**DATABASE SECURITY**

Table of Contents

[**1.** **Introduction** 2](#_Toc197432230)

[**1.1.** **Data Dictionary** 2](#_Toc197432231)

[**1.1.1.** **Database Tables** 2](#_Toc197432234)

[**1.1.2.** **Database Views** 3](#_Toc197432238)

[**1.1.3.** **Stored Procedures and Security Policies** 4](#_Toc197432239)

[**1.1.4.** **Triggers** 4](#_Toc197432240)

[**2.** **Permission Management** 4](#_Toc197432241)

[**2.1.** **Authorization Matrix** 4](#_Toc197432242)

[**2.2.** **User Management Solutions** 6](#_Toc197432243)

[**2.2.1.** **Access Control** 6](#_Toc197432244)

[**2.2.2.** **Row Level Security Policy** 9](#_Toc197432245)

[**2.2.3.** **View** 11](#_Toc197432246)

[**2.2.4.** **Store Procedures** 15](#_Toc197432247)

[**3.** **Data Protection** 23](#_Toc197432248)

[**3.1.** **Data Classification Matrix** 23](#_Toc197432249)

[**3.2.** **Data Protection Solutions** 28](#_Toc197432250)

[**3.2.1.** **Data Encryption** 28](#_Toc197432251)

[**3.2.2.** **Backup and Recovery** 32](#_Toc197432252)

[**4.** **Auditing** 35](#_Toc197432253)

[**4.1.** **Audit Matrix** 35](#_Toc197432254)

[**4.2.** **Auditing Data Modifications** 37](#_Toc197432255)

[**4.3.** **Auditing Structural Modifications** 39](#_Toc197432256)

[**4.4.** **Auditing Permissions Modifications** 41](#_Toc197432257)

[**4.5.** **Auditing Login and Logout** 43](#_Toc197432258)

[**5.** **Summary** 46](#_Toc197432259)

[**6.** **References** 47](#_Toc197432260)

1. **Introduction**

The purpose of this project is to strengthen the safety of the database that is part of the Academic Information System (AIS). In spite of the fact that the institution relies on this system to handle its day-to-day operations, the database was discovered with significant security flaws in the design and implementation of the database.

The establishment of comprehensive auditing processes to monitor all database activity and user logins, the implementation of automatic and regular backups to minimize the amount of data that is lost, and the guaranteeing of data categorization and protection are key areas of attention. It is necessary for the security model to contain role-based access management, which must carefully follow the principle of least privilege, in order to effectively handle various user roles, such as Database Administrators, Students, and Lecturers, each of which has a unique set of authority and restrictions.

* 1. **Data Dictionary**

1. 1. 1. **Database Tables**

1. Student

* ID (varchar(6)): Primary key. Unique identifier for each student.
* SystemPwd (varbinary(max)): Encrypted password for the student’s system access.
* Name (varchar(100)): Full name of the student.
* Phone (varchar(20)): Contact phone number of the student.

2. Lecturer

* ID (varchar(6)): Primary key. Unique identifier for each lecturer.
* SystemPwd (varbinary(max)): Encrypted password for the lecturer’s system access.
* Name (varchar(100)): Full name of the lecturer.
* Phone (varchar(20)): Contact phone number of the lecturer.
* Department (varchar(30)): Department to which the lecturer belongs.

3. Subject

* Code (varchar(7)): Primary key. Unique code for each subject.
* Title (varchar(40)): Title or name of the subject.

4. Result

* ID (int): Primary key. Auto-incrementing identifier for each result entry.
* StudentID (varchar(6)): Foreign key referencing Student(ID).
* LecturerID (varchar(6)): Foreign key referencing Lecturer(ID).
* SubjectCode (varchar(7)): Foreign key referencing Subject(Code).
* AssessmentDate (date): Date when the assessment was conducted.
* Grade (varchar(2)): Grade awarded for the subject.
* CreatedBy (varchar(6)): ID of the user who created this record.
* Department (varchar(30)): Department associated with the result.

