**Diary Management System**

**Advanced Database Design**

**CS-603-B**

##### TechME



Sacred Heart University

School of Computer Science & Engineering The Jack Welch College of Business & Technology

Submitted To Dr. Reza Sadeghi

Spring 2022

CS-603-B Project Progress Report Phase #4 TechME

**TechME**

# Project Phase 04 for

# Diary Management System (DMS)

### Team Members

1. Kiera Cutri cutrik@mail.sacredheart.edu
2. Ankush Chaudhari chaudharia@mail.sacredheart.edu

### Roles of Team Members

1. Kiera Cutri (Team Head)
2. Ankush Chaudhari (Team Member)

2

CS-603-B Project Progress Report Phase #4 TechME

## Table of Contents

Team Member Introduction 4

Project Summary 5

Purpose

Scope

Motivation

Literature Summary

Pros and Cons

Entities and Keys 6

Entity-Relationship (ER) to Enhanced Entity-Relationship (EER) Model

SQL Code Desc.

Models and Figures 7

Entity-Relationship (ER) Model

Enhanced Entity-Relationship (EER) Model

Entities, Attributes, and Keys 8

SQL Code 9  
 Example of Insertion Error

Constraints Protect from Invalid Entry

Altering Our Code 10, 11

Citations 12

**Table of Figures**

Fig. 1 – Entity-Relationship (ER) Model 7  
Fig. 2 – Enhanced Entity-Relationship (EER) Model 7  
Fig. 3-12 – Alter Modifications 10

3

**Team Member Introduction**

**Kiera Mariah Cutri**

SHU ID Number: 0813395

SHU Email: [cutrik@mail.sacredheart.edu](mailto:cutrik@mail.sacredheart.edu)

Kiera Cutri is a current Computer Science graduate student at Sacred Heart University; they have previous experience with C++, C#, Java, and Python. Kiera has previously worked at the Sacred Heart Factory as a level one technician for 3 years where she gained experience with troubleshooting various software issues; she has also performed various repairs and upgrades to her laptop.

I met Ankush after the first Advanced Database course and we conversed about interests, backgrounds, and hobbies. After finding common ground for interests, I thought that he would be a good partner for the project, he reached out to me first regarding group work. The team leader was decided through a discussion of responsibilities and through deciding who would prefer to take the role.

**Ankush Chaudhari**

SHU ID Number: 0882634

SHU Email: [chaudharia@mail.sacredheart.edu](mailto:chaudharia@mail.sacredheart.edu)

I am Ankush Chaudhari. Currently, I am a graduate student at Sacred Heart University. I have completed my Undergrad in Electronics and Telecommunication Engineering. After I completed my degree, I worked for MNC as an implementation consultant for 1.6 years. My role was to handle ERP systems, Client relations Project Management, Vendor Management, provide clients hands on training on the system, and design and integrate the system according to client requirements. I am very passionate about my work role. my team had received an extra mile reward for completing the project before the deadline.

It was the first day of my Database Management class. I was seated in the first row. Kiera was seated towards my right. We had little conversation regarding the software installation and some doubts. After class we had a brief conversation regarding our backgrounds. Kiera completed her undergraduate in gaming, she also told me about her interest in various programming languages which she has known. On that point I decided that Kiera is a perfect team partner and head, so I approached her and now we are a team.

4

**Purpose**

The purpose of the Diary Management System (DMS) is to aid staff members within a department to collaborate and organize in a school setting. This system allows faculty members to update their activity details within a diary that is accessible to relevant users. Permitted users will have access to the activity of specific events that have been logged within the system. This proposed system aims to minimize the time burden imposed on faculty members by allowing users to search for a colleague's log within their diary. The Head of Department (HOD) will be able to view each faculty member's diary logs.

**Scope**

TechME aims to produce an application that will provide greater ease for users by allowing users to effortlessly search and find records that pertain to other colleagues. This system provides users with an advanced and secure way to store documents than can be accessed by higher ups within a department.

**Motivation**

These days, everything is stored within databases. Our staff continues to use more traditional logging methods when recording work information. To make the most out of current technology, we are creating a work diary management system which will allow faculty members to upgrade their work from traditional logging techniques to databases.

