Documentation for Interactive Learning Management System

CS 377 Course Project

Juhao "Jerry" Zhang

Emory University

Contents

1 Introduction

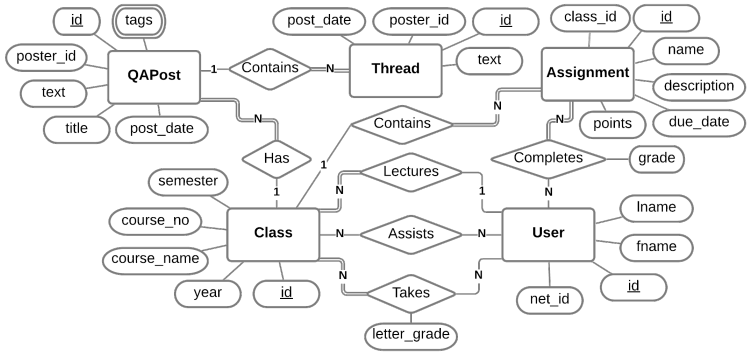
This document contains information pertaining to the final project and its structure, usage notes, and other important annotations.

2 ER Design

The following assumptions are made:

- A class must contain at least one student.
- A class can only be taught by one professor.
- A thread does not contain "reply-to-posts", i.e., threads are only organized chronologically

The entity-relation design diagram is as follows:



3 Relational Model Creation

The initial relational model consists of the following relations:

- User(id, net_id, fname, lname)
- Class(id, course_no, course_name, semester, year, lecturer_id→User.id)
- Assignment(id, name, due_date, description, points, class_id→Class.id)
- QAPost(id, title, post_date, text, poster_id→User.id, class_id→Class.id)
- Thread(id, post_date, text, poster_id→User.id, parent_id→QAPost.id)
- Tags(post_id→QAPost.id, tag)
- Takes(user_id→User.id, class_id→Class.id, letter_grade)
- Completes(user_id→User.id, assignment_id→Assignment.id, grade)
- Assists(user_id→User.id, class_id→Class.id)

4 Database Normalization

To ensure that the database conforms to 3NF, start by ensuring that the relation schema is in 1NF, which states that

- Every attribute in the relational model has atomic values.

By this rule, the relation schema is already in 1NF. 2NF states that

- 1NF is satisfied, and
- Non-candidate-key attributes are not partially dependent on any key of the relation, i.e., every non-key attribute is fully functionally dependent on the primary key.

Since the relation schema is already in 1NF, I argue the second point of the 2NF constraints: none of the non-key attributes are partially dependent on their respective relation's primary key. This can be seen readily where the primary key is a single element, and for those relations that contain a key with size greater than 1, there are only one or no non-key attribute.

Now, to convert to 3NF, its constraints must include that

- 2NF is satisfied, and
- Every non-key attribute is non-transitively dependent on all the keys.

I argue that the relational model proposed above satisfies the 3NF conditions, as no non-key attribute in any of the relations is dependent on another non-key attribute in that same relation.

With the satisfaction of 3NF conditions, the normalization of the database is considered complete.

5 Data Description

All of the `.csv` files are generated using the `extract.py` script and some further manual tweaking on the data source files `canvas.csv` and `qa.csv`. There may exist some slight differences in data due to manual corrections.
For a more graphical view, please see section 6 ("Data Diagram").

Table user
Defines all existing users in Canvas.

Attribute Name	Attribute Type	Description
`id`	`CHAR(10)`	Personal ID for the user. (PK)
`net_id`	`VARCHAR(64)`	NetID used for login.
`fname`	`TINYTEXT`	First name of the user.
`lname`	`TINYTEXT`	Last name of the user.

Table class
Defines all classes that exists in Canvas.

Attribute Name	Attribute Type	Description
`id`	`CHAR(10)`	Unique identifier for the class. (PK)
`course_no`	`VARCHAR(8)`	Course number (e.g., CS377) for the class.
`course_name`	`TINYTEXT`	Course name (e.g., Database Systems) for the class.
`semester`	`VARCHAR(6)`	Semester the course is offered in.
`year`	`INT`	Year the course is offered in.
`lecturer_id`	`CHAR(10)`	The personal ID for the teaching professor. (FK→user.id)

Table assignment
Defines all assignments that exists in Canvas.

Attribute Name	Attribute Type	Description
`id`	`CHAR(10)`	Unique identifier for the assignment. (PK)
`name`	`VARCHAR(64)`	Name of the assignment.
`due_date`	`TIMESTAMP`	Due date of the assignment.
`description`	`TEXT`	Detailed description of the assignment.
`points`	`INT`	Maximum amount of points achievable.
`class_id`	`CHAR(10)`	The class ID that this assignment belongs to. (FK→class.id)

Table qapost
Defines all Q&A posts that exists in the Q&A Corner.

Attribute Name	Attribute Type	Description
`id`	`CHAR(10)`	Unique identifier for the Q&A post. (PK)
`title`	`TINYTEXT`	The title of the Q&A post.
`post_date`	`TIMESTAMP`	The date when this Q&A post was posted.
`text`	`TEXT`	The body of the Q&A post.
`poster_id`	`CHAR(10)`	The personal ID of the poster. (FK→user.id)
`class_id`	`CHAR(10)`	The class ID that this Q&A post belongs to. (FK→class.id)

Table thread

Defines all non-parent posts in the Q&A Corner.

Attribute Name	Attribute Type	Description
id	CHAR(10)	Unique identifier for the thread post. (PK)
post_date	TIMESTAMP	The date when this thread post was posted.
text	TEXT	The body of the thread post.
poster_id	CHAR(10)	The personal ID of the poster. (FK→user.id)
parent_id	CHAR(10)	The Q&A post ID that this thread post belongs to. (FK→qapost.id)

Table tags

Defines all tags associated with posts in the Q&A Corner.

