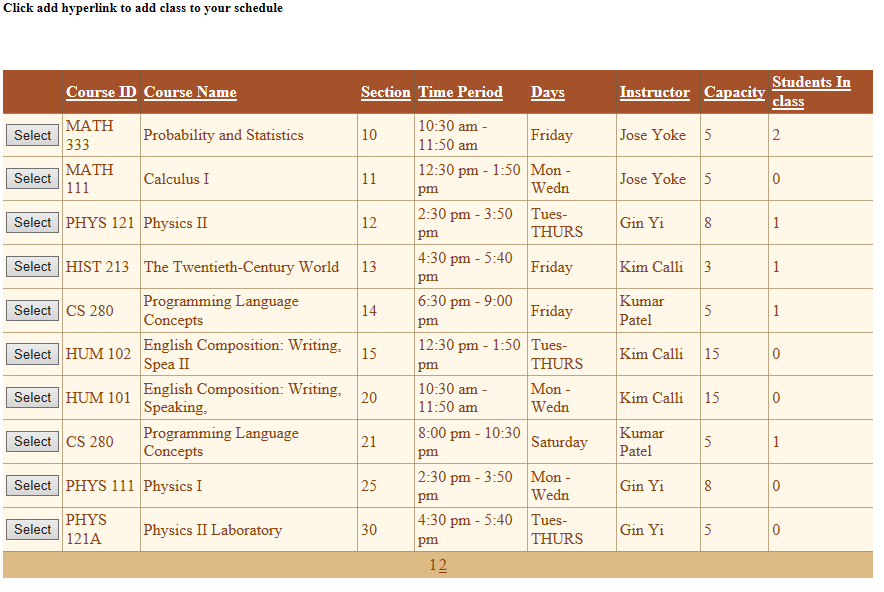
**Figure 1.1 Student Page**

****

**Figure 1.2 Registered Classes**

****

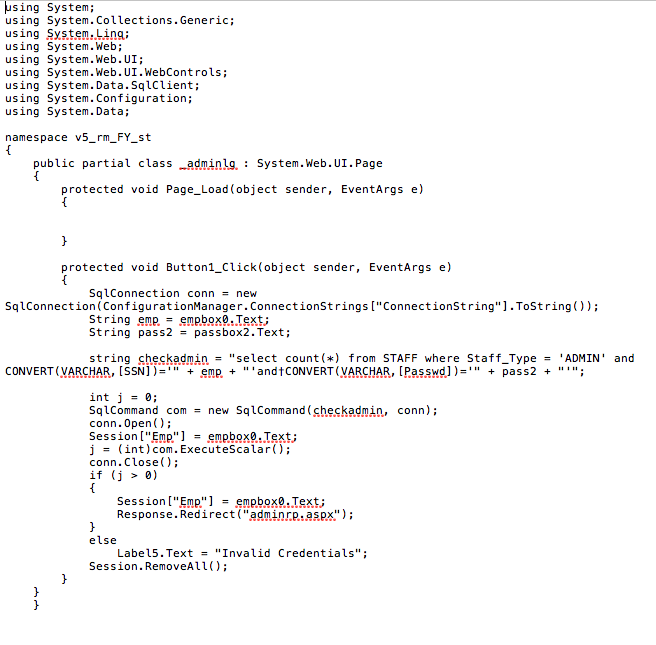
**Figure 1.3 Available Classes**

****

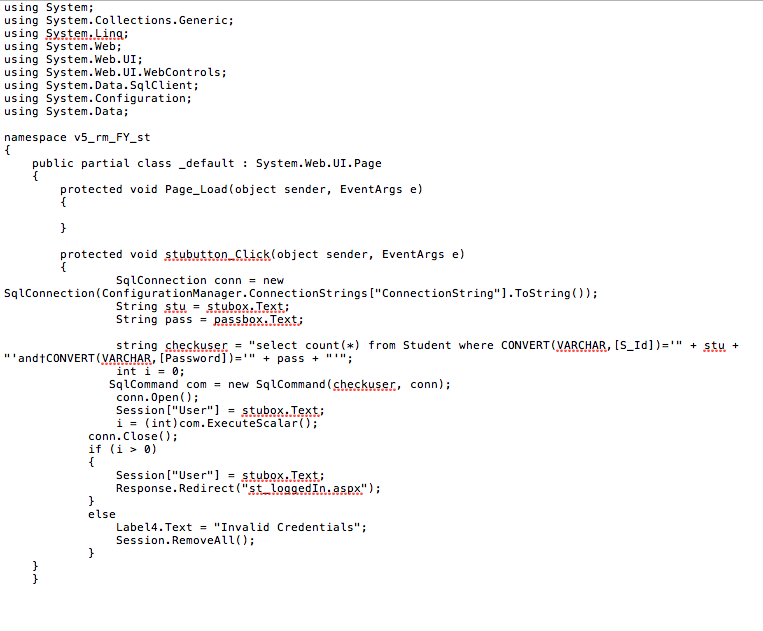
**Figure 1.4 Admin Page**

****

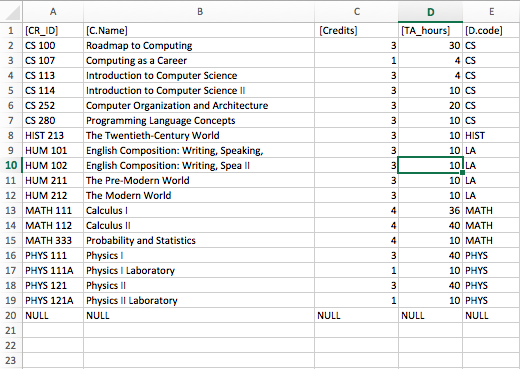
**Figure 1.5 Admin Log in Code**

****

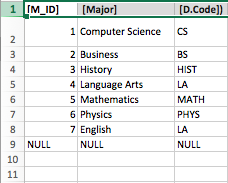
**Figure 1.6 Student Login Code**

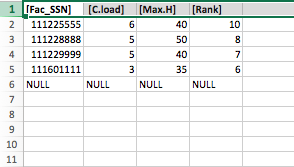
