COMP3335 - Project

Section 1: Overview

Databases are widely used in universities now. Many records, such as student information and academic records, rely on databases for storage. This project is about data storage in a university and discusses how to protect sensitive information.

Section 2: Deadlines

1. Submission of required materials: 11:59 PM, November 22nd, 2025.
    a. The materials include your report, codes and demonstration video.

Section 3: Objectives

1. Design a database for a university, *ComputingU*.
2. Data in this database should be secured.
    a. Sensitive data should be encrypted.
    b. The access to data should be controlled based on job roles, according to the Least Privilege Principle.
3. A simple web interface should be provided to help users interact with the database.
    a. The web interface must be safe from SQL injection attacks.

Section 4: Roles

1. *student*: maintains his/her personal information and accesses his/her grades and disciplinary records.
2. *ARO*: manages students' grades, including adding, deleting, querying and modifying them.
3. *guardian*: maintains his/her personal information and accesses his/her child's grades and disciplinary records.
4. *DRO*: manages students' disciplinary records, including adding, deleting, querying and modifying them.

Section 4: Tables

This database consists of six tables.

1. *students*: personal information of students
2. *guardians*: personal information of guardians

3. *staffs*: personal information of staff members
4. *courses*: the course information
5. *grades*: the course grades of all students
6. *disciplinary_records*: the records of disciplinary actions to students

The details of tables are listed below.

***Table 1*:**

Name: students

Columns:

1. *id*: the student ID (surrogate key)
2. *last_name*: the last name of a student
3. *first_name*: the first name of a student
4. *gender*: the gender of a student
5. *identification_number*: the number of a student's identification document, such as Hong Kong Identity Card number or Passport number
6. *address*: the address of a student
7. *email*: the email of a student
8. *phone*: the phone number of a student
9. *enrollment_year*: the year of a student's enrollment
10. *guardian_id*: the ID of a student's guardian (foreign key)
11. *guardian_relation*: the relationship to the guardian

Table 2:

Name: guardians

Columns:

1. *id*: the guardian ID (surrogate key)
2. *last_name*: the last name of a guardian
3. *first_name*: the first name of a guardian
4. *email*: the email of a guardian
5. *phone*: the phone number of a guardian

Table 3:

Name: staffs

Columns:

1. *id*: the staff ID (surrogate key)
2. *password*: the password of a staff member
3. *last_name*: the last name of a staff member
4. *first_name*: the first name of a staff member
5. *gender*: the gender of a staff member
6. *identification_number*: the number of a staff member's identification document, such as Hong Kong Identity Card number or Passport number
7. *address*: the address of a staff member
8. *email*: the email of a staff member
9. *phone*: the phone number of a staff member
10. *department*: the department of a staff member
11. *role*: the role of a staff member

Table 4:

Name: courses

Columns:

1. *id*: the course ID (surrogate key)
2. *course_name*: the name of the course

Table 5:

Name: grades

Columns:

1. *id*: the record ID (surrogate key)
2. *student_id*: the student ID (foreign key)
3. *course_id*: the course ID (foreign key)
4. *term*: the term of a course (e.g., "202526S1" is Semester 1 of 2025/26)
5. *grade*: the grade of this student on this course (e.g., "A", "A-")
6. *comments*: comments from the staff member.

Table 6:

Name: disciplinary_records

Columns:

1. *id*: the record ID (surrogate key)
2. *student_id*: the student ID (foreign key)
3. *date*: the date of the record
4. *staff_id*: the responsible staff of this case (foreign key)
5. *descriptions*: case description from the responsible staff.

Section 5: Requirements

1. Use Percona Server as DBMS (similar to MySQL).
2. Design the tables based on the previous description.
    a. Creates some sample users in different roles.
    b. Note that DBMS accounts may represent parts of the application, not end users.
3. Sensitive data should be protected.
    a. Identify the columns that should be protected, based on column description.
        i. Please include your choices and your reasons in the report.
        ii. Encrypting an entire table is **prohibited**.
        iii. There is no single correct answer.
    b. Use TDE (supported by Percona) or MySQL's encryption functions.
    c. If you use MySQL's encryption functions (such as AES_ENCRYPT), the keys should not be stored in the database (can be stored in hard drives alternatively).
    d. Keys should be managed so that roles and keys match.
4. A simple web interface should be provided, in which
    a. enables users to log in
        i. SQL injections should be prevented.
        ii. Passwords should be stored adequately.
    b. enables users to perform allowed operations
        i. If any foreign keys are displayed in your result, the corresponding names should be displayed instead (e.g., *course_name* should be displayed if the result includes *course_id*).
        ii. Hint: Due to foreign keys, a user may need the access privilege to multiple tables to query.
    c. Your web interface should prevent users from performing unauthorized operations.

    d. No specific required frameworks. To reduce your workload, there is no need to make the interface very beautiful.
5. The access to database should be monitored.
    a. Any inappropriate access, SQL injection attempts, or policy violations must be logged.
    b. Any modifications to the data should be logged.

Section 7: Report File

Your report should be within 8 pages. More pages do not lead to higher grades.

There is no specific format. Please explain your solution and justify your choices.

Please show autonomy and creativity when requirements allow.

There are various ways you can implement your project. It is encouraged to think more.

Section 8: Demonstration Video

A team should record a 5-min demonstration video to demonstrate your solution, which shows if your application works or not. You should also demonstrate at least 3 attack trials (e.g., some SQL injection attacks and other attacks which try to perform unauthorized operations) and show your application can prevent them.

Section 9: Code

Your code must contain all the source codes, an SQL file that can be imported to Percona Database and **a step-by-step document about how to deploy and use your application**.

This document must be able to guide a person to deploy and run your application from a **clear** Windows 11.0 OS (i.e., no assumptions on pre-installed software/libraries), i.e., your document should guide a person to install the required software/libraries and use your application.

Your code should be well documented that is comprehensive comments and is readable.

Section 10: Submission Guidelines

- Create a folder with the name *TeamName*
    - Put all your code in a folder with the name *code*

    - Rename your report with the name *report* (with the extension name, such as *pdf*)
    - Rename your video with the name *video* (with the extension name, such as *mp4*)
    - Put *code*, *report* and *video* in the folder *TeamName*
    - You should replace *TeamName* with your actual team's name, which will be released after registration period.
- Compress this folder as one zip file.
- Follow the example below to name your zip file by replacing *TeamName* with your actual team's name:
    - *TeamName.zip*
- Your submission should be submitted by your **TEAM COORDINATOR** before the deadline.

Section 11: Grading Rubrics

- Code (40%)
- Report (40%)
- Demonstration (20%)

Outcome Presentations	%	**A+/A/A-**	**B+/B/B-**	**C+/C/C-**	**D+/D**	**F**
Code	40%	Programs are well-organized, making good use of whitespace and comments. Variables have helpful names.	Programs are well-organized, easy to read and understand.	Programs can be read and are in a logical order.	Programs are runnable but barely readable.	Absent
Report	40%	Excellent, comprehensive and in-depth analysis with concrete facts/evidence	Clear analysis with good analysis supported by plenty of facts/evidence	Basic analysis with some level of facts/evidence	Barely relevant analysis with minimal facts/evidence	Absent
Demonstration	20%	Very clear and logical	Good, easy to follow	Understandable, structured	Barely understandable	Absent