**Literature Survey**

Keeping logs to maintain work records daily allows faculty members to keep track of their work in their department. Each faculty member keeps and maintains a traditional diary which stores logs of their workday. Faculty members input various information relevant to their workday such as attendance, course number, semester taught, comments for the class, and content covered during lecture. The diary logs are later reviewed by the Department Head at the end of the day. Within our college, every faculty member keeps and maintains a work diary to enter logs of their workday daily. By maintaining a work diary, it allows faculty members to detail their workday and keep a log that would be available to other coworkers within their department. By having a diary system in place, it allows others to know the daily routines of faculty members and allows for easier assignment based on availability.

We were able to gauge an idea of how this system would work based on an online example of a diary management system on sourcecodester.com [2].

**Pros and Cons**

When considering the Diary Management System. We considered potential positives and negatives that would attribute to the system. The primary positives of the system would be that it would provide an accurate timeline of events and permit for easier scheduling of employees. The system would be able to support multiple team members at once and would allow coworkers to hold each other accountable for each other for specific actions preformed. The primary potential negatives we came across for this project would be a decrease in teamwork among members, large data size which use large pieces of software which would occupy multiple megabytes of diskspace, there could be an annual recurrent maintenance cost, and an ethical concern for privacy.

For more information regarding this project, please refer to the following github repository: <https://github.com/cutrik/CS603-FinalProject>

5

**Entities and Keys**

When selecting entities for our project, we examined how a university or college works and attempted to mimic how they operate within the database. We began by looking at the direct relationships among members of an institution; we noticed that there were individuals who worked for the institution which fell under faculty or staff members. We noticed that students interacted with both faculty and staff. Since faculty and staff members played a vital role in the Diary management System, we included them in our entity list.

We observed that the institution utilized multiple buildings and classrooms which members of the institution interacted with. When we looked to how faculty members use the building, we included aspects which were being used. Building name, classroom location, department, and the subject which the faculty member taught were all aspects of the institution which would allow for more detailed diary entries. Faculty members have information that pertain to themselves and to other aspects of the institution such as scheduling and teaching. Faculty members belong to different departments based on their specialized subjects which impacted what they would teach to students.

We noticed that the faculty members schedules aligned with what students would take based on what the faculty member would teach; we thought it important to include aspects of the student and faculty interaction in our database using student class status and student schedule. We noticed that students typically had declared one major but would still take a variety of subjects for their schedule. With each subject that a student would take, each class would occur at different times and different days for varied durations.

**ER to EER Model**

After considering what all our entities would be for the Diary Management System (DMS), we broke down what attributes would apart of each entity. We viewed each entity and determined which ones were strong and which ones would be weak; afterwards, we listed what type of relationship each of our entities would have with each other such as many to many, one to one, and one to many. After determining the relationships between entities, we transferred this into an ER Model. Once the ER Model was created, we transferred the information into an EER Model by adding datatypes to each attribute. We kept the names of entities and attributes across models.

**SQL Code**

When writing the code for the database we referred to w3schools.com [5] and guru99.com [4] for how to write and format primary keys and foreign keys. The image above links to a copy of the SQL code for the Diary Management System.

6

**Models and Figures**

**Diagram

Description automatically generated**

Figure 1: ER Model

Created in Lucidchart at [www.lucidchart.com](http://www.lucidchart.com) [3]

Timeline

Description automatically generated[1]