****

**Figure 1.7 Course Table**

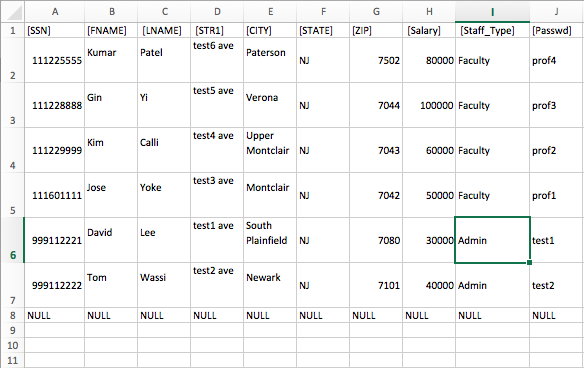
****

**Figure 1.8 Course Apprev Table**

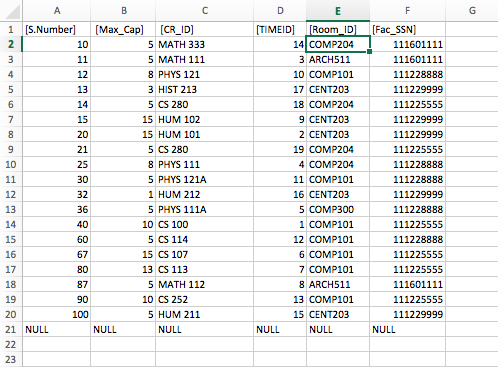
****

**Figure 1.9 Adminstrator Table  
 **

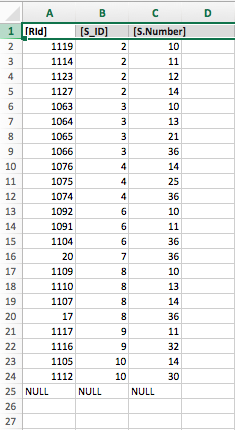
**Figure 2.1 Staff Table**

****

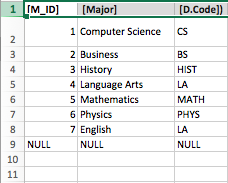
**Figure 2.2 Section Table**

****

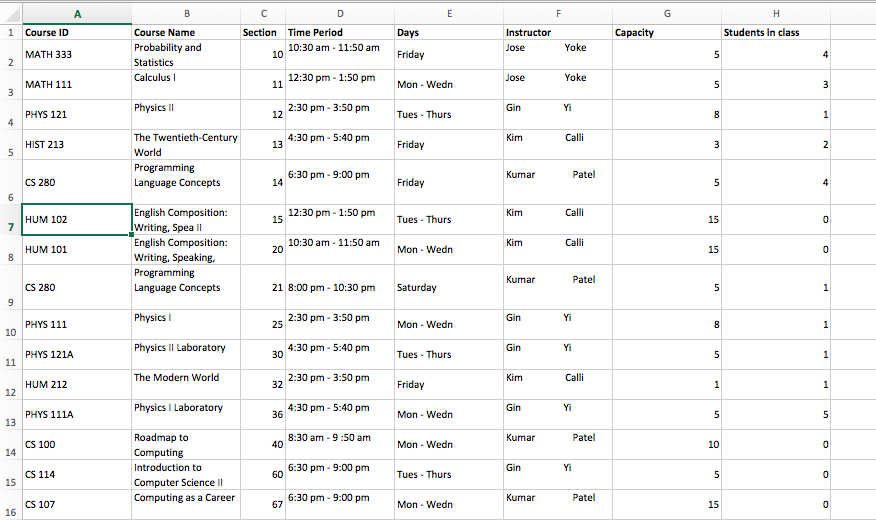
**Figure 2.3 Registrar**

****

**Figure 2.4 Courses Major**

****

**Figure 2.5 Course View**

****