5. AuditLog

* AuditLogID (int): Primary key. Auto-incrementing identifier for each audit log entry.
* EventType (nvarchar(100)): Type of event (e.g., INSERT, UPDATE, DELETE).
* EventData (xml): Detailed XML data describing the event.
* EventDateTime (datetime): Date and time when the event occurred.
* UserName (nvarchar(128)): Username of the user who triggered the event.
* SchemaName (nvarchar(128)): Schema name where the event occurred.
* ObjectName (nvarchar(128)): Name of the database object affected.
* SqlStatement (nvarchar(max)): SQL statement executed that triggered the log entry.

6. LoginHistory

* ID (int): Primary key. Auto-incrementing identifier for each login attempt.
* UserID (nvarchar(100)): Identifier of the user who logged in.
* LoginTime (datetime): Timestamp of the login.
* LogoutTime (datetime): Timestamp of the logout.
* Succeeded (bit): Indicates whether the login attempt was successful (1) or not (0).


  3. 1. **Database Views**

1. StudentInfo

* Provides a view of students’ IDs, names, and phone numbers, excluding sensitive data like passwords.

2. LecturerInfo

* Provides a view of lecturers’ IDs, names, phone numbers, and departments, excluding sensitive data like passwords.

3. StudentAcademicData

* Provides a view for individual students to access their own academic records.

4. AllStudentsInfo

* View accessible by lecturers to view basic information of all students.

5. DepartmentResults

* View showing results entered by lecturers from the same department, including detailed result information.
  + 1. **Stored Procedures and Security Policies**
* AddNewStudent: Adds a new student record with encrypted password.
* AddNewLecturer: Adds a new lecturer record with encrypted password.
* UpdateStudentDetails: Allows students to update their own information securely.
* UpdateLecturerDetails: Allows lecturers to update their own information securely.
  + 1. **Triggers**
* AuditDataChanges: Triggers that capture and log changes in Student, Lecturer, Subject, and Result tables.

1. **Permission Management**
   1. **Authorization Matrix**

|  |  |  |  |
| --- | --- | --- | --- |
| **Role** | **Permission Type** | **Object** | **Privilege** |
| DB Admins | Grant | Database: AIS | Create |
| DB Admins | Grant | Security: Users | Create, Alter |
| DB Admins | Grant | Tables: Student, Lecturer | Select, Insert, Update |
| DB Admins | Deny | Columns: Student.SystemPwd, Lecturer.SystemPwd | Select, Insert, Update, Delete |
| DB Admins | Deny | Tables: Result | Select, Insert, Update, Delete |
| Students | Grant | Tables: Student (Own) | Select, Update |
| Students | Grant | Tables: Result | Select |
| Students | Deny | Tables: Result (Other Students) | Select, Insert, Update, Delete |
| Lecturers | Grant | Tables: Lecturer (Own) | Select, Update |
| Lecturers | Grant | Tables: Student | Select |
| Lecturers | Grant | Tables: Result (They added) | Update, Delete |
| Lecturers | Grant | Tables: Result (Same Department) | Select |
| Lecturers | Grant | Tables: Result (Subject and Students they teach) | Insert, Select, Update, Delete |
| Lecturers | Deny | Column: Student.SystemPwd | Select |

The authorization matrix is an indispensable security part of the database security strategy for the Academic Information System (AIS). It ensures that permissions are given in accordance with user roles, and therefore, the security and confidentiality of the database is maintained. The matrix is set up in compliance with the least privilege principle. This implies giving DB admins, students, and lecturers the minimum access required to do their job. Such as providing the DB Admins with the permission to add and manage the student and lecturer details but are not allowed to delete them. In the provided authorization matrix, DB Admins are given access to create new table, view, or procedure and they are responsible for the creation and the management of user roles. Restrictions are however limited to only areas where it necessary, such as not granting them the ability to view or change sensitive information which include password and academic data like results. Students are granted privileged access to their own data, but they are not allowed to view and edit other students' details. It allows for privacy, making sure that forgery and unauthorized changes are avoided. Lecturers are granted the right to control the data related to their lectures, but they are unable to access or manipulate the student security passwords, preserving both the data integrity and system security.

