School Management System Database

Danial Tahir

Student Number: 920838929

Github: DanTahir

Checkpoint #	Date Submitted
Checkpoint 1	2/22/2024
Checkpoint 2	3/3/2024

Table of Contents

Project Description

The marketplace for digital management solutions to school management has many entrants promising a wide variety of services, so a new entrant providing a School Management System Database needs to provide a product that offers the full range of services required to run a school, from course management to teacher management to staff management to resource management to parking management. Every aspect of the modern campus has to be managed to perfection for a modern school to function and a platform needs to be able to provide the power and flexibility required to deal with the deep complexity inherent in this task.

School Management System Database provides powerful features allowing for the complete management of an educational system. It allows creation of students, teachers, and staff members, allowing you to manage your school's diverse personnel. We include the powerful feature of categories enabling you to further manage your registered users.

School Management System Database has powerful course management features including the proctoring of tests and the collection of assignments. Not only that, it enables you to manage all the resources you'll need to administrate a class, from classroom space to classroom equipment to class required reading and other required resources. It even includes parking management.

Many products including Canvas and Dreamclass would be enhanced by our comprehensive resource management options in addition to our fully featured offerings in other areas. We believe School Management System Database is a powerful product that can compete in the marketplace.

Functional Requirements

1. Student (Strong)
    1.1. A student shall have many majors.
    1.2. A student shall take many class sections.
    1.3. A student shall complete many assignments.
    1.4. A student shall complete many quiz answers.
    1.5. A student shall view many documents.
    1.6. A student shall make many student payments in many semesters.
    1.7. A student shall belong to many organizations.
2. Teacher (Strong)
    2.1. A teacher shall have many teacher roles.
    2.2. A teacher shall have no more than one department.
    2.3. A teacher shall receive many teacher payments.
    2.4. A teacher shall have no more than one parking lot.
    2.5. A teacher shall teach many class sections.
    2.6. A teacher shall create many modules.
    2.7. A teacher shall create many assignments.
    2.8. A teacher shall create many quizzes.
    2.9. A teacher shall create many quiz questions.
    2.10. A teacher shall create many quiz answers.
    2.11. A teacher shall create many documents.
    2.12. A teacher shall assign many modules to many class sections.
    2.13. A teacher shall assign many assignments to many modules.
    2.14. A teacher shall assign many quizzes to many modules.
    2.15. A teacher shall assign many documents to many modules.
    2.16. A teacher shall assign many quiz questions to many quizzes.
    2.17. A teacher shall assign many quiz answers to many quiz questions.
    2.18. A teacher shall create many required resources.
    2.19. A teacher shall assign many required resources to many class sections.
    2.20. A teacher shall oversee many organizations.
3. Staff Member (Strong)
    3.1. A staff member shall have many staff roles.
    3.2. A staff member shall have no more than one department.
    3.3. A staff member shall receive many staff payments.
    3.4. A staff member shall have no more than one parking lot.
4. Teacher Role (Strong)
    4.1. A teacher role shall have many teachers.
5. Teacher Payment (Weak)
    5.1. A teacher payment shall be made to only one teacher.
6. Staff Role (Strong)
    6.1. A staff role shall have many staff members.

