# Introduction

This assignments discusses the threats that can occur to a relational database, specifically the security issues with transactions, concurrency, backups and SQL. After addressing the common issues, a database will be designed and implemented that shows a higher level of security. This will include project features such as an Entity Relationship Diagram and a Data Dictionary. To give an idea of the general usage of the database, this assignment will also feature queries that are relevant with a brief explanation on how SQL attacks works. Finally, the project concludes with an evaluation on the Database and what security features can be used to better the database.

# Task 1 - Threats to the security of relational databases

A database can be considered as a form of expressing information. This is usually done by tables with rows and columns. What makes the database relational is the fact that “these tables can be joined together to show links or “relationships” between the information stored in them. (Google Cloud, 2023). Because anything can be stored in a database it is important to keep that data secure. Like other methods of data storage, relational databases are susceptible to cyber security risks.

## Transactions

Any action that is performed in a database is called a transaction. A real-world example could be performing an account transaction in a bank account. Withdrawing the money from your account would be seen as a transaction. For a transaction to occur in a database, it must follow a set of operations. These include:

Atomic – for a transaction to be atomic, it must be completely classed as one individual unit. “Every change that transaction makes to the database happens or none of them do.” *(fauna, 2021).* This means if 1 part of the transaction encounters an error then the whole transaction is cancelled in what is called a “Rollback”.

Consistency – when a transaction occurs, the previous and new databases should be in their consistent states, meaning all the data should be accurate. If there were any inconsistencies during the transaction, then there would also be a rollback.

Isolation – when there is more than 1 transaction going on at the same time, they all need to be isolated when being executed. This means that the system will act as a queue, only letting one transaction occur at a time. If multiple transactions are going on in the system, there is an increased chance of data loss in the event of a crash or power cut.

Durability - after the transaction has been completed, any change that is made is permanent and will not be reverted if the system fails, they should also be stored in a durable location like a hard drive, so they are secure from software error and failures.

## Concurrency

Data that is being retrieved from a database is vulnerable when it is being pulled from more than one user at a time. This is where concurrency comes in. each transaction that is ran concurrently “must be isolated to allow no mix ups” in the database. *(solarwinds, 2023)* Here is an example:

Let us say 2 people have access to a joint bank account and there is a number for the starting balance, if they both spend money from the account and those transactions are ran concurrently but not isolated then there can be errors in the final balance, causing a rollback. Concurrency would ensure that both the transactions are completed at the same time however the calculations for the balances are separately accounted for.

There can be many risks for not having concurrency in a relational database. Some of them are:

Performance issues – if 2 or more transactions are going on at the same time then they are both fighting for resources from the system. This can cause a crash which can result in a loss of data.

Deadlocks – if the above situation occurs then there is a chance that there will be a deadlock in the system. This is when both transactions are waiting for each other to be completed so neither of them can progress causing a standstill in the database.

## Database Backups

If one of the previously mentioned errors happens in a database, all the data is likely to be deleted. This is where backups come in. Backups are an essential part of many computing industries that carry any sort of information. They are an exact copy of the current data in a system which can be useful for many reasons such as:

Data Recovery - if an error such as a concurrency failure causes data to be wiped from the system, the backup will be used to restore any lost data. Preventing hardware failures, software errors, human error, or malicious attacks.

Disaster Recovery – sometimes, data damage is out of the user's control. “Circumstances like natural disasters can cause the hardware to be destroyed.” *(IBM, 2021).* A backup on a new drive would be useful here to prevent further damage from this.

## SQL

Structured Query Language (SQL) is a programming language used to “edit data stored in a relational database.” *(w3schools, 2023).*  It can be used to perform tasks such as reading, updating, and deleting data. Just like any programming language, it can also be used to attack databases. This is done through an SQL injection attack. A hacker would exploit a vulnerability in the database to trick the system into executing malicious code to gain access to data. SQL can also be used in other ways such as Denial of service attacks (DoS) and Data Manipulation.

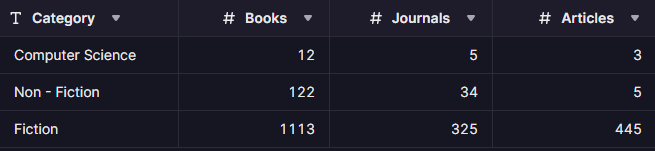
# Task 2 Design a secure database containing at least 5 tables

Following the scenario, a secure database has been created which mimics a Library management system. The database tool that will be used is Xata.