* 1. **User Management Solutions**
     1. **Access Control**

Access control is one of the basic security features of databases and requires users to be granted access to data adequate to their roles. In the scope of Academic Information System (AIS), a powerful access control model has been implemented, built upon Role-Based Access Control (RBAC) and additionally strengthened with Row-Level Security (RLS). This combination was selected to ensure a holistic and multi-layered approach to security which includes both prevention of specific threats and application of the principle of least privilege.

RBAC was one of the methods that was used because it precisely follows the organizational structure of the universities environment, evident from the fact that people take on distinctive roles such as students, lecturers, and database administrators. Instead of just focusing on individuals and providing them with specific roles and permissions, AIS will create roles within the organization such as DB Admins, Students, and Lecturers with the respective permissions assigned to them thus retaining manageability and scalability.

The RBAC model is also chosen for its security and efficiency reasons in allowing users to alter permission management processes (SecurePass, 2021). In an academic setting, when roles may change such as students may graduate and lecturers may take administrative positions, a RBAC allows for instant and effortless adjustments, reflecting such changes without the need of lengthy re-configuration processes.

A white background with black text

Description automatically generated

The first step in establishing RBAC in this database is to create different roles within the database. Each of these roles is then assigned to different permission to allow them to perform their responsibilities. The ‘CREATE ROLE’ statement will define a role for the database which in this case is DBAdmins, Students, and Lecturers. Their specific permission will be discussed later.

**DBAdmins**

A screenshot of a computer program

Description automatically generated

Figure above shows the permission assigned to DB Admins with ‘*GRANT*’, and ‘*DENY*’. First, the ‘*CREATE SCHEMA*’ creates a schema in the database follow by granting DB Admins to modify the schema structure such as creating or altering the objects within the schema. Next, DB Admins can alter any user and role like Students and Lecturers indicates that DB Admins are responsible for managing access control by creating new roles or altering existing roles and users. This is essential for performing user onboarding and altering roles to fit the current access requirements. The ‘*GRANT ALTER ANY LOGIN TO DBAdminLogin*’ permission allows DB Admins to create or modify logins for other roles. Since logging into the database required username and password and DB Admins are not allowed to add password for Students and Lecturers in their table. Therefore, DB Admins will create a temporary password for them and require them to change on their first login and update their details accordingly. Furthermore, DB Admins is also granted permission to access the view created to limit the information that they can view from the students and lecturers’ table which will be discussed later this part. Then, direct access to the base table will be denied for DB Admins. DB Admins are also granted the permission to execute the stored procedure created like ‘*AddNewStudent*’, ‘*AddNewLecturer*’, and ‘*GenerateLoginLogoutReport*’ which will ensure consistency and security and will be discussed later in this report. The DB Admins are also provided with the ability to generate reports on user login and logout activities and the changes made to the database including structural, data, and permissions.

**Students**

A screenshot of a computer screen

Description automatically generated

The figure above shows the students’ permission. First, students are granted to view their personal details in the ‘*Student*’ table. This access is further controlled by a row level security which will be discussed later to ensure they can only see their own details but not others. With ‘*GRANT SELECT ON dbo.StudentAcademicData TO Students;*’, students are granted read access to the ‘*StudentAcademicData*’ view. This view is designed to display academic information that is directly related to the student who is logged in. The view ensures that students have access to their academic records without exposing the data of other students, adhering to privacy requirements. The execution permission on ‘*dbo.UpdateStudentDetails*’ granted by ‘*GRANT EXECUTE ON dbo.UpdateStudentDetails TO Students;*’ empowers students to update their own details. It encapsulates the update operation within a stored procedure, which not only streamlines the process but also adds a layer of security by ensuring that the update operation is performed in a controlled manner, preventing any unauthorized updates to student records. The students are also granted permission to record their login and logout activities by executing the stored procedure created. This feature is necessary for functioning with the purpose of keeping safe records of user activities within the system.