7. Staff Payment
    7.1. A staff payment shall be made to only one staff member.
8. Major (Strong)
    8.1. A major shall have many students.
    8.2. A major shall have many required classes.
9. Subject (Strong)
    9.1. A subject shall have many classes.
    9.2. A subject shall have no more than one department.
10. Department
    10.1. A department shall have many teachers.
    10.2. A department shall have many staff members.
    10.3. A department shall have many subjects.
11. Class (Strong)
    11.1. A class shall have no more than one subject.
    11.2. A class shall be required by many majors.
    11.3. A class shall have many class sections.
12. Class Section (Strong)
    12.1. A class section shall have no more than one class.
    12.2. A class section shall have many students.
    12.3. A class section shall have many teachers.
    12.4. A class section shall have many modules.
    12.5. A class section shall have no more than one classroom.
    12.6. A class section shall have many required resources.
    12.7. A class section shall have no more than one semester.
13. Module (Strong)
    13.1. A module shall have no more than one class.
    13.2. A module shall have many assignments.
    13.3. A module shall have many quizzes.
    13.4. A module shall have many documents.
14. Assignment (Strong)
    14.1. An assignment shall belong to no more than one module.
    14.2. An assignment shall be submitted by many students.
15. Quiz (Strong)
    15.1. A quiz shall belong to no more than one module.
    15.2. A quiz shall have many quiz questions.
16. Quiz Question (Strong)
    16.1. A quiz question shall have no more than one quiz.
    16.2. A quiz question shall have many quiz answers.
17. Quiz Answer (Strong)
    17.1. A quiz answer shall have no more than one quiz question.
    17.2. A quiz answer shall be answered by many students.
18. Document (Strong)
    18.1. A document shall belong to no more than one module.
19. Classroom (Strong)

    19.1. A classroom shall be assigned to many class sections.
    19.2. A classroom shall have many classroom resources.

20. Required Resources (Strong)
    20.1. A required resource shall be assigned to no more than one class section.

21. Classroom Resource (Strong)
    21.1. A classroom resource shall be assigned to many classrooms.

22. Semester (Strong)
    22.1. A semester shall belong to many class sections.
    22.2. A semester shall belong to many student payments.

23. Parking Lot (Strong)
    23.1. A parking lot shall be assigned to many students.
    23.2. A parking lot shall be assigned to many teachers.
    23.3. A parking lot shall be assigned to many staff members.

24. Organization
    24.1. An organization shall have many students.
    24.2. An organization shall have many teachers.

25. Admin (Strong)
    25.1. An admin shall create many students.
    25.2. An admin shall create many teachers.
    25.3. An admin shall create many staff members.
    25.4. An admin shall create many classrooms.
    25.5. An admin shall create many classroom resources.
    25.6. An admin shall create many parking lots.
    25.7. An admin shall create many classes.
    25.8. An admin shall create many class sections.
    25.9. An admin shall create many organizations.
    25.10. An admin shall assign many class sections to many classes.
    25.11. An admin shall assign many classrooms to many classes.
    25.12. An admin shall assign many teachers to many class sections.
    25.13. An admin shall assign many teacher roles to many teachers.
    25.14. An admin shall assign many departments to many teachers.
    25.15. An admin shall assign many positions to many staff members.
    25.16. An admin shall assign many students to many parking lots.
    25.17. An admin shall assign many teachers to many parking lots.
    25.18. An admin shall assign many staff members to many parking lots.
    25.19. An admin shall assign many teachers to many organizations.
    25.20. An admin shall assign many students to many organizations.

Non-functional Database Requirements

1. Performance
   1.1. The database shall respond to user requests within 2 seconds.
   1.2. The database shall respond to 95% of user requests within 100 milliseconds.
   1.3. The system shall be able to handle a minimum of 50 concurrent users without impacting performance.
   1.4. There shall be no caching of results to improve performance.
   1.5. Regular load testing shall be carried out to make sure the database can handle loads during peak events.
2. Security
   2.1. All passwords shall be stored encrypted.
   2.2. All passwords shall be transmitted encrypted.
   2.3. The database shall implement role-based permissions so that only approved users can perform necessary functions.
   2.4. Steps shall be taken to protect queries against injection attacks.
   2.5. Regular security audits shall be performed to assess and mitigate vulnerabilities.
3. Scalability
   3.1. The database shall run on one server and shall not be scalable to run on multiple servers.
   3.2. The server shall allocate resources to the database on a fixed basis and shall be restarted if those resources need to be changed.
   3.3. There will be no caching.
   3.4. Connection pooling shall not be used and all connections shall be handled as new.
   3.5. The database shall run as a single instance.
4. Capability
   4.1. The database shall be compatible with many frontends.
   4.2. The database shall have a user-friendly interface.
   4.3. The system shall provide simple reports of basic database information.
   4.4. The database shall support integration with other systems such as email.
   4.5. The database shall support basic data analysis such as querying and filtering data.
5. Environmental
   5.1. Software and hardware configurations shall be optimized to minimize energy usage.
   5.2. Paperless operations shall be encouraged by having all documentation be electronic.
   5.3. Obsolete hardware shall be disposed of responsibly via recycling.
   5.4. Virtualization shall be used where possible to minimize resource consumption.
   5.5. Remote access shall be enabled so that administrators do not need to travel to the database, minimizing carbon footprint.
6. Coding Standards
   6.1. Consistent naming conventions for database entities including tables, columns, and indexes shall be enforced.