The first table is in its 1st normal form. This is because there are no repeated entries, and the ID column is a primary key which can be used to directly gain information from the other columns about the book

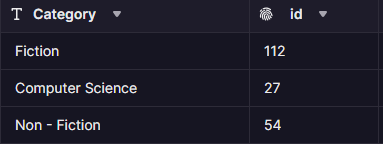
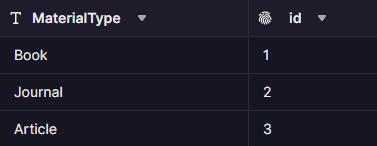
A screenshot of a computer

Description automatically generated*Figure 1, Image of borrowing management page on system*

*Figure 2, Image of Reading materials stock per category*

In the 2nd image, the table is in its 2nd normal form. This is because it has already been put in its 1NF however the information in the table does not rely on the primary key from the 1st table. The Book ID cannot give us information about the availability of reading material in each category.

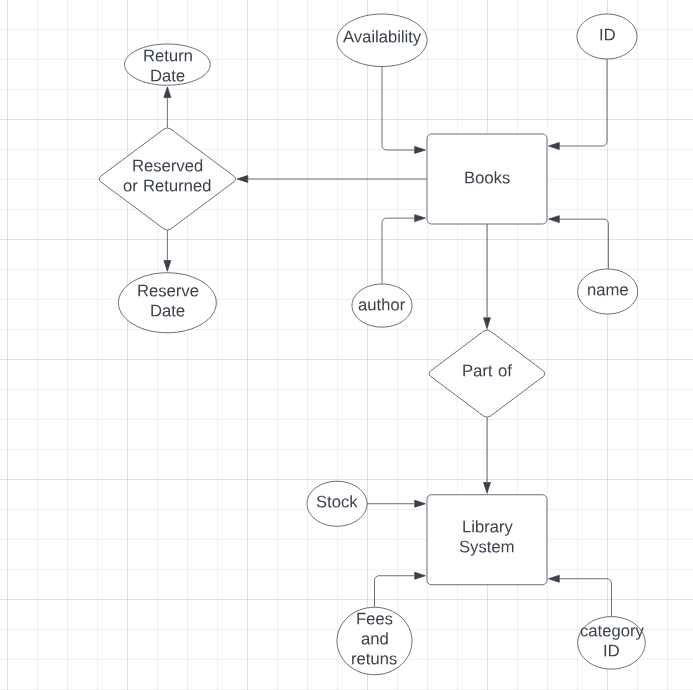
To put the table in its 3rd normal form, all we must do is keep it in 2nf and remove any redundancies. We do this by separating the columns into tables even further



*Figures 3 and 4, category and material type column separated into their own tables*

## Entity Relationship Diagram

An Entity Relationship Diagram’s job is self-explanatory. It shows how each item or ‘entity’ is related or linked and what characteristics it has. Here is the one for the library.



The ERD explains how the book which have their own characteristics are part of the entire system and how they be returned and reserved.

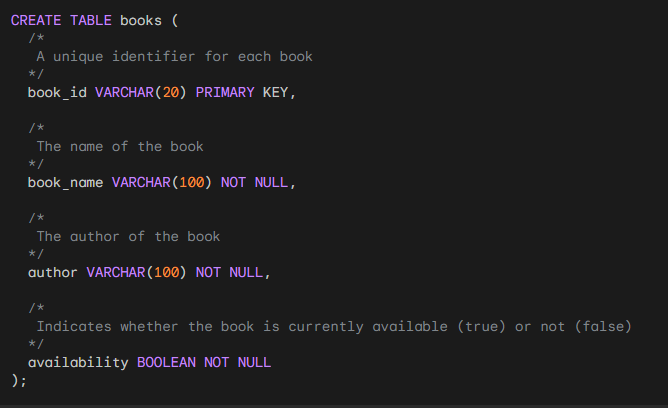
## Data Dictionary

A data dictionary acts like a huge table of every asset that will be used in a project. They are quite simple to create and help managers understand how each entity will be used.