**Lecturers**

A screenshot of a computer program

Description automatically generated

The figure above shows the permission management for Lecturers. Similarly to student permission, lecturers are granted to view their personal details in the ‘*Lecturer*’ table. This is also further controlled by a row level security policy to avoid them viewing other lecturers’ details. The lecturers are then granted the permission to have full control on the ‘*Subject*’ table, allowing them to add new subject or modify the existing one. A view is then created specifically for lecturers to view all students’ non sensitive information and to view only the result from the same department. Lecturers are then granted execute permission on the stored procedure such as ‘*AddResult*’, ‘*UpdateResult*’, ‘*DeleteResult*’, ‘*RecordLogin*’, and ‘*RecordLogout*’ stored procedure. Each of the procedures has its own security measures implemented such as in ‘*AddResult*’, lecturers are only allowed to enter for the subject and student they taught. Whereas in ‘*UpdateResult*’ and ‘*DeleteResult*’, lecturers are only allowed to perform this action for the result that they added.

* + 1. **Row Level Security Policy**

Row-Level Security (RLS) is a model which prevents access to records in any particular table at the row level of the database (Berning, 2023). Unlike traditional access control mechanisms at the table level or the column level, RLS allows you to apply restrictions directly on data access from any table in the database through which the data is accessed. It implies that RLS can implement a uniform access strategy whenever different applications are fetching the same data. In AIS, the rule of row level security is enforced by security functions and policies that can be altered so that only certain rows are accessible. These functions are user-based instances of a query that restrict data visibility in query results to only the requested information. An instance can be when students and lecturers can only access their own personal data.

RLS is a layer above RBAC, providing another security control that complements database data security. The system is therefore capable of restricting data access at the row-level within the database. This was the priority of the AIS project, which there was no doubt deterred students and lecturers from accessing information and data that did not belong to them. Via RLS, this can resolve the issue of authorized access attempts to confidential records. Besides, RLS provides strong security in privacy data, being a precondition in a learning environment where data about both students and assistant professors should be protected. Through granting data access only at the lowest level, RLS reduces the chance of unprotected data leakage, which, in turn, improves data privacy (Zahid, 2023). The next important consideration is that universities are subject to highly regulated standards that require the confidentiality of student information. RLS plays a role in proper compliance with these regulations by directing access restrictions to the database and ensuring that exposure of data is limited to the authorized users as well as thus significantly reducing possibilities of compliance violations. Moreover, the principle of least privilege is a key aspect in the security policy of AIS. Implementing RLS helps to block the chance for users of acquiring data that is not relevant to their educational or administrative purposes thus reducing the attack surface and the possibility for data misuse (Zahid, 2023).

A screen shot of a computer code

Description automatically generated

Figure above shows the RLS policy statement implemented to the database. The function ‘*dbo.fn\_securitypredicate\_Student*’ is a security predicate that specifies the access conditions. This enabled displaying a row only if the student ID in the table matches the login ID. This makes sure that the students see their own data only, and not for others, so that the confidentiality and safety of the personal details of the students are preserved. Moreover, certain privileged roles such as the system administrators (sysadmin), database administrators (DBAdmins), and lecturers (Lecturers) are allowed to bypass this restriction. This design creates an environment whereby the required functions can easily be managed without altering security principles. This approach is also utilized in the lecturers to ensure that they only access their personal information and cannot access other lecturers.

* + 1. **View**

Views in database act very critical with user management by offering a wide range of abstraction and security shield. In the AIS, users are provided with the view capabilities to access and perform actions with only the data that is relevant for their role. Views act like a virtual table and are a sort of window through which a database can show only certain tables which have been carefully chosen (Pavel, 2022). For instance, the view can be created that shows only student names and contact details, which the lecturer can use to access the details of the student without having access to their passwords.

The rationale for employing views for managing the data a user can access within the AIS was based on several key considerations. First, views do such a job by adding a level of abstraction that improves security (TutorChase, 2023). By including only columns that users need to perform their tasks, like ID, Name, and Phone for students, and Department for lecturers, the system ensures that no confidential data is displayed accidentally. This is very important because the privacy of data ownership is not only an ethical issue but a legal standard in most academic institutions.

Additionally, views reduce the complexity of the DB interface (Team Post, 2022). Users, especially those with no digital skills and experience, can now employ data easily. For example, the view dbo.StudentAcademicData allows students to obtain their grades and academic history in a simple manner, without getting into the complexity of the underlying database´s schema.

