**Introduction**

This document contains description of our entities and relationships. The schemas are in 3rd normal form. We are creating a “Social Networking Platform” for the university. We started the project by identifying the entities who would use our system. Then, we started identifying the relationships between those entities. Later, we discussed the multiplicity in the relationships such as one-to-one, one-to-many or many-to-many. We introduced subclass relationship to hold entity specific attributes and the parent class held common attributes. We also had discussion if we had a weak entity. Once the entities and relationships were finalized, we started creating tables in our database with all constraints. Procedures and triggers were used to handle some scenarios. Once the database was prepared, we started working on the web interface. We created HTML forms for client-end and PHP code for server-end to handle the submitted data. JQuery was also used in few use-cases. Once the functionality was built and tested, styling was completed with a CSS file. We named our application as: “IIT Connect”. Please read further for more specific details.

**Description for Entities**

1. **User**

This is the parent entity for two subclasses. It holds the common attributes like name, email, username and password, along with unique userId .

1. **Student**

Student is a subclass of user. Each user has a set of attributes that can be changed and the visibility can be modified. Student can mention the year and semester when he/she joined. The degree name and current status such as “pursuing” or “completed” can be updated. Students can choose to make their GPA private.

1. **TA** (Teaching Assistant)

TA is a Student. A TA inherits all the attributes of its superclass: ‘Student’. A TA helps to moderate the Interest Group along with faculty.

1. **Faculty**

Faculty is a user. Faculty teaches courses and moderates the default interest group for a course with help from TA. Faculty can specify year of joining and current position. List of research projects and past experience can also be listed in profile.

1. **Course**

Courses are taught by faculty and students enroll in courses. Courses have a default interest group. The attributes of a course are courseId and title.

1. **Interest Group/Club**

The type attribute can be either ‘Interest Group’ or ‘Club’. Users (Student or Faculty) can request to join an Interest Group/Club. Upon request, the ‘status’ attribute can hold either ‘pending’ or ‘approved’. Since we need a separate set of username and password for Interest Group/Club login, we have two attributes: username and password to assign to the user upon approval. Each Interest Group/Club can be identified by a unique ID (groupClubId).

1. **Course Interest Group**

Each course will have a default Interest Group. groupId and name describe this entity. Faculty and TA are the default moderators for the default Interest Group. They can approve membership and filter content. They can also make announcement for all students in their course.

1. **Discussion Forum**

Discussion forums can be created by members of course interest group and interest group club. Each discussion forum will have a forum name and forum id. The name of member who created the forum will also be stored.

1. **Alert**

The alert table can store time of an alert and the message describing the alert condition. The moderator who should see the alert is also stored in the same tuple.

1. **Topic**

A course interest group or an interest group/club can have discussion forums. Each discussion forum can have discussions on multiple topics. In the ‘Topic’ table we are storing topic ID, name of topic and forum ID. We retrieve topics related to a particular discussion forum using this table.

1. **Comments**

In a discussion forum, a member can make comments on various topics. To capture this information, we have Comments relation. We store information about who commented and in which topic did the person comment. The complete comment is identified by an auto-generated comment ID. The comments table also has a date and time stamp.

1. **Criteria**

The criteria table helps us to store points against various actions by a user. The actions can be ‘created’, ‘joined’ and ‘commented’. User can create or join an interest group/club or course interest group. User can comment on various topics available in discussion forums.

**Description for Relationships**

1. **Enrolled**

Enrolled is a many-many relationship between Student and Course entity sets. A student can enroll in many courses and a course can have many students enrolled in it. The userId attribute is the student id and courseId attribute is the course id. userId and courseId attributes together uniquely identifies the tuple of this relation.

1. **Assists**

Assists is a many-many relationship between TA which is a subclass of student entity set and course. A TA can assist in many courses and a course can have many TAs. The userId attribute is the user id of TA and courseId is the course id in which the TA assists. userId and courseId together uniquely identifies the tuple of this relationship.

1. **ModeratesCIG**

Moderators can be assigned to Course Interest Groups. They can approve or deny requests and filter content on discussion forums. There are two attributes stored in this table, moderator ID and group ID.

1. **ModeratesIGC**

Similar to above description, Interest Group or Club have moderators with certain privileges. Moderator ID and group ID are stored in this table.

1. **Assigns**

The entity type Moderator/Administrator appears more than once in this relationship. It has two roles, site wide administrator and moderator. A site wide administrator can assign many moderators and each moderator must be assigned by one site wide administrator. The attribute moderator\_uid is the user id of moderator and administrator\_uid id the user id of the administrator.

1. **Approves**

A moderator can approve many users to join InterestGroup/clubs and each user must be approved by one moderator in order to join the group or club. The attribute userId is the user id, it can be either faculty or student, and groupClubId is the group/club id. groupClubId is the primary key of this relation.