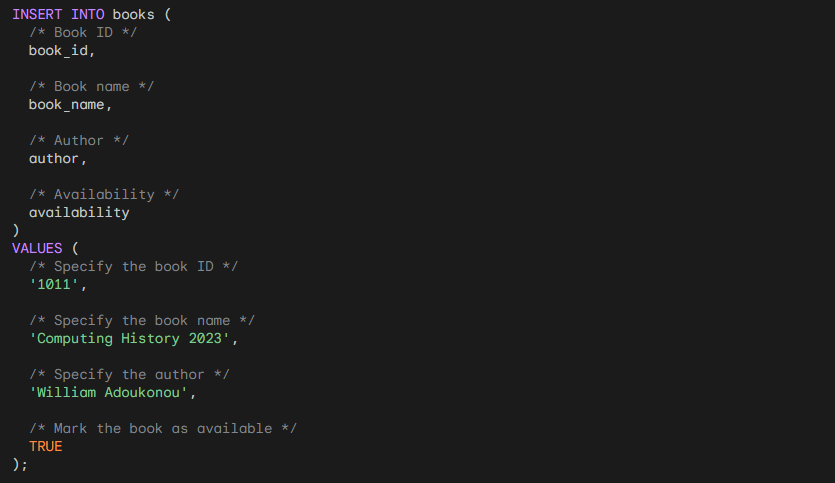
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Data type | Field Size | Source | Input method | Validation check |
| bookname | String | max length = 50 characters | system | Auto - assign | Yes |
| bookID | String | max length = 4 characters | System | Auto – assign | yes |
| author | String | Length = 50 characters | System | Auto-assign | yes |
| publishYear | Integer | Max 4 characters | System | Auto-assign | Yes |
| TotalPrice | Float | Total of book | System | Auto – assign | Yes |
| CreateBookName | String | max length = 50 characters | User | User | Yes |
| CreateBookID | String | max length = 4 characters | User | User | yes |
| CreateAuthor | String | Length = 50 characters | User | User | yes |
| CreatePublishYear | Integer | Max 4 characters | User | User | Yes |

## Implementation

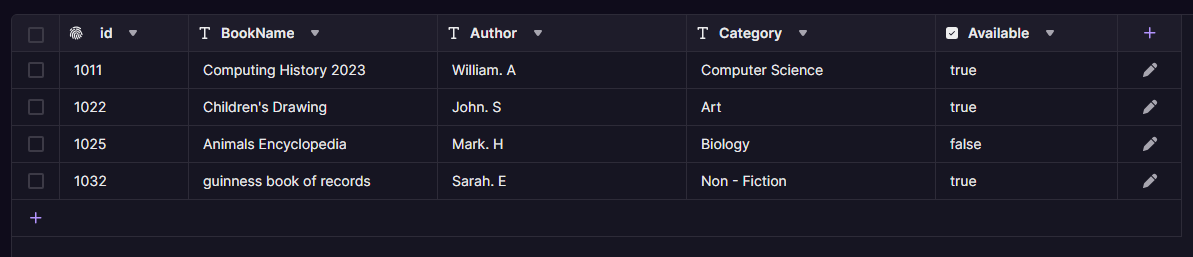
This Code is in PostgreSQL. The code will create a table called books with these records.



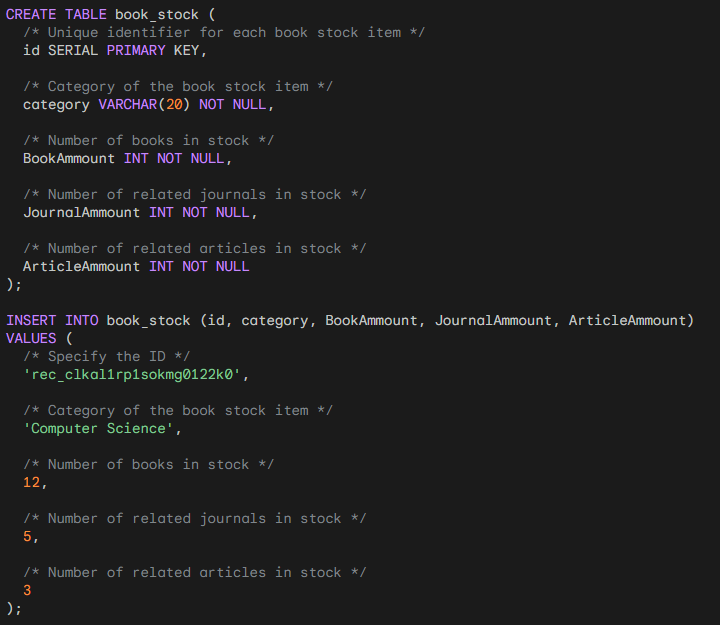
To add information into the table, we can use this code.