6.2. Code shall be documented including documentation of queries, schemas, and stored procedures to facilitate teamwork and future development.

6.3. Robust error handling with meaningful error messages shall be implemented.

6.4. Code shall not be optimized but shall be provided in working order.

6.5. Best security practices including input validation and parameterized queries shall be followed.

7. Media Storage and Privacy

7.1. Privacy shall be maintained by not tracking IP addresses.

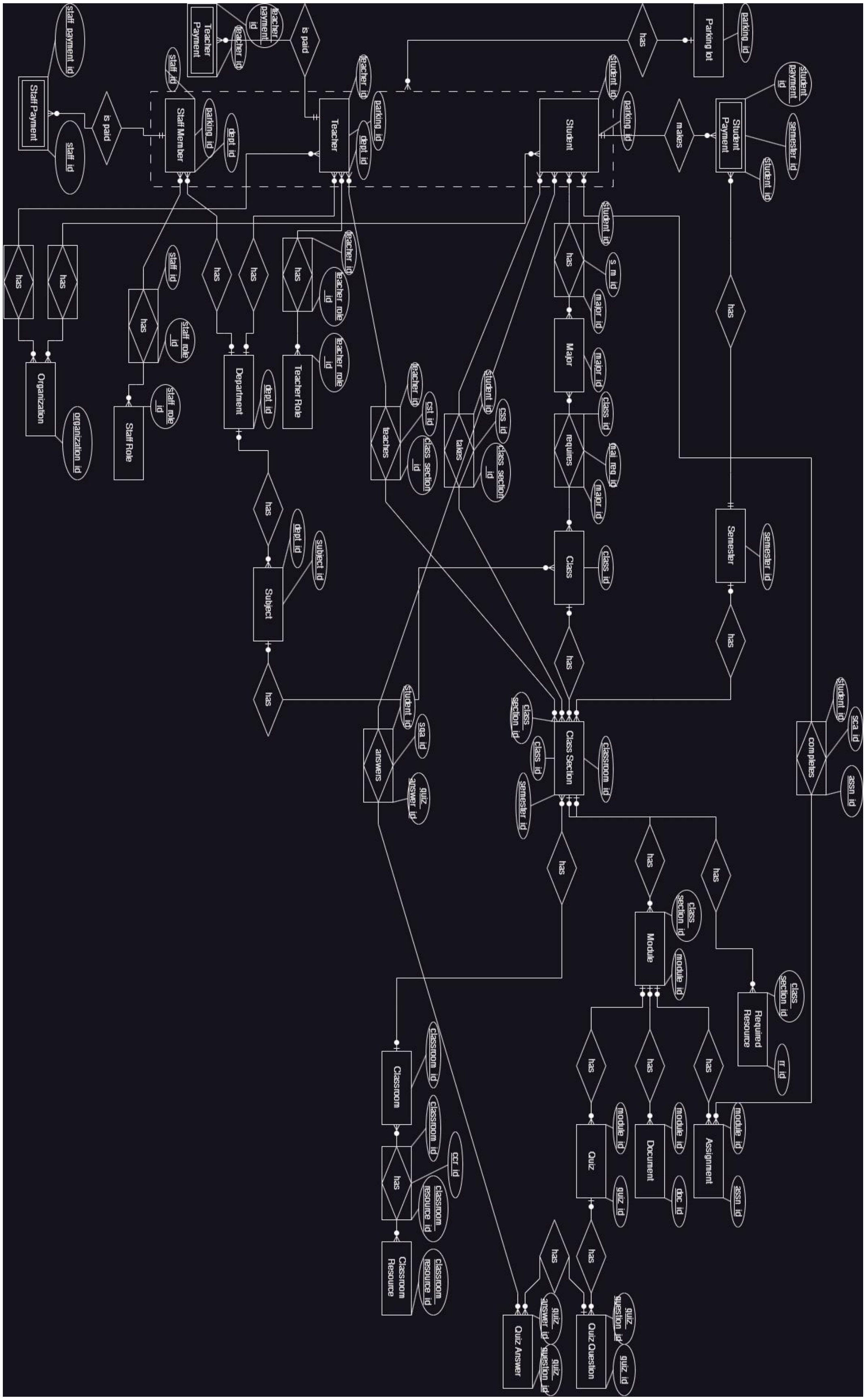
7.2. There shall be a privacy policy that details how data is stored and used.