In addition to that, the justification for choosing views is also related to keeping data integrity and lessening the complexity of the administrative effort. Interactions of users with the AIS will be channeled through the views, which will reduce the access to the base tables. This minimizes the possible risks that could occur during accidental or unauthorized data manipulation. It also provides a method for data centralized access control which lessens the burden of administration procedure as changes to users’ permissions can be made by adjusting view definitions instead of modifying permissions across many tables.

A screen shot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

The creation of the ‘*dbo.StudentInfo*’ and ‘*dbo.LecturerInfo*’ views in the AIS database exemplifies a streamlined approach to data presentation and access control. These views provide a secure and simplified interface to the underlying tables by selecting only the non-sensitive columns such as ID, Name, and Phone for students, with the addition of Department for lecturers. By intentionally excluding the ‘*SystemPwd*’ column, the views prevent the exposure of password data, enhancing the security of personal information stored in the database.

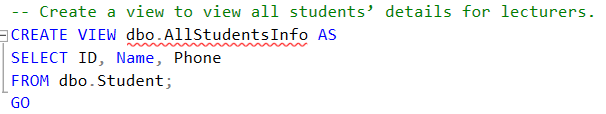
A computer code with text

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated

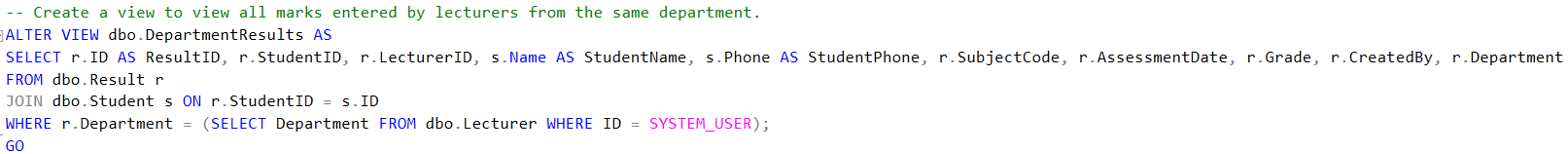
Figure above shows the view created for students to access only their personal details without being exposed to the data belonging to the others. This view forms a virtual table and will select the ID, Name, Phone from student table and Subject Code, Assessment Date, Grade from result table. The two tables are linked by a student's ID that ensures that the report of academic results always matches the correct student. Privacy protection is achieved using the ‘*WHERE*’ clause, which filters the data according to the current system user's login ID, which is obtained by the ‘*SYSTEM\_USER*’ function. What it means more specifically is that whenever a student signs in and goes to that view, they will see only their data because the view limits the dataset to the individual’s records based on their secret login ID.



A screenshot of a computer

Description automatically generated

This view is configured to provide lecturers with access to certain details about all students. It specifically selects the ID, Name, and Phone columns from the ‘*dbo.Student*’ table. The view excludes any sensitive columns that may be present in the student table, such as the Password column.



A screenshot of a computer

Description automatically generated

This view's purpose is to consolidate and display academic results entered by lecturers, filtered to include only those pertaining to lecturers' own department. The view achieves this by selecting a range of columns: the result ID, student ID, lecturer ID, student name, student phone, subject code, assessment date, grade, creator of the result, and the department from the Result and Student tables. These tables are joined on the student ID to match students with their academic results.

Crucially, the view is tailored dynamically for each lecturer using it. When a lecturer queries the view, the results are filtered so that only the entries related to their department are returned. This is managed by a ‘*WHERE*’ clause that matches the ‘*Department*’ column from the Result table with the lecturer's department, which is obtained through a nested ‘*SELECT*’ statement. This subquery determines the department of the currently logged-in user, fetched from the Lecturer table based on the system user's ID.

* + 1. **Store Procedures**

This process is a critical part of the AIS. This procedure covers primary database operations into predefined SQL statements that are run with more speed and a higher level of security than the standalone queries (Uikey, 2023). The selection of stored procedure in AIS is because of its performance, stored procedures are pre-compiled and optimized by the database system resulting in an efficient execution of queries when compared to dynamically generated SQL statements (Ravikiran, 2021). This is a most valued advantage in an educational system which may require an additional load of operations at times of high attendance like during student registrations or grade submissions.