The database will now look like this with more added inputs

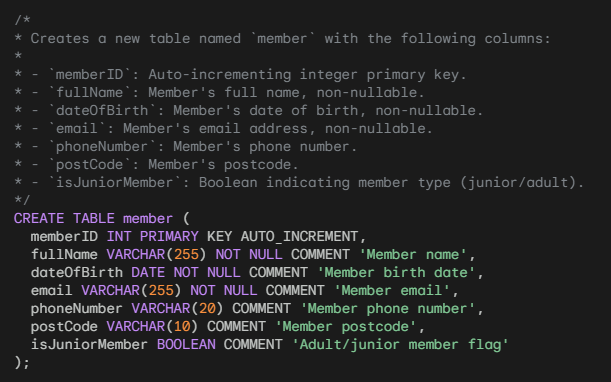


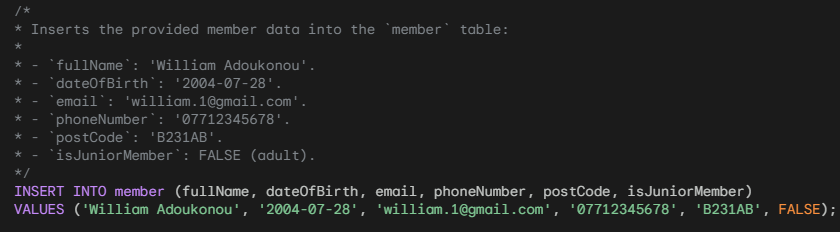
**Table 2**

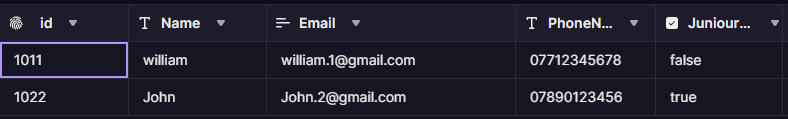




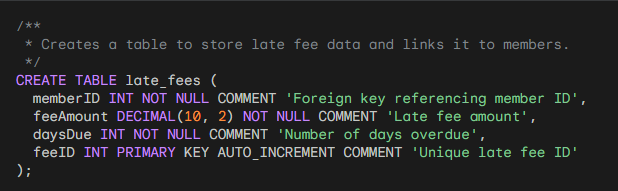
**Table 3**

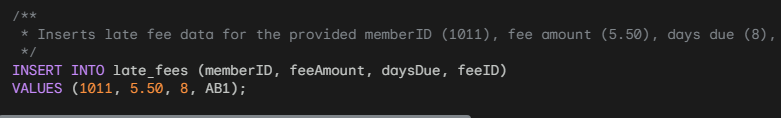


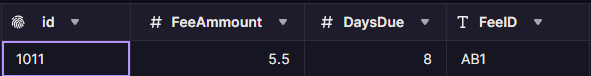




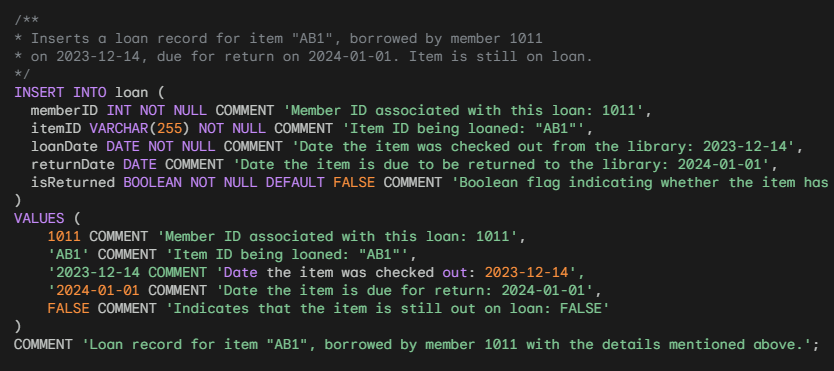
**Table 4**

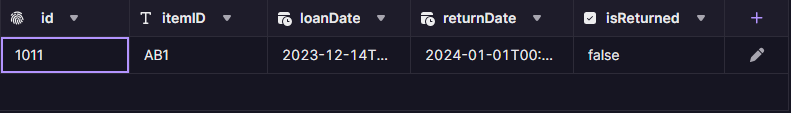






**Table 5**





# Task 3 Create meaningful queries and Explain SQL injection attacks

A query is a request for information from the database. It acts as a line of instructions that ask for specific information, adds conditions to it and explains how it should be presentation. Some queries for the database could be:

1 – Query for getting members with overdue borrow

SELECT \*

FROM member

JOIN loan ON member.memberID = loan.memberID

WHERE returnDate < CURDATE() AND isReturned = FALSE;

2 – Query for most borrowed item

SELECT itemID, COUNT(itemID) AS numLoans

FROM loan

GROUP BY itemID

ORDER BY numLoans DESC

LIMIT 1;

3 – Query for finding members with overdue fees

SELECT memberID, SUM(feeAmount) AS totalFees

FROM late\_fees

GROUP BY memberID

HAVING SUM(feeAmount) > 0;

4 – Query for finding total made from all late fees

SELECT SUM(feeAmount) AS totalFeeRevenue

FROM late\_fees;

5 – Query for finding books with the more than 10 copies

SELECT itemID, COUNT(\*) AS numCopies

FROM item

GROUP BY itemID

HAVING COUNT(\*) > 10;

6 – Query for seeing inactive members

SELECT memberID, COUNT(\*) AS numLoans

FROM loan

WHERE isReturned = FALSE

GROUP BY memberID;

7 – Query for oldest loan date

SELECT loanDate, MIN(loanDate) AS oldestLoan,

FROM loan;

8 – Query for most recent loan date

SELECT loanDate, MAX(loanDate) AS latestLoan

FROM loan;

9 – Query for which months the loans are the highest

SELECT MONTH(loanDate) AS month, COUNT(\*) AS numLoanRequests

FROM loan

GROUP BY MONTH(loanDate)

ORDER BY numLoanRequests DESC

LIMIT 10;

10 – Query for which items have been overdue for the longest

SELECT itemID, loanDate, DATEDIFF(CURDATE(), returnDate) AS overdueDays

FROM loan

WHERE returnDate < CURDATE() AND isReturned = FALSE

ORDER BY overdueDays DESC

LIMIT 10;

## How SQL Attacks Works

As mentioned prior. SQL attacks work by the user being able to implement/inject malicious code into a poorly secured database. The database would then return whichever data the hacker wants. The several types of SQL attacks are:

Blind SQL Attacks – This is when the hacker injects SQL code that does not gain direct information from the attack. It closely watches the timing or behavior of the application's responses to gain the info.

Error Based Attacks – Like any other database, if an error occurs then the user is presented with an error code. The hacker can use this to inject code that “forces these error codes which leak information about how the Database is structured.” *(avast, 2023)*

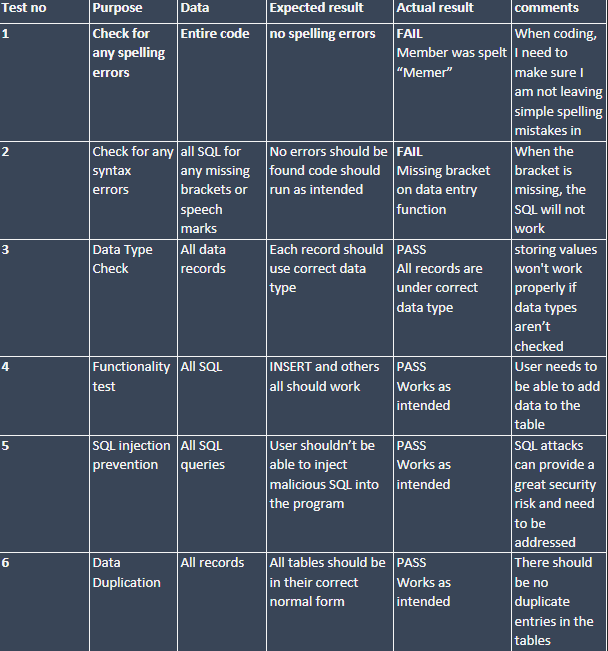
In-band SQL Attack – this type of attack is the most used. The code that the hacker injects directly returns the wanted information. Here is an example query:

SELECT \* FROM member WHERE memberID = 1 OR '1=1';

Since the OR '1=1' clause always evaluates to true, the query would always return the records from the member table. Giving hackers unauthorised access to every member’s information.

# Task 4 Test Plan

A test plan helps find and mitigate risks associated with database’s development by highlighting potential problems that could come up later. This can help to prevent expensive delays and problems after the database is released.