7.3. Media shall not be stored in the database.

7.4. Links to media may be stored in the database.

7.5. Each table shall be allocated at least 10Mb of disk space.

Entity Relationship Diagram



Entity Description

1. Student (Strong)
   * student_id (key, numeric)
   * date_of_birth (multivalue, numeric)
   * date_of_enrollment (multivalue, timestamp)
   * name (composite, alphanumeric)
   * password (alphanumeric)
   * email (key, alphanumeric)
   * parking_lot_id (key, alphanumeric)
2. Teacher (Strong)
   * teacher_id (key, numeric)
   * date_of_hiring (multivalue, timestamp)
   * name (composite, alphanumeric)
   * password (alphanumeric)
   * email (key, alphanumeric)
   * dept_id (key, numeric)
   * parking_lot_id (key, alphanumeric)
3. Staff Member (Strong)
   * staff_id (key, numeric)
   * date_of_hiring (multivalue, timestamp)
   * name (composite, alphanumeric)
   * password (alphanumeric)
   * email (key, alphanumeric)
   * dept_id (key, numeric)
   * parking_lot_id (key, alphanumeric)
4. Teacher Role (Strong)
   * teacher_role_id (key, numeric)
   * name (composite, alphanumeric)
   * description (alphanumeric)
   * payscale (numeric)
5. Staff Role (Strong)
   * staff_role_id (key, numeric)
   * name (composite, alphanumeric)
   * description (alphanumeric)
   * payscale (numeric)
6. Teacher-Teacher Role (Weak)
   * ttr_id (key, numeric)
   * teacher_id (key, numeric)
   * teacher_role_id (key, numeric)
7. Staff-Staff Role (Weak)
   * ssr_id (key, numeric)
   * staff_id (key, numeric)
   * staff_role_id (key, numeric)

8. Department (Strong)
    * dept_id (key, numeric)
    * name (alphanumeric)
    * description (alphanumeric)
9. Major (Strong)
    * major_id (key, numeric)
    * name (alphanumeric)
    * description (alphanumeric)
10. Student-Major (Weak)
    * student_major_id (key, numeric)
    * student_id (key, numeric)
    * major_id (key, numeric)
11. Subject (Strong)
    * subject_id (key, numeric)
    * name (alphanumeric)
    * description (alphanumeric)
    * dept_id (key, numeric)
12. Class (Strong)
    * class_id (key, numeric)
    * name (alphanumeric)
    * description (alphanumeric)
    * credit_hours (numeric)
    * cost (numeric)
    * teacher_salary (numeric)
    * subject_id (key, numeric)
13. Major Requirement (Weak)
    * major_req_id (key, numeric)
    * major_id (key, numeric)
    * class_id (key, numeric)
14. Class Section (Strong)
    * class_section_id (key, numeric)
    * name (alphanumeric)
    * description (alphanumeric)
    * days (composite, alphanumeric)
    * hours (composite, numeric)
    * classroom_id (key, numeric)
    * semester_id (key, numeric)
15. Class Section Student (Weak)
    * css_id (key, numeric)
    * class_section_id (key, numeric)
    * student_id (key, numeric)
16. Class Section Teacher (Weak)
    * cst_id (key, numeric)
    * class_section_id (key, numeric)