1. **Teaches**

Teaches is a multiway relationship between Faculty, Course and CourseInterestGroup entity sets. We have assumed that a faculty member can teach more than one course and a course can be taught by more than faculty member. Teaches has an attribute avgGPA which is the functionality of both faculty and course. Other attributes of this relation are userId user id of faculty, courseId course id of course and groupID group id of course interest group.

1. **Course\_has**

Course\_has is a one-one relationship between course and interest group entity sets. A course can have one interest group and interest group belongs to one course. C\_has has courseId and groupId as attributes which uniquely determines the tuples. The courseId attribute uniquely determines the course and groupId attribute determines the interest group.

1. **Joins**

Joins is a many-many relationship between users and Interestgroup/Clubs. Many users can join many interest groups and an interest group can have many users. Joins has userId and groupClubId which uniquely determines the tuples of the relation. The userId uniquely determines the user and groupClubId determines the Interestgroup/Clubs.

1. **Creates**

Creates is a multi-way relationship between user, interestgroup/club and discussion forum. A user can create an interestgroup/club and the interestgroup/clubs can be created by a user. A user can create many discussion forum and a discussion forum can be created by one user. The userId, groupClubId and discussionId uniquely determines the tuples of the relation.

1. **CourseInterestGroup\_has**

Course Interest Groups have groups. Each group can have multiple forums. This table is used to capture the information regarding the different forums which a group has.

1. **InterestGroupClub\_has**

Similar to above description, Interest Group or Club has groups, which have multiple forums. This information is captured in this table.

1. **joinsCIG**

This table captures information about students joining course interest groups.

**Schemas**

1. **User** (userId, email, name, username, password)

*Primary Key: userId*

*Functional Dependency: userId -> email, name, username, password*

1. **Student** (userId, year, semester, GPA, degreeStatus, degreeType, jobs, bonus Credit)

*Primary Key: userId*

*Functional Dependency: userId -> year, semester, GPA, degreeStatus, degreeType, job, bonusCredit*

1. **TA** (userId)

*Primary Key: userId*

1. **Faculty** (userId, year, position, experience, projects)

*Primary Key: userId*

*Functional Dependency: userId -> year, position, experience, projects*

1. **Course** (courseId, title)

*Primary Key:* course*Id*

*Functional Dependency:* course*Id -> title*

1. **InterestGroup/Club** (groupClubId, name, type, description, status, username, password)

*Primary Key:* groupClub*Id*

*Functional Dependency:* groupClub*Id -> name, type, description, status, username, password*

1. **CourseInterestGroup** (groupId, name)

*Primary Key:* group*Id*

*Functional Dependency:* group*Id -> name*

1. **DiscussionForum** (discussionID, comments)

*Primary Key: {*discussion*ID, comments}*

1. **Enrolled** (userId, courseId)

*Primary Key: {*userId*,* courseId}

1. **Assists** (userId, courseId)

*Primary Key: {userId, courseId}*

1. **Assigns** (moderator\_uId, administrator\_uId)

*Primary Key:* moderator\_uId

*Functional Dependency:* moderator\_uId -> administrator\_uId

1. **Approves** (userId, groupClubId)

*Primary Key:* groupClubId

*Functional Dependency:* groupClubId *-> userId*

1. **Teaches** (userId, courseId, groupId, avgGPA)

*Primary Key: {*userId*,* courseId*,* groupId}

*Functional Dependency:* userId*,* courseId*,* groupId *-> avgGPA*

1. **course\_has** (courseId, groupId)

*Primary Key: {*courseId*,* groupId}

1. **Joins** (userId, groupClubId)

*Primary Key: {*userId*,* groupClubId}

1. **Creates** (userId, groupClubId, discussionId)

*Primary Key: {*userId*,* groupClubId*, discussionId}*

1. **InterestGroupClub\_has** (groupClubId, discussionId)

*Primary Key:* discussionId

18. **Alert** (moderatorId, alert, alertTime, forumId)

Primary Key: alertTime

Functional Dependency: alertTime -> alert

19. **Topic** (topicID, topicName, forumID)

Primary Key: topicID

Functional Dependency: topicID -> topicName

20. **Comments** (commentID, commentedBy, topicID, comments, dateAndTime)

Primary Key: commentID

Functional Dependency: commentID -> comments

21. **Criteria** (CriteriaNumber, Action, Description, Points)

Primary Key: CriteriaNumber

Functional Dependency: CriteriaNumber -> {Action, Description, Points}

22. joinsCIG (memberID, groupId, username, password, approvalStatus)

Primary Key: memberId, groupId

Functional Dependency: { memberId, groupId } -> {username, password, approvalStatus}

**Feature Ownership**

|  |  |
| --- | --- |
| ***Feature*** | **Owner** |
| **Insert, update and delete** | *Anshul Gupta* |
| **Authorization and privileges** | *Rittick Datta* |
| **General processes** | *Pavithra Vinay* |

Document Editor: Rittick Datta