# Task 5 - Produce an evaluation of the finished product

Looking back at the client's requirements, they wanted:

-A library system that will be useful for members as well as a librarian

-To keep the track of book availability and store the membership info

-To see the number of books issued/returned

-To see the late fine charge record etc

These conditions were met well in the designed system. The system offers admin features for the librarians such as the BookStock table, as well as insight into account settings which may be useful for users through the MembersTable.

The Client also wanted to be able to keep track of book availability and store membership information. Tables 2 and 3 both do this. Through queries, the admin can also insert membership information into the table. It will also automatically update since the junior member Boolean is only true if the DateTime condition is under 18 for a member's date of birth.

Another one of the client’s requests was to include the books return/issue status. This was included in the LoanTable which uses a primary key based on the MemberID. Not only does this tell the admin if the book is borrowed, but it also tells them who is currently using it meaning they can award fines appropriately. In the future, it would be simpler to add Returned/Available as their own records to the MaterialType table. This way the database would keep track of the total amount of books borrowed and returned.

The library also requested a feature to see the charge record for late fines. This was implemented through the LateFee table. This was successful as the database assigns an ID to each fee as well as inform the admin of how long it's been up for. To make this even better, it could be implemented into the Member Table to inform users of how long their fees have been due for.

## Security Improvements

With SQL, Input Validation and Parameterization are some “simple methods to prevent attacks.” *(sqlshack, 2022).* These methods aren't 100% secure though. Another way to boost the security of this system would be to add Role Based Access Control or RBAC. For it to work, the admin needs to define different roles within the database and assign permissions to each role. Access control is “the principle of making sure that users have as little permissions as possible.” *(digitalguardian, 2023)* Giving someone more permissions than needed means they can always have sensitive information for misuse at any time. The administrator would then use access control lists to manage which roles have access to specific tables, columns, and data. The admin would get full control, and lower employees would have restricted access.

*End of assignment*

*References*

Belcic, Ivan . “SQL Injection: What Is It, How Does It Work, and How to Stay Safe?” *SQL Injection: What Is It, How Does It Work, and How to Stay Safe?*, 11 Oct. 2023, [www.avast.com/c-sql-injection#:~:text=This%20is%20how%20SQL%20injections](#:~:text=This%20is%20how%20SQL%20injections). Accessed 6 Jan. 2024.

Erkec, Esat. “SQL Server Database Parameterization Option and Its Query Performance Effects.” *SQL Shack - Articles about Database Auditing, Server Performance, Data Recovery, and More*, 18 Feb. 2022, [www.sqlshack.com/sql-server-database-parameterization-option-and-its-query-performance-effects/#:~:text=The%20SQL%20Server%20query%20optimizer%20may%20decide%20to%20parameterize%20some](#:~:text=The%20SQL%20Server%20query%20optimizer%20may%20decide%20to%20parameterize%20some). Accessed 6 Jan. 2024.

“Types of Database Backups.” *Www.ibm.com*, 13 Aug. 2014, www.ibm.com/docs/en/tsmfm/7.1.1?topic=processing-types-database-backups.

w3schools. “SQL Introduction.” *W3schools.com*, 2019, www.w3schools.com/sql/sql\_intro.asp.

“What Is a Database Transaction?” *Fauna*, 2 Aug. 2021, fauna.com/blog/database-transaction.

“What Is a Relational Database.” *Google Cloud*, cloud.google.com/learn/what-is-a-relational-database#:~:text=A%20relational%20database%20(RDB)%20is.

“What Is Database Concurrency? - IT Glossary | SolarWinds.” *Www.solarwinds.com*, www.solarwinds.com/resources/it-glossary/database-concurrency.

Zhang, Ellen. “What Is Role-Based Access Control (RBAC)? Examples, Benefits, and More.” *Digital Guardian*, 7 Nov. 2022, www.digitalguardian.com/blog/what-role-based-access-control-rbac-examples-benefits-and-more.

Figure 2, Code for adding an entry to the enrolment table

A screenshot of a computer

Description automatically generated

Figure 3, image of table for student scores

A screenshot of a computer

Description automatically generated

Figure 4, code which updates all scores to a certain value

A screenshot of a computer

Description automatically generated

Figure 5, code which updates the score when it is less than a certain value

A screen shot of a computer

Description automatically generated

Figure 6, SQL code to add a table to a database using PGAdmin

A screenshot of a computer

Description automatically generated