|  |  |
| --- | --- |
| **INFO 210 — Database Management Systems Assignment – Milestone 2** |  |

**1. ER diagram modification and description**

# Please submit the modified graph before other modification

**2. Relational Schema**

The following relational schema is presented in bold, and the description of its constraints will be presented in [...] directly below it.

**user (uid: INTEGER, email: CHAR, password: CHAR, profile: CHAR)**

**posts (pid: INTEGER, title: CHAR, content: CHAR, datetime: DATETIME, uid: INTEGER)**

[ (1) uid is the foreign key reference to user and uid is not null.

(2) If the user is deleted, the corresponding post would be deleted(uid is cascade)]

**comments (cid: INTEGER, content: CHAR, datetime: DATETIME, uid: INTEGER, pid: INTEGER)**

[ (1) Respectively, uid and pid are foreign key reference to user and posts and both are not null.

(2) If a post or a user is deleted，respectively, the corresponding comment got deleted(both uid and pid is cascade) ]

**admin (uid: INTEGER, permission\_level: INTEGER)**

[uid is the foreign key reference to user.(on deleted no action)]

**advertisement (id: INTEGER, content: CHAR, uid: INTEGER)**

[ uid is the foreign key reference to admin. (on deleted no action)]

**categories (cat\_id: INTEGER, cat\_name: CHAR, cat\_desc: CHAR)**

**manage (uid: INTEGER, cat\_id: INTEGER)**

[uid and cat\_id are also foreign key reference to admins and categories respectively. ]

**subscribe (uid: INTEGER, cat\_id: INTEGER)**

[ uid and cat\_id are also foreign key reference to users and categories respectively. ]

**classify (pid: INTEGER, cat\_id: INTEGER)**

[pid and cat\_id are also foreign key reference to posts and categories respectively. ]

**groups ( gid: INTEGER, des: CHAR)**

**create ( uid:INTEGER, gid:INTEGER)**

[uid and gid are foreign key reference to admin and posts respectively. ]

**join ( gid: INTEGER，uid: TEGER)**

[ uid and gidare also foreign key reference to users and groups respectively. ]

**announcement (gid: INTEGER, aid: INTEGER, title: CHAR, content: CHAR)**

[ (1) aid is the partial key of announcements and gid is the primary key of groups. The two key combines the primary key of announcements. gid is not null.

(2) If an announcement is deleted, delete the corresponding row from the group table (on delete cascade) ]

**Back list (from:INTEGER, to:INTEGER)**

[ Both from and to (reference to the user) comprise the primary key. ]

**Friend list(uid: INTEGER, other\_uid: INTEGER)**

[ Both uid and other\_uid (reference to user) comprise the primary key. ]

**3.** **Explanation**

**I. 1 to N relations**

Groups & categories: both of the primary key is generated auto incrementally

Posts: Posts and users have a N-to-1 relationship (because one post could only be owned by one user). So we should add uid of user as foreign key to the posts schema and uid should not be null. And the primary key pid is auto incrementally.

Comments: Comments has a many-to-one relationship with posts and user ( because one comment could only be owned by one user and be attached to only one post, while a user could add multiple comments and a post could have multiple comments.) And comments are total participations of both two relations. So we should add uid reference to user and pid reference to posts as foreign keys to the schema and they should not be null. Plus, the primary key cid is auto incrementally.

Announcements: It is a weak entity and it is n to 1 with the groups entity (because each group can generate more than one announcement). The partial key of announcements, aid depends on the groups, because the aid is defined by groups. So, the primary key for announcements is composed of aid and the gid of groups (gid should not be null, and when a group is deleted, the group's announcements are also deleted).

**II. N to N relations**

Subscribe: Subscribe is n to n relationship (because one category could be subscribed by multiple user and one user could subscribe multiple categories). So both uid of user and cat\_id of categories comprise the primary key

Manage: Same as subscribe, uid and cat\_id comprise its primary. However, Its uid is admin's uid.

Join: Join is n to n relationship (because many user could join in one group and a user could join in multiple group). So Both gid of groups and uid of user consist the primary key

Classify: It is n to n relationship, because multiple posts could be classified under one or more category while one category could have many posts. What’s more, posts are total participations in this relation, every posts must be classified. So pid of posts and cat\_id of categories are primary key and cat\_id should not be null.

Friend\_list: It is n to n relationship (one user could add multiple user into friend list and one user could be added by multiple user into friend list). So both uid and other\_uid are primary key.

Black\_list: It is n to n relationship (one user could add multiple user into black list and one user could be added by multiple user into black list). So from and to which store the uid of user are primary key.

**III. ISA**

Admin: It requires a separate schema. Although admin inherits from user, admin obviously cannot coverage user in its entirety. So it inherits user's primary key uid and has its own attribute, permission\_level.