|  |
| --- |
| Security and Data Integrity Analysis |
| Gradebook Assistant |
| Rose-Hulman Institute of Technology – CSSE 333 |

|  |
| --- |
| Vince Anderson, Kiana Caston, and Linde Schaffer  Spring 2017 |

Contents

[Executive Summary 1](#_Toc481663984)

[Privacy Analysis 1](#_Toc481663985)

[Security Analysis 2](#_Toc481663986)

[Entity Integrity Analysis 2](#_Toc481663987)

[Referential Integrity Analysis 3](#_Toc481663988)

[Business Rule Integrity Analysis 4](#_Toc481663989)

[Invalid Data Entry 4](#_Toc481663990)

# Executive Summary

This purpose of this document is to give an overview of how our team plans to handle our data. There are many key components that go into protecting our data and ensuring all data is legitimate. We will discuss what personal data needs to be protected in the Privacy Analysis section. In the Security Analysis, we will discuss how we will be protecting the private data. Each table in the database has specific constraints, which will be laid out in the Entity Integrity Analysis and Referential Integrity Analysis sections of this document. A brief explanation of our database and its use of Business Rule Integrity Analysis will be stated. Finally, we will go over how to handle invalid data entry.

# Privacy Analysis

For our database, all grades on assignments and points awarded should only be accessible to the professor who created the assignment and the student who completed the assignment. Here are some particular rules: Each professor has access to the students’ grades for their particular section of course. The professor does not have access to a student’s grade if he or she is not enrolled in their course.  Each student has access to their own grades and assignments that have been assigned to them.  We don’t want other students to be able to access other student’s information because that is personal. Each users’ pin number will be private as well and should not be shared with others. The admin of the database will not have access to the user’s pin number either, as a hash function will be used to store it.

# Security Analysis

Each user has a restricted amount of access to information and data.  Professors can only alter data they have permissions on.  Students should not be allowed to alter anything, except when answering questions.  All users can only see items following the constraints laid out in the Privacy Analysis. The user must use their pin in the application in order to determine what information they can access and alter. In order to respond to a potential breach to obtain or alter a user pin number, we will use a hash function to store the user’s pin, so their pin number will not be stored directly into the database.

# Entity Integrity Analysis

1. For the Assignment table, AssignmentID must be an int, which automatically increments by 1, that is the primary key.  TotalPoints, ProfessorID, and SectionID must be ints that are not null. ProfessorID is a foreign key to the Professor table and is unique. SectionID is a foreign key to the Section table and is unique.
2. For the Assignment For Section table, AssignmentID is a foreign key to the Assignment table, and is an int, not null, and is unique. SectionID is a foreign key to the Section table, and is also an int, not null, and is unique.  Both AssignmentID and SectionID act as the primary keys.
3. For the Assignment Has Questions table, AssignmentID is a foreign key to the Assignment table, and is an int, not null, and is unique. QuestionID is a foreign key to the Question table, and is also an int, not null, and is unique.  Both AssignmentID and QuestionID act as the primary keys.
4. For the Course Table, CourseID must be an int, which automatically increments by 1, that is the primary key.  Name is a varchar that cannot exceed 50 characters and cannot be null.
5. For the GBUser (Gradebook User) table, UserID must be an int, which automatically increments by 1, that is a primary key.  Name is a varchar that cannot exceed 30 characters and cannot be null.
6. For the Professor table, ProfessorID is a foreign key to the GBUser table, a unique int, and the primary key.  Department is a varchar that cannot exceed 3 characters (each department will be recognized using an abbreviation) and cannot be null.
7. For the Question table, QuestionID must be an int, which automatically increments by 1, that is the primary key.  Points must be an int that is greater than or equal to 0.  Prompt is a varchar that cannot exceed 140 characters and the default value is ‘No prompt given.’  Correct answer is an int that cannot be null.
8. For the Section table, SectionID must be an int, which automatically increments by 1, that is the primary key.  CourseID is a foreign key to the Course table, a unique int, and cannot be null. ProfessorID is a foreign key to the Professor table, a unique int, and cannot be null.
9. For the Student Table, StudentID is a foreign key to the GBUser table, a unique int, and the primary key.  GradYear is a date and cannot be null.
10. For the Student Answers Question table, QuestionID is a foreign key to the Question table, and is an int, not null, and is unique. StudentID is a foreign key to the Student table, and is also an int, not null, and is unique.  Both QuestionID and StudentID act as the primary keys. Answer is an int that has a default value of -1.
11. For the Student Has Assignments table, StudentID is a foreign key to the Student table, and is an int, not null, and is unique. AssignmentID is a foreign key to the Assignment table, and is also an int, not null, and is unique.  Both StudentID and AssignmentID act as the primary keys. Grade is a char that cannot exceed length 1 and has a default value of ‘N’.  Score is an int and has a default value of 0.
12. For the Student Is In Section table, StudentID is a foreign key to the Student table, and is an int, not null, and is unique. SectionID is a foreign key to the Section table, and is also an int, not null, and is unique.  Both StudentID and SectionID act as the primary keys.
13. For the Student Major table, StudentID is a foreign key to the Student table, a unique int, and a primary key.  Major is a varchar that cannot exceed 3 characters and cannot be null.

# Referential Integrity Analysis

On delete, operations will cascade.  We decided on using a cascade because of the functionality and logistics of our database.  For example, a professor or student could no longer be associated with the institute, so we would like to be able to delete them from the database and deny them access. However, much of our database relies on users, so the deletion would have to be cascaded throughout the database in order to still operate correctly.

On update, operations will cascade.  However, we would ensure that if the update was not possible, a trigger would handle the issue.  We chose to cascade on update because much of our data is dependent on information from other tables, therefore, the selected table to update is not the only table that needs to be updated.

# Business Rule Integrity Analysis

This system is designed to benefit professors and engage students in the classroom and has few business rules.  An example of a business rule in our database is that a student cannot be in enrolled in multiple sections of a course.

# Invalid Data Entry

In order to guard against invalid data entry, we have placed many constraints on our tables.  However, this isn’t always enough to prevent invalid data from entering or even altering the database.  We plan to take measures to prevent SQL injections by escaping special characters in data entered in order to create legal SQL statements. To prevent Cross-Site Scripting, we will convert special characters to the correct HTML equivalent. Another way to handle invalid data is to ensure that the interface does not include any string concatenation, especially where a user is entering data.