Attribute Name	Attribute Type	Description
post_id	CHAR(10)	The Q&A post's ID. (FK→qapost.id)
tag	VARCHAR(32)	A tag that the post contains.

Table takes

Defines all student-taking-class relationships, and their letter grades in Canvas.

Attribute Name	Attribute Type	Description
user_id	CHAR(10)	Student's personal ID. (PK, FK→user.id)
class_id	CHAR(10)	Class's unique ID. (PK, FK→class.id)
letter_grade	VARCHAR(3)	Letter grade the student obtained for class.

Table completes

Defines all student-completing-assignment relationships, and their numeric grades in Canvas.

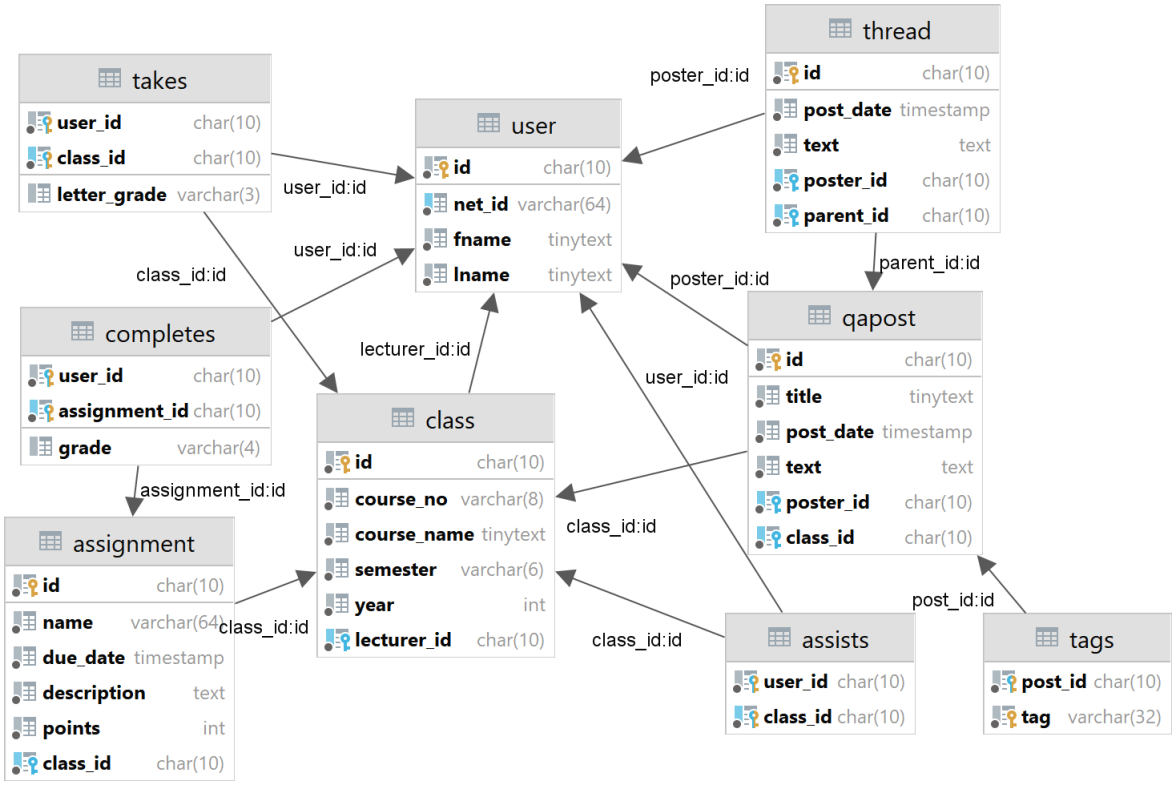
Attribute Name	Attribute Type	Description
user_id	CHAR(10)	Student's personal ID. (PK, FK→user.id)
assignment_id	CHAR(10)	Assignment's unique ID. (PK, FK→assignment.id)
grade	VARCHAR(4)	Grade the student obtained on assignment.

Table assists

Defines all TA-assists-class relationships.

Attribute Name	Attribute Type	Description
user_id	CHAR(10)	TA's personal ID. (PK, FK→user.id)
class_id	CHAR(10)	ID of the class being assisted. (PK, FK→class.id)

6 Data Diagram



7 Project File Directory

File Directory	
`cs377-course-project/`	Project root
`index.php`	Login page (Q10)
`home.php`	Home page (Q10)
`view_course.php`	Course view page (Q7, Q8)
`create_assignment.php`	Assignment creation page (Q8)
`qa_posts.php`	Q&A posts page (Q9)
`view_thread.php`	Single post view page (Q9)
`transcript.php`	Transcript view page (Q11)
`helpers.php`	Helper functions file
`logout.php`	Logout redirect page
`README.txt`	Honor code file
`er_diagram.png`	ER diagram used in Section 2 (“ER Design”)
`canvas_db.png`	Database diagram used in Section 6 (“Data Diagram”)
`extract.py`	Python extraction helper script
`report.pdf`	This report
`*.sql`	SQL database setup/teardown scripts
`*.csv`	Table data files

8 Closing Remarks and Error Handling

- My personal Google Cloud compute instance can be found at `http://jerry.games/377`.
- Please, do *not* run the `extract.py` file. It will corrupt all exported `csv` data. Feel free to take a look inside however.
- There is a small chance that I have left my personal compute instance's address as a database connection argument. If you find your data failing to update on the local server, please navigate to the project root and execute `sed -i 's/jerry.games/localhost/g' *` to replace all occurrences of `jerry.games` with `localhost`.
- A lot of fields do not have error checking for apostrophes. Avoid entering apostrophes if errors occur.