    * teacher_id (key, numeric)

17. Semester (Strong)
    * semester_id (key, numeric)
    * name (composite, alphanumeric)
    * description (alphanumeric)
18. Classroom (Strong)
    * classroom_id (key, numeric)
    * building (alphanumeric)
    * number (alphanumeric)
19. Module (Strong)
    * module_id (key, numeric)
    * name (alphanumeric)
    * description (alphanumeric)
    * class_section_id (key, alphanumeric)
20. Document (Strong)
    * document_id (key, numeric)
    * name (alphanumeric)
    * description (alphanumeric)
    * link (alphanumeric)
    * module_id (key, numeric)
21. Assignment (Strong)
    * assignment_id (key, numeric)
    * name (alphanumeric)
    * description (alphanumeric)
    * link (alphanumeric)
    * possible_score (numeric)
    * module_id (key, numeric)
22. Quiz (Strong)
    * quiz_id (key, numeric)
    * name (alphanumeric)
    * description (alphanumeric)
    * possible_score (numeric)
    * module_id (key, numeric)
23. Quiz Question (Strong)
    * quiz_question_id (key, numeric)
    * title (alphanumeric)
    * question (alphanumeric)
    * quiz_id (key, numeric)
24. Quiz Answer (Strong)
    * quiz_answer_id (key, numeric)
    * answer (alphanumeric)
    * is_correct (numeric)
    * quiz_question_id (key, numeric)
25. Student-Completed-Assignment (Weak)

* sca_id (key, numeric)
* student_id (key, numeric)
* assignment_id (key, numeric)
* submittal_link (alphanumeric)
* grade (numeric)

26. Student Quiz Answer (Weak)
    * sqa_id (key, numeric)
    * student_id (key, numeric)
    * quiz_answer_id (key, numeric)
27. Required Resource (Strong)
    * req_resource_id (key, numeric)
    * category (alphanumeric)
    * name (alphanumeric)
    * description (alphanumeric)
    * class_section_id (key, numeric)
28. Classroom Resource (Strong)
    * class_resource_id (key, numeric)
    * category (alphanumeric)
    * name (alphanumeric)
    * description (alphanumeric)
    * quantity (numeric)
29. Classroom-Classroom Resource (Weak)
    * ccr_id (key, numeric)
    * classroom_id (key, numeric)
    * class_resource_id (key, numeric)
    * quantity_assigned (numeric)
30. Parking Lot (Strong)
    * lot_id (key, numeric)
    * name (alphanumeric)
    * spaces_available (numeric)
31. Student Payment (Weak)
    * student_payment_id (key, numeric)
    * student_id (key, numeric)
    * payment_amount (key, numeric)
    * date_made (multivalue, timestamp)
    * semester_id (key, numeric)
32. Teacher Payment (Weak)
    * teacher_payment_id (key, numeric)
    * teacher_id (key, numeric)
    * payment_amount (key, numeric)
    * date_made (multivalue, timestamp)
33. Staff Payment (Weak)
    * staff_payment_id (key, numeric)
    * student_id (key, numeric)

    * payment_amount (key, numeric)
    * date_made (multivalue, timestamp)

34. Organization
    * organization_id (key, numeric)
    * name (alphanumeric)
    * description (alphanumeric)
35. Organization-Student
    * org_student_id (key, numeric)
    * organization_id (key, numeric)
    * student_id (key, numeric)
36. Organization-Teacher
    * org_teacher_id (key, numeric)
    * organization_id (key, numeric)
    * student_id (key, numeric)