Maintainability is also significant among the advantages of stored procedures (geeksforgeeks, 2020). All SQL code is placed under a single umbrella of the database to simplify updates and maintenance. When radical changes are needed, it is more beneficial to rewrite a single stored procedure than editing different occurrences of in-line SQL statements placed in countless applications. This centralized method enables the process to be conducted in a uniform manner and minimized the possibility of mistakes. Besides, stored procedures allow for controlling of data, assure business rules and data validation at the database level. This is necessary to keep the AIS consistent, which means that all transactions with the database take place under the same predefined rules, independently of the application through which the database is accessed.

A screenshot of a computer code

Description automatically generated

The ‘*dbo.RecordLogin*’ procedure is very detail oriented as it records the exact time a user login to AIS. The process begins by capturing the user's login name through the ‘*ORIGINAL\_LOGIN()*’ function, which ensures that the true user account is logged, even if the context is switched within the session. It then attempts to identify whether the login name corresponds to a student or lecturer and records this information alongside the login time to ‘*LoginHistory*’. The logout procedure, though not included in the documentation for brevity, is conceptually similar to the login procedure. It captures the exact time a user logs out of the system, providing a complete picture of user session duration.

A screenshot of a computer program

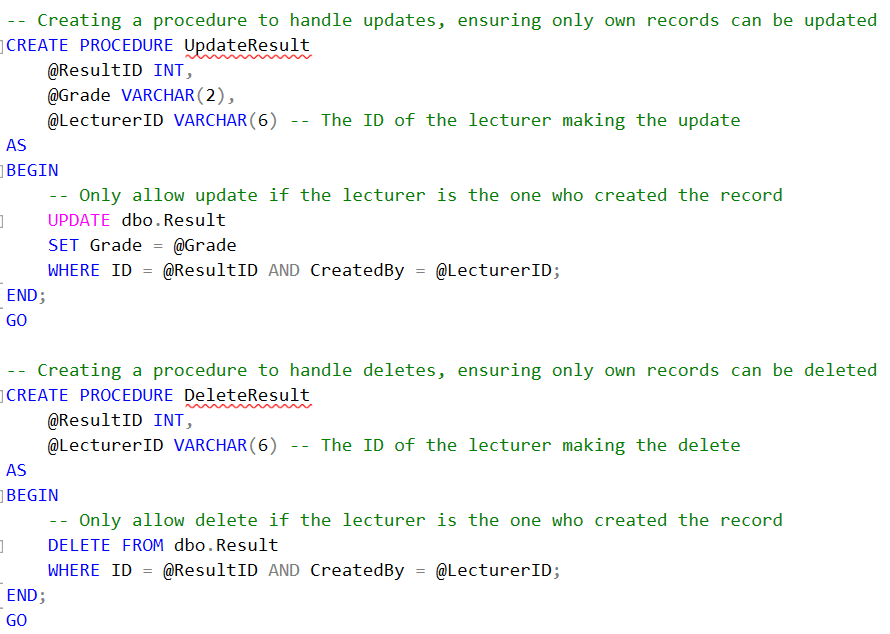
Description automatically generated

A screenshot of a login report

Description automatically generated

The figure above shows the procedure created to generate login and logout history for DB Admins. The procedure will start by verifying whether the user running the job is a member of the 'DBAdmins' or 'sysadmin' roles, which the users with these roles can make the report. If the role of the user was any of these roles, the procedure proceeds to query the ‘*LoginHistory*’ table. It allows the selection of all records where the ‘*LoginTime*’ column is comparable to the current date. This results in a report that contains only the login and logout activities for that day with the help of ‘*CAST(LoginTime AS DATE) = CAST(GETDATE() AS DATE)*’ statement.

As shown in the second figure, the result only shows the login and logout record of the day. The succeeded column is column indicates whether the login was successful or not, 0 means for failed login while 1 means for successful login.