Figure 2: EER Model

7

**All Entities, Attributes, and Keys**

|  |  |
| --- | --- |
| **Faculty**  StaffID PK  Title  Status  Tenured  **Staff**  StaffID PK  StaffFirstName  StaffLastName  StaffPhoneNumber  Salary  DateHired  Position  **Departments**  DepartmentID PK  DeptName  DeptChair FK  **FacultyClasses**  StaffID CPK  ClassID CPK  SubjectID CPK  **Subjects**  SubjectID PK  CategoryID FK  SubjectCode  SubjectName  SubjectPreReq  SubjectDescription  **Classes**  ClassID PK  SubjectID FK  ClassRoomID FK  Credits  StartDate  StartTime  Duration  MondaySchedule  TuesdaySchedule  WednesdaySchedule  ThursdaySchedule  FridaySchedule  SaturdaySchedule | **Classrooms**  ClassRoomID PK  BuildingCode FK  PhoneAvailable  **Building**  BuildingCode PK  BuildingName  NumberOfFloors  ElevatorAccess  SiteParkingAvailable  **StudentSchedule**  ClassID CPK  StudentID CPK  ClassStatus FK  Grade  **Students**  StudentID PK  StudentFirstName  StudentLastName  StudentPhoneNumber  StudentGPA  StudentMajor  **SubjectsHaveStudents**  SubjectID  StudentID  MajorID  **StudentClassStatus**  CategoryID PK  CategoryDescription  **Majors**  MajorID PK  Major  **Categories**  CategoryID PK  CategoryDescription  DepartmentID |

8

**SQL Code**

When creating our database, we observed how most systems work in a school setting. When looking to things like if a faculty member were tenured, if a phone would be available, or if an elevator would be available in the building, we decided to use a single character to indicate if these things were available by having a yes (‘y’) value or a no (‘n’) value. Information such as titles, names, positions, and schedule we were unsure how many characters the user would input so we made sure to make these attributes variable characters. For attributes such as date hired, phone number, salary, and credits we knew that these values would be integer values and have allotted a value for how many digits would be input by the user; for example, phone numbers generally take the format of ‘(xxx) xxx-xxxx’ in the United States, so we have allowed for 10 integer values to be input. The grade point average (GPA) uses a float value based on how averages are calculated on a 4-point scale, at a school such as Sacred Heart University, GPA values appear in the format 0.000, based on the number of digits we have set the value to allow 4 digits. For attributes such as subject code, subject name, and other class identifying attributes, we will only be using integer values; unlike some school districts which use a mixture of characters and integers, we will identify by using certain hundreds or thousands of values to differentiate between courses, subjects…etc. When adjusting our database, we had adjusted datatypes to be more appropriate to the attribute; an example of one of these changes is adjusting date to be the datatype datetime instead of integer.

**Example of Insertion Error**

INSERT INTO subjectsHaveStudents (subjectID, studentID, majorID) VALUES ("ASTO" , "0835019", 11650)

Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails (`universitymanagement`.`subjectshavestudents`, CONSTRAINT `fksubjectID` FOREIGN KEY (`subjectID`) REFERENCES `subjects` (`subjectID`)) 0.016 sec

INSERT INTO subjectsHaveStudents (subjectID, studentID, majorID)

VALUES ("ASTO", "0835019", 11650);

**Constraints Protect from Invalid Entry**

Different constraints were used in our database to prevent invalid inputs; some such constraints were in the form of primary keys and foreign keys. Some examples that we had used in our database were through the datatype float, datetime, and time. Float was used for the attribute “studentGPA”, by having this value be a limited decimal value, it allows the data to be specified as a numeric value while if this value were a different datatype, it would open the ability for incorrect data. Another datatype that was used was time and datetime; these datatypes require input in a valid format and will reject the input if the formatting is incorrect, this decreases the input of incorrect data through mandatory formatting and only acceptance of numeric values. We had used the time and datetime through the “startDate” and “startTime” attributes. Additionally, we used multiple constraints through keys. Some such examples of keys we had used were primary and foreign keys. By using these keys, it assures that the data must remain consistent with the primary key to run without error; an example of a primary key we had used was “studentID”, we had referenced this through foreign keys in tables such as “subjectID”. As a foreign key, if the information about the datatype, the information input, or the length allotted did not match the primary key, the data would not store and would need to be edited.

9

**Altering Our Code**

When adjusting our code, we had to either add columns or adjust datatypes of attributes. We preformed these changes through a sequence of alter commands in our code. For the table “staff” we added the column for birth date and used an update command to input the data per identifying number for staff members in the database. We additionally preformed multiple modify commands to change previous datatypes to more efficient and relevant ones. The following are some code excepts for where these changes were made:

1. ALTER TABLE staff

ADD dateOfBirth datetime(6);

UPDATE staff

SET dateOfBirth = CAST('1989/01/22' AS DATETIME)

WHERE staffID = 0216659;

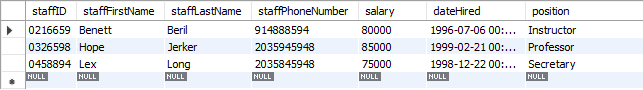


Figure 3: Before Alter staff

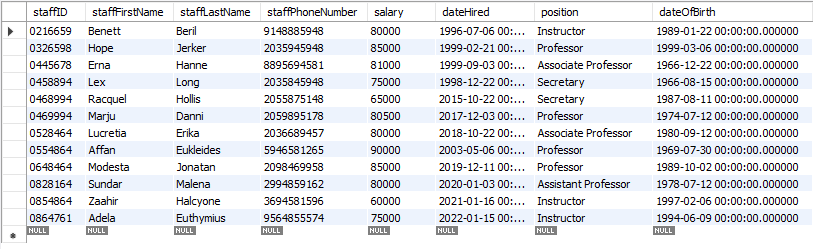


Figure 4: After Alter Staff

1. ALTER TABLE falcultyClasses

ADD subjectID varchar(8);



Figure 5: Before Alter falcultyClasses

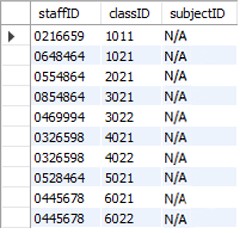


Figure 6: After Alter falcultyClasses

10

1. ALTER TABLE categories

MODIFY subjectID char(8);



Figure 7: Before Alter categories

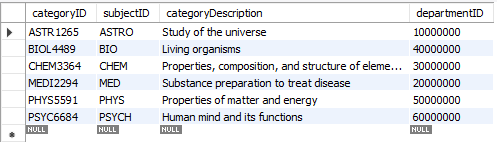


Figure 8: After Alter categories

1. ALTER TABLE building

MODIFY buildingCode varchar(6);

UPDATE building

SET buildingCode = CAST('1945-A' as varchar)

WHERE buildingName = 'Astronomy Building';

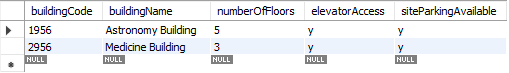


Figure 9: Before Alter building

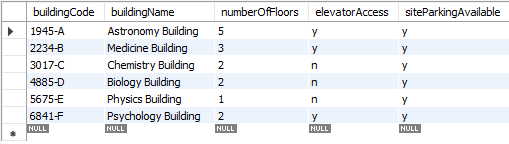


Figure 10: After Alter building

1. ALTER TABLE studentSchedule

MODIFY studentID char(8);



Figure 11: Before Alter studentSchedule

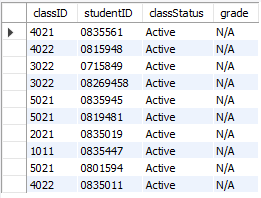


Figure 12: After Alter studentSchedule

11

**Citations**

1. Viescas, John L. “School Scheduling Modify Database Page 883.” *SQL Queries for Mere Mortals: A Hands-on Guide to Data Manipulation in SQL*, Pearson Technology Group Canada, 2018.
2. elinet. “Elinet.” *Free Source Code, Projects & Tutorials*, 11 Sept. 2014, https://www.sourcecodester.com/php/7943/online-diary-management-system.html.
3. W, Megan. “Urgent! -- Cite LudicChart – Lucidchart.” *Lucidchart*, 20 Sept. 2018, https://lucidchart.zendesk.com/hc/en-us/community/posts/360018227346-URGENT-Cite-LudicChart.
4. Peterson, Richard. “SQL Primary Key: How to Create & Add to Existing Table.” *Guru99*, 29 Jan. 2022, https://www.guru99.com/sql-server-primary-key.html.
5. “SQL FOREIGN KEY Constraint.” *SQL Foreign Key Constraint*, https://www.w3schools.com/sql/sql\_foreignkey.asp.

12