Project Assignment EBA3420 Databases, Spring 2023

Final Report

Introduction:

In this project, we were tasked with creating a web application for collecting and analyzing survey data for courses offered by DataScience@BI. The project involved several tasks, including creating an ER model based on a textual specification of the information system, converting the ER model into a relational model and implementing it in a relational database, populating the database with data, and writing a web application for querying and modifying the database. This report will describe the steps we took to complete the project and present the deliverables.

A. Entity Relationship (ER) Diagram:

We began the project by creating an ER model for the database. We identified the entities that needed to be stored in the database and their relationships. We then converted the ER model into a relational model and created an SQLite database with SQL DDL statements to create the tables. We ensured that every table in the database was in 3NF to avoid data redundancy. We also made several assumptions during the project and documented them in the report.

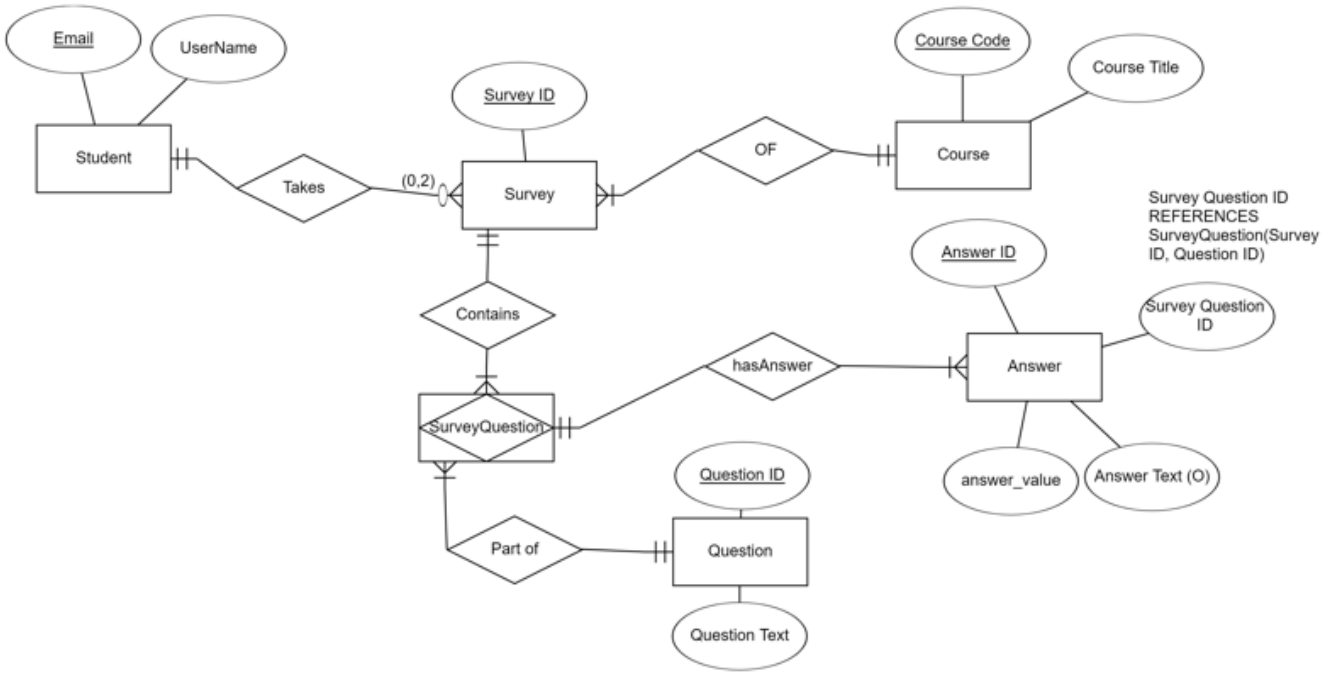


Fig.1 ER Diagram

Entities: Student, Survey, Course, Answer, Question, SurveyQuestion

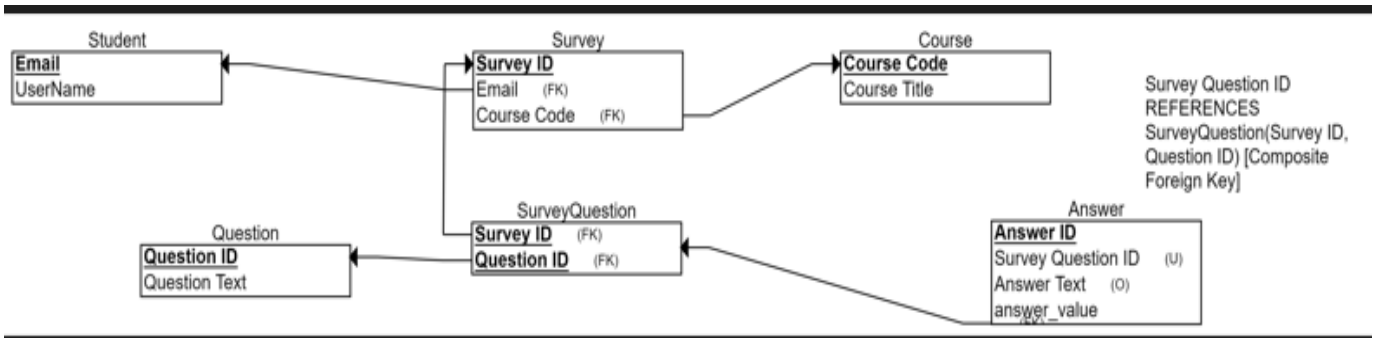
Relationships:

1. Student takes Survey
2. Survey is of Course
3. Survey contains Questions with Answer
    - To represent this relationship, create a new associative entity called 'SurveyQuestion' with the following attributes: survey_id (foreign key referencing Survey.survey_id) question_id (foreign key referencing Question.question_id)
    - This creates a relationship between Survey and Question entities, where each survey can contain multiple questions, and each question can be part of multiple surveys.
    - Each Survey ID & Question ID combined gives us a compositive foreign key for the answer table since there is an answer for each Survey's Question.

Cardinalities:

→ One student can take multiple surveys
→ Each survey is taken by at least one student
→ A survey question can have multiple-answer choices
→ Each answer choice is associated with one survey question
→ A survey can have multiple questions
→ Each SurveyQuestion record is associated with one survey
→ A question can be part of multiple surveys
→ Each SurveyQuestion record is associated with one question

B. Relational Database Schema:



1. **Student (email, user_name)**

- email: TEXT (Primary Key)

- user_name: TEXT (Not Null)

2. **Course (course_code, course_title)**

- course_code: TEXT (Primary Key)
- course_title: TEXT (Not Null)

3. **Survey (survey_id, course_code, email)**

- survey_id: INTEGER (Primary Key, Autoincrement)
- course_code: TEXT (Foreign Key referencing Course)
- email: TEXT (Foreign Key referencing Student)

4. **Question (question_id, question_text)**

- question_id: INTEGER (Primary Key, Autoincrement)
- question_text: TEXT (Not Null)

5. **SurveyQuestion (survey_id, question_id)**

- survey_id: INTEGER (Foreign Key referencing Survey, Composite Primary Key)
- question_id: INTEGER (Foreign Key referencing Question, Composite Primary Key)

6. **Answer (survey_id, question_id, answer_value, answer_text)**

- survey_id: INTEGER (Foreign Key referencing SurveyQuestion, Composite Primary Key)
- question_id: INTEGER (Foreign Key referencing SurveyQuestion, Composite Primary Key)
- answer_value: INTEGER (Not Null)
- answer_text: TEXT

C. Additional Assumptions:

Throughout the process of designing the relational database and ER diagrams, several assumptions have been made:

1. A student can take multiple surveys, but only one survey per course.

2. Each survey has a unique set of questions related to a specific course. These questions can be shared across multiple surveys for different courses.

3. Each survey question has a set of predetermined answer choices with a corresponding numeric value (e.g., Strongly Agree, Agree, etc.). The answer choices are the same for all questions and surveys.

4. The survey is not anonymous, and the student's email address is used to identify a student uniquely.

5. The relationships between the entities are as follows:

- A one-to-many relationship between Student and Survey entities, where each survey is taken by one student, and each student can take multiple surveys.
- A many-to-many relationship between Survey and Question entities, represented by the SurveyQuestion associative entity.
- A one-to-many relationship between SurveyQuestion and Answer entities, where each survey question can have multiple answer choices.

6. For simplicity, we have assumed that the course information (course code and title) is already stored in the database.

Hence, these assumptions have guided the design and relationships among entities in the ER diagram and relational database schema.