The ‘*UpdateResult*’ stored procedure gives lecturers an opportunity to alter the grades of students in their courses. Through this method, a lecturer can only update the results which he/she entered, this is carried out by checking the ‘*CreatedBy*’ data field against the lecturer's ID. This function is a vital securing mechanism to ensure that the student grades are not subjected to any unauthenticated alteration, promoting the security of academic records. Similarly, the ‘DeleteResult’ procedure allows instructors to remove only the academic results they themselves created. Using the CreatedBy field as a condition in the WHERE clause within the DELETE statement is vital to avoid the lecturers from removing the records of others. This limitation is crucial because it stops unintended or intentional information loss and protects historical consistency of the student performance data.

A screenshot of a computer program

Description automatically generated

A close-up of a check list

Description automatically generated

The ‘*dbo.AddNewStudent*’ stored procedure is a secure method for database administrators (DBAdmins) to add new student records to the AIS. This procedure includes an essential security feature that is it automatically encrypts the temporary password assigned to the new user using the system's encryption key. Although it is a temporary password, the password will also be encrypted before inserting into the student table, ensuring the student password is secured. Similar approach is also use in creating new lecturer in the AIS. Both students and lecturers are required to change their password on their first login with the statement ‘*MUST\_CHANGE*’ to further secure their account.

A screenshot of a computer program

Description automatically generated

This process starts with making sure that the one who is trying to make the update is really the specific student by looking through the ‘*SYSTEM\_USER*’ and student ID. This is a mandatory security measure to keep aside the students from modifying data of others. After this validation, procedure carries out the ‘*PasswordEncryptionKey*’ operation and allows for decryption by the ‘*AISServerCert*’ certificate. This specifically provides that the modifications to any password made are performed securely using an encryption technique keeping all the confidential login details safe. Upon the completion of the update, the procedure closes the symmetric key to ensure that the encryption tools are secure. A similar approach is also utilized for lecturers to update their own details on their first login.

A screenshot of a computer program

Description automatically generated

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer

Description automatically generated

The figure above shows the procedure to allow students and lecturers to view only their own details with the password shown in plain text for them. The procedure will start by identifying the current login user and check if they exist either in the student or lecturer table. Based on that, the user will then have their details retrieved and have their login password display as plain text for them. This is also controlled by RLS policy that discussed previously so that they are only allowed to view own details.

1. **Data Protection**
   1. **Data Classification Matrix**

A Data Classification Matrix is a tool that is used for the purpose of organising and categorising various sorts of data in accordance with its significance, sensitivity, and secrecy. The matrix is a structure that resembles a grid and is used to classify data into several levels according to the amount of danger they pose. Additionally, the matrix will set security controls that correlate to each level. In most cases, data is divided into four distinct categories: public, internal, secret, and restricted. The access controls, authentication requirements, and permission constraints that are particular to each category are individual to that category. Through the use of suitable security measures that are determined by the categorization level of the data, the matrix assists organisations in effective management and protection of their sensitive data. (Szentgyorgyi-Siklosi, 2023)

In the AIS system, the data can be classified into several categories based on sensitivity and criticality as follows:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Data Item** | **Classification** | **Description** | **Protection Measures** | **Example Data** |
| Student ID | Confidential | Unique identifier for students. | Encrypted storage; Access restricted to system and authorized personnel. | 'S1001' |
| Student Password | Restricted | Authentication information for student access. | Encrypted storage and transmission; No direct access, even by database admins. | Encrypted binary |
| Student Name | Internal | Full name of the student. | Protected against unauthorized updates; viewable by students and staff. | 'John Doe' |
| Student Phone | Internal | Contact number of the student. | Access restricted to authorized personnel; logged access. | '123-456-7890' |
| Lecturer ID | Confidential | Unique identifier for lecturers. | Encrypted storage; Access restricted to system and authorized personnel. | 'L2001' |
| Lecturer Password | Restricted | Authentication information for lecturer access. | Encrypted storage and transmission; No direct access, even by database admins. | Encrypted binary |
| Lecturer Name | Internal | Full name of the lecturer. | Protected against unauthorized updates; viewable by students and staff. | 'Jane Smith' |
| Lecturer Phone | Internal | Contact number of the lecturer. | Access restricted to authorized personnel; logged access. | '987-654-3210' |
| Lecturer Department | Internal | Department to which the lecturer belongs. | Access limited to university personnel; important for internal operations. | 'Computer Science' |
| Subject Code | Public | Code identifying a subject. | Basic access controls; public information. | 'CS101' |
| Subject Title | Public | Title of the subject. | Basic access controls; public information. | 'Introduction to Programming' |
| Grade | Confidential | Academic grades of students. | Encrypted storage; access restricted to students, relevant lecturers, and staff. | 'A', 'B', etc. |
| Assessment Date | Internal | Dates on which assessments were held. | Protected to ensure data integrity; logged for changes. | '2023-12-01' |
| Audit Logs | Restricted | Logs containing details about data access and changes. | Access strictly controlled; monitored and audited regularly. | Log entries |
| Login/Logout Records | Restricted | Information about user authentication sessions. | Access strictly controlled; used for security monitoring and compliance audits. | Timestamps, user IDs |
| Created By (in Results) | Internal | Identifier of who created academic records. | Access controlled; necessary for audits and tracking academic record entries. | 'L2001' |
| System Logs | Restricted | Logs that monitor and record system operations. | Access strictly controlled; critical for troubleshooting and security monitoring. | System event entries |

