CS2102 Database Systems

Group Project Part #2

Translation of ERD to Relational Database Schema

Submitted by

Team 65

Chew Jun Heng	-	A0252103Y
Neo Haowei	-	A0264683U
Zhang Lanyu	-	A0258235B
Tricia Ang Jean Ee	-	A0254573Y

AY2023/2024 Semester 1

25 September 2023

Database Schema

```
— educator
CREATE TABLE educator (
    staff_nr CHAR(5) PRIMARY KEY,
    first_name TEXT NOT NULL,
    last_name TEXT NOT NULL,
    email TEXT UNIQUE NOT NULL
);

— student
CREATE TABLE student (
    student_nr CHAR(10) PRIMARY KEY,
    first_name TEXT NOT NULL,
    last_name TEXT NOT NULL,
    email TEXT UNIQUE NOT NULL,
    last_active TIMESTAMP NOT NULL
);

— group
CREATE TABLE "group" (
    code INTEGER UNIQUE,
    name TEXT NOT NULL UNIQUE,
    description TEXT,
    PRIMARY KEY (code, name)
);

— question
CREATE TABLE question (
    question_id INTEGER PRIMARY KEY,
    creator_staff_nr CHAR(5) NOT NULL REFERENCES educator (staff_nr),
    statement TEXT NOT NULL,
    description TEXT,
    type TEXT NOT NULL CHECK (type IN ('MCQ', 'MRQ')),
    status TEXT NOT NULL CHECK (status IN ('public', 'private')),
    valid BOOLEAN DEFAULT FALSE
);

— tag
CREATE TABLE tag (
    text TEXT PRIMARY KEY
);
```

```
— quiz
CREATE TABLE quiz (
    quiz_id INTEGER PRIMARY KEY,
    creator_staff_nr CHAR(5) NOT NULL REFERENCES educator (staff_nr),
    name TEXT NOT NULL,
    published BOOLEAN NOT NULL DEFAULT FALSE,
    max_attempts INTEGER NOT NULL DEFAULT 1,
    total_points INTEGER NOT NULL,
    status TEXT NOT NULL CHECK (status IN ('public', 'private')),
    mandatory BOOLEAN,
    time_limit INTEGER,
    avail_from TIMESTAMP NOT NULL DEFAULT NOW(),
    avail_to TIMESTAMP
    CHECK (
        (published = TRUE AND avail_to IS NOT NULL)
        OR (published = false AND avail_to IS NULL)
    )
);

— answer
CREATE TABLE answer (
    answer_id INTEGER,
    question_id INTEGER REFERENCES question (question_id) ON UPDATE CASCADE,
    content TEXT,
    position INTEGER NOT NULL,
    correct BOOLEAN NOT NULL,
    PRIMARY KEY (answer_id, question_id)
);

— submission
CREATE TABLE submission (
    submission_id INTEGER PRIMARY KEY,
    student_nr CHAR(10) NOT NULL REFERENCES student(student_nr),
    quiz_id INTEGER NOT NULL REFERENCES quiz (quiz_id),
    attempt INTEGER
);
```

Junction tables (Join tables)

```
— contains (quiz - question)
CREATE TABLE contains (
    question_id INTEGER NOT NULL REFERENCES question (question_id)
        ON UPDATE CASCADE,
    quiz_id INTEGER NOT NULL REFERENCES quiz (quiz_id)
        ON UPDATE CASCADE,
    mandatory BOOLEAN NOT NULL DEFAULT TRUE,
    position INTEGER NOT NULL,
    points INTEGER NOT NULL CHECK (points >= 0),
    PRIMARY KEY (question_id, quiz_id)
);

— part_of (answer - submission)
CREATE TABLE part_of (
    answer_id INTEGER NOT NULL,
    question_id INTEGER NOT NULL,
    submission_id INTEGER NOT NULL REFERENCES submission (submission_id)
        ON UPDATE CASCADE,
    PRIMARY KEY( answer_id, question_id, submission_id),
    FOREIGN KEY (answer_id, question_id) REFERENCES answer (answer_id,
        question_id) ON UPDATE CASCADE
);

— member_of (student - group)
CREATE TABLE member_of (
    student_nr CHAR(10) NOT NULL REFERENCES student (student_nr)
        ON UPDATE CASCADE,
    group_code INTEGER NOT NULL,
    group_name TEXT NOT NULL,
    FOREIGN KEY (group_code, group_name) REFERENCES "group" (code, name)
        ON UPDATE CASCADE,
    PRIMARY KEY (student_nr, group_code, group_name)
);
```

```
— assigned_to (group - quiz)
CREATE TABLE assigned_to (
    quiz_id INTEGER REFERENCES quiz (quiz_id) ON UPDATE CASCADE,
    group_code INTEGER NOT NULL,
    group_name TEXT NOT NULL,
    FOREIGN KEY (group_code, group_name) REFERENCES "group" (code, name)
        ON UPDATE CASCADE,
    PRIMARY KEY (quiz_id, group_code, group_name)
);

— labels (question - tag)
CREATE TABLE labels (
    tag_text TEXT NOT NULL REFERENCES tag (text) ON UPDATE CASCADE,
    question_id INTEGER NOT NULL REFERENCES question (question_id)
        ON UPDATE CASCADE,
    PRIMARY KEY (tag_text, question_id)
);
```

Justification for non-trivial design decisions

1. Student table: last_active is set to NOT NULL.
    - A student will always have a last active time automatically recorded by the system. When the account is first created, the initial last active time could be the account creation time.

2. Quiz table: time interval and published are made mandatory while max_attempt, mandatory, and time_limit are made optional.
    - "**For each** quiz, the database of Squiz stores its creator, the time interval during which the quiz is available (i.e., from/until) and whether it is published or not." implies **mandatory** attributes.
    - "The creator **can also** specify if the quiz is public…" implies **optional** attributes.

3. Reference to foreign key in weak entity types and junction tables are set to ON UPDATE CASCADE.
    - As each row of these tables are identified with the foreign keys, it is important to preserve the relation/entity whenever there is an update on the foreign keys it is associated with.

4. The join table for ternary relationship "submits" is omitted.
    - Since each submission must correspond to exactly 1 student and 1 quiz, storing the identifier of the student and the quiz as attributes of submission is sufficient to capture the data.
    - In terms of selecting the data, looking for instances with desired values for "student" and "quiz" in the "submission" table is as efficient as doing so in the "submits" join table.
    - Therefore, to reduce complexity, no additional join table is used in this case.

5. Question table: the default value of valid is False.
    - A question can only be valid after creating a valid list of answer(s) for it.
    - The answers can only be created after the question is created, since the answer is a weak entity type.
    - Therefore, when a question is created, it has no answer initially and is invalid. The validity of a question will only be updated with subsequent creation of answers.

Application's Constraints (not captured by relational schema)

1. The total number of points for a quiz needs to be updated when adding or removing a question to or from the quiz, or when changing the points for a question.

2. Constraints on educators' and students' access to private questions and quizzes
    - All creators can use public questions to create quizzes, while private questions can only be used by their creator.
    - A student can only take public quizzes and quizzes assigned to a group to which he/she belongs.

3. A student taking a quiz must answer at least all mandatory questions, if any.

4. A student must not submit a quiz after reaching the maximum number of attempts.

5. Constraints on validity of questions
    - Both MRQs and MCQs require at least two answers to be valid.
    - MCQ requires exactly one correct answer, while MRQ requires one or several correct answers to be valid.