D. SQL DDL and SQL DML Statements:

We implemented using SQL DDL AND DML Statements. We manually populated the database with some initial amount of data of our choice, including all the survey questions and answer choices mentioned in the assignment. We used SQL DML statements to insert the data into the tables.

-- Create tables

```
CREATE TABLE Student (
    email TEXT PRIMARY KEY,
    user_name TEXT NOT NULL
);

CREATE TABLE Course (
    course_code TEXT PRIMARY KEY,
    course_title TEXT NOT NULL
);

CREATE TABLE Survey (
    survey_id INTEGER PRIMARY KEY AUTOINCREMENT,
    course_code INTEGER NOT NULL,
    email TEXT NOT NULL,
    FOREIGN KEY (course_code) REFERENCES Course(course_code),
    FOREIGN KEY (email) REFERENCES Student(email)
);

CREATE TABLE Question (
    question_id INTEGER PRIMARY KEY AUTOINCREMENT,
```

```
    question_text TEXT NOT NULL
);

CREATE TABLE SurveyQuestion (
    survey_id INTEGER NOT NULL,
    question_id INTEGER NOT NULL,
    PRIMARY KEY (survey_id, question_id),
    FOREIGN KEY (survey_id) REFERENCES Survey(survey_id),
    FOREIGN KEY (question_id) REFERENCES Question(question_id)
);

CREATE TABLE Answer (
    survey_id INTEGER NOT NULL,
    question_id INTEGER NOT NULL,
    answer_value INTEGER NOT NULL,
    answer_text TEXT,
    PRIMARY KEY (survey_id, question_id),
    FOREIGN KEY (survey_id, question_id) REFERENCES
SurveyQuestion(survey_id, question_id)
);
```

-- Insert initial data

```
------Now, let's populate the database with some initial data:

-- Insert students
INSERT INTO Student (email, user_name) VALUES ('student1@example.com',
'John Doe');
INSERT INTO Student (email, user_name) VALUES ('student2@example.com',
'Jane Smith');

-- Insert courses
INSERT INTO Course (course_code, course_title) VALUES ('EBA3420',
'Databases');
INSERT INTO Course (course_code, course_title) VALUES ('EBA3400', 'Python
Programming');

-- Insert questions
INSERT INTO Question (question_text) VALUES ('I have got a clear idea of
what is expected of me in this course.');
```

```
INSERT INTO Question (question_text) VALUES ('The lecturer(s) in this
course presented the course contents well.');
INSERT INTO Question (question_text) VALUES ('The course literature has
supported my learning.');
INSERT INTO Question (question_text) VALUES ('I have acquired new and
relevant knowledge in the course area.');

--Insert Surveys
INSERT INTO Survey (course_code, email) VALUES ('EBA3420',
'student1@example.com');
INSERT INTO Survey (course_code, email) VALUES ('EBA3400',
'student2@example.com');

-- Insert survey questions for each course
INSERT INTO SurveyQuestion (survey_id, question_id) VALUES
(1, 1), (1, 2), (1, 3), (1, 4),
(2, 1), (2, 2), (2, 3), (2, 4);

-- Insert answer choices
INSERT INTO Answer (survey_id, question_id, answer_value, answer_text)
VALUES
(1, 1, 4, 'Agree'),
(1, 2, 5, 'Strongly Agree'),
(1, 3, 3, 'Neutral'),
(1, 4, 4, 'Agree'),
(2, 1, 3, 'Neutral'),
(2, 2, 2, 'Disagree'),
(2, 3, 5, 'Strongly Agree'),
(2, 4, 1, 'Strongly Disagree');
```

Task 3: Implementing the web application

Finally, we implemented the web application described in the project using Flask. We created a homepage with a welcome message and two buttons, one for taking the survey and the other for viewing survey results. Clicking the "Take a survey" button took the user to a survey page containing a survey form. Clicking the "View survey results" button took the user to the course list page containing a list of links to all the courses.

Conclusion:

In conclusion, we successfully completed all the tasks of the project, including creating an ER model, converting it into a relational model, implementing it in an SQLite database, populating the database with data, and writing a web application for querying and modifying the database. The deliverables of the project include an ER diagram, a relational database schema, all additional assumptions, all SQL DDL and SQL DML statements, the SQLite database, and the source code of the web application. Through this project, we gained practical experience in working with databases and web applications, which will be valuable in our future endeavors.