As shown in the table above, data in the AIS system are classified into four types which are Public Data, Internal Data, Confidential Data and Restricted Data.

**Public Data**

**A close up of words

Description automatically generated**

As shown in the figure above, subject codes and titles are examples of the types of information that are included in this category. Both of these types of information are permitted to be disclosed to the general public without causing any negative effects. The level of security that is present here is rather low, and its major objective is to ensure that the data has both availability and integrity.

**Internal Data**

This category includes the bits of information that are necessary for day-to-day activities but are not particularly sensitive about the information they include. A few examples include the names and contact information of the students enrolled in the course. In order to prevent any unnecessary exposure, access is often restricted to those who are already employed by the organization.

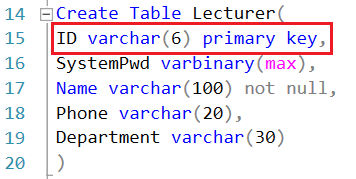
A screenshot of a computer code

Description automatically generated

As shown in the figure above, the highlighted section is an example of internal data. Lecturers' names may usually be found on public platforms such as university websites or published papers. While they need security against unauthorized alterations and exploitation, they are not as sensitive as IDs or passwords. Lecturer phone numbers must be protected to avoid abuse or unauthorized disclosure, but they provide a smaller risk than password information. The department’s information is important for internal organizational and operational reasons but does not cause major damage if leaked. It is largely used to organize and divide data and duties inside of the institution.

**Confidential Data**

This includes very sensitive information that, if disclosed, has the potential to cause harm, such as the identification numbers of students and their grades. For the purpose of preventing unauthorized access to sensitive data and ensuring that it is kept confidential, stringent security measures, such as encryption and access limits, are carried out in order to safeguard it.



As shown in the figure above, the highlighted section is an example of confidential data. The Lecturer ID is a unique identifier that creates a connection between the system and each specific lecturer. The fact that it is used to access and manage lecturer-specific information and functions, which need to be secured in order to avoid unauthorized access and impersonation, which is the reason why it is categorized as confidential.

**Restricted Data**

All of this information, which includes audit logs and passwords, is safeguarded by the highest level of security technology that is currently available. In the event that it were possible to hack into this information, it may lead to significant legal and security complications. For the purpose of monitoring access alterations and changes, some examples of protection mechanisms that are used include encryption, strong access limits, and thorough audits.

A screenshot of a computer code

Description automatically generated

As shown in the figure above, the highlighted section is an example of restricted data. This column is where the passwords for lecturer accounts are stored in an encrypted format. Due to the fact that it provides access to the lecturer's personal and professional information as well as administrative authorities inside the AIS, it is of the utmost importance to maintain this data under restricted practices.

* 1. **Data Protection Solutions**
     1. **Data Encryption**

One method for protecting the confidentiality of data is known as data encryption. This technique involves converting the data into ciphertext, which can only be decoded by using a unique decryption key that was generated during the time of the encryption or before it. The process