**Description of Conceptual and Logical Database Design**

Based on the feedback, create response entity to make instructors able to response users’ comment. Besides we also rearranged the comments and response entity. The description of all entity set and relationship set are listed below.

# Relational schema：

## Entity relationship:

Create Table USER (

    User\_ID INT,

    User\_Name Varchar(225)

    User\_Gender Varchar(30),

    User\_Type Varchar(225),

    PRIMARY KEY (User\_ID)

);

USER*: A general account type for all users. (Without the authorization of Prof.)*

*User\_ID: Primary Key for the USER table, uniquely define each individual user.*

*User\_Name: Name of each user.*

*User\_Gender: User could choose their gender, as well as Unknown to protect their Info.*

*User\_Type: Use to distinguish from Student and Prof.*

Create Table INSTRUCTOR (

    Inst\_ID INT,

    Inst\_Name VARCHAR(225),

    Inst\_Department VARCHAR(225),

    PRIMARY KEY (Inst\_ID)

);

INSTRUCTOR: A specific/special account type for those who are authorized as being a Prof.

*Inst\_ID: Primary key for Instructor Type account, uniquely defines each Prof kind's user.*

Inst\_Name: Name of each Prof.

Inst\_Department: The college the Prof is currently in.

Create Table COURSE (

    Course\_ID INT,

    Course\_Name INT,

    Course\_Department VARCHAR(225),

    PRIMARY KEY (Course\_ID)

);

COURSE: Contains all essential info of each specific course.

Course\_ID: Primary key for Course. Uniquely define each courses.

Course\_Department: Implies the college the course being in.

Course\_Name: Contains the course name.

Create Table COMMENT (

    Comment\_ID INT,

    Comment\_Score Short(5),

    Comment\_Text VARCHAR(1024),

    Course\_ID INT,

    PRIMARY KEY (Course\_ID),

    FOREIGN KEY(Course\_ID) REFERENCES Course(Course\_ID)

);

COMMENT: Each User could post a comment on the website.

Comment\_ID: Primary Key for COMMENT. Used to identify each comment.

Comment\_Score: For User to rate the corresponding course. Rating level is 1 ~ 5.

Comment\_Text: Posted messages for other Users or Prof to extract useful info about the corresponding course.

Course\_ID: Foerign Key referencing the Course\_ID inside COURSE table.

Create Table RESPONSE(

Response\_ID INT,

Response\_Text VARCHAR(1024),

Comment\_ID INT,

PRIMARY KEY(Response\_ID),

{FOERIGN KEY (Comment\_ID) REFERENCES COMMENT(Comment\_ID)}

);

RESPONSE: Responding function is only opened for Prof. It is aimed to be exhibited to other Users who want to extract useful info from this website.

Response\_Text: Prof could choose a random comment to reply.

Response\_ID: Primary Key for uniquely identify each Response making by Each Prof.

Comment\_ID: Foreign Key referencing to the Comment table. Identity which comment it belongs to.

Create Table GROUP(

Group\_ID INT,

Group\_Name VARCHAR(225),

Group\_founder INT,

PRIMARY Key (Group\_ID),

FOREIGN Key (Group\_founder) References USER(User\_ID)

 );

GROUP: Like a discord team. Students could create a group and chat.

Group\_ID: Primary key for identifying each group.

Group\_Name: Student could name a group as well as change them.

Group\_founder: Foreign Key referencing User\_ID inside User Table. Keep tracking of the User\_ID of the Founder.

Create Table Member(

Member\_ID INT,

User\_ID INT,

Group\_ID INT,

PRIMARY KEY (Member\_ID),

FOREIGN KEY (User\_ID) REFERENCES USER(User\_ID),

FOREIGN KEY (Group\_ID) REFERENCES GROUP(Group\_ID)

ON DELETE CASCADE

);

Member: A completely new ID created for User who is currently in a Group.

User\_ID: A Foreign Key referencing User\_ID in USER table.

Group\_ID: A Foreign Key referencing Group\_ID in GROUP table.

A user could get several Member\_ID which indicates they are in different groups. And a group can have several Member\_ID which indicates it contains several different users.

Create Table GROUP\_CHAT(

Chat\_ID INT,

Message\_Info VARCHAR(1024),

Send\_To\_ID INT,

PRIMARY KEY (Chat\_ID),

FOREIGN KEY (Send\_To\_ID) REFERENCES USER(User\_ID)

/\*ON UPDATE CASCADE \*/

);

Chat\_ID: A primary key used to uniquely identify each single text.

Message\_Info: Message contained in each single text.

Send\_To\_ID: Foerign Key referencing to User\_ID of User table. Indicate the user the text is sending to.

Create Table Response (

Comment\_ID INT,

Instructor\_ID INT,

Response\_Text VARCHAR(1024),

PRIMARY KEY Comment\_ID,

FOREIGN KEY (Comment\_ID) REFERENCES COMMENT (Comment\_ID)

ON DELETE CASCADE

);

RESPONSE: Responding function is only opened for Prof. It is aimed to be exhibited to other Users who want to extract useful info from this website.

Response\_Text: Prof could choose a random comment to reply.

Response\_ID: Primary Key for uniquely identify each Response making by Each Prof.

Comment\_ID: Foreign Key referencing to the Comment table. Identity which comment it belongs to.

## Relationship:

Create Table Target\_To(

Comment\_ID INT,

Course\_ID INT,

PRIMARY KEY (Comment\_ID),

FOREIGN KEY (Comment\_ID) REFERENCES COMMENT (Comment\_ID),

FOREIGN KEY (Course\_ID) REFERENCES COMMENT (Course\_ID)

);

Target\_To ：We want to make every comment into one course. Each course can have many comments targeted to it.

Comment\_ID：Indicate which comment we want to assign.

Course\_ID INT : The course that comments belong to. Every course can have multiple comments.

CREATE Table Create\_Group(

Founder\_ID INT,

Group\_ID INT,

PRIMARY KEY (Group\_ID),

FOREIGN KEY (Group\_ID) REFERENCES GROUP(Group\_ID)

FOREIGN KEY (Founder\_ID) REFERENCES GROUP (User\_ID)

);

Create\_Group : Founders can create many groups.

Founder\_ID : The one that creates the group, founders comes from the user.

Group\_ID : Every group has a group id, when the group is created, we give it an unique groupiD

Create Table Join\_As(

User\_ID INT,

Member\_ID INT,

PRIMARY KEY (User\_ID),

PRIMARY KEY (Member\_ID),

FOREIGN KEY (User\_ID) REFERENCES USER(User\_ID),

FOREIGN KEY (Member\_ID) REFERENCES MEMBER(Member\_ID)

ON DELETE CASCADE

);

Join\_As: The userID and memberID will together secure one join as relation

User\_ID : The user who want to join as a group member

Member\_ID : When a user joins as a group member, we give him a unique user\_ID, one person registered as many members as he can.

Create Table Contains(

Member\_ID INT,

Group\_ID INT,

PRIMARY KEY (Member\_ID),

FOREIGN KEY (Member\_ID) REFERENCES MEMBER(Member\_ID),

FOREIGN KEY (Group\_ID) REFERENCES MEMBER(Group\_ID)

ON DELETE CASCADE

);

Contains: Groups can contain many users and record their Member\_ID.

Member\_ID: Member\_ID is a primary key for Contains, and it will uniquely identify each member   in the Group. Member\_ID is a foreign key to the MEMBER Table. It will cascade the deletion occurred in MEMBER Table.

Create Table Belongs\_To(

Chat\_ID INT,

Group\_ID INT,

PRIMARY KEY (Chat\_ID),

FOREIGN KEY (Chat\_ID) REFERENCES GROUP\_CHAT(Chat\_ID),

FOREIGN KEY (Group\_ID) REFERENCES GROUP\_CHAT(Group\_ID)

ON DELETE CASCADE

);

Belongs\_To: Will contain the relation information between Chat and Group. Indicates each chats belonging to its group. A group can contain any chats.

Chat\_ID: Being a primary key and foreign key at the same time. Use to Identify each chat inside the Group.

Group\_ID: A foreign key used to indicate which group the chats are belonging to.

Create Table Sent\_To(

Member\_ID INT,

Chat\_ID  INT,

PRIMARY KEY (Member\_ID),

PRIMARY KEY (Chat\_ID),

FOREIGN KEY (Member\_ID) REFERENCES MEMBER(Member\_ID),

FOREIGN KEY (Chat\_ID) REFERENCES GROUP\_CHAT(Chat\_ID)

ON DELETE CASCADE

);

Sent\_To: Contains the Information of the sending chat. It is a many-to-many relation.

Chat\_ID And Member\_ID: Together form the primary key and both being used as foreign key in Send\_to Relation. Used uniquely to identify each chat, and the people sending to.

Create Table post\_response (

Inst\_ID INT,

Response\_ID INT,

PRIMARY KEY Inst\_ID, Response\_ID

FOREIGN KEY (Inst\_ID, Response\_ID) REFERENCES COMMENT (Comment\_ID, User\_ID)

ON DELETE CASCADE

);

Comments: Responding function is only opened for Prof. It is aimed to be exhibited to other Users who want to extract useful info from this website.

User\_ID : The user who want to join as a group member

Create Table Comments (

Comment\_ID INT,

User\_ID INT,

PRIMARY KEY Comment\_ID, User\_ID

FOREIGN KEY (Comment\_ID, User\_ID) REFERENCES COMMENT (Comment\_ID, User\_ID)

ON DELETE CASCADE

);

Comments: Responding function is only opened for Prof. It is aimed to be exhibited to other Users who want to extract useful info from this website.

User\_ID : The user who want to join as a group member

Create Table Teach (

Course\_ID INT,

Instructor\_ID INT,

PRIMARY KEY COURSE\_ID, Instructor\_ID

FOREIGN KEY (Course\_ID, Instructor\_ID) REFERENCES COMMENT (Course\_ID, Instructor\_ID)

ON DELETE CASCADE

);

*Inst\_ID: Primary key for Instructor Type account, uniquely defines each Prof kind's user.*

Course\_ID INT : The course that comments belong to. Every course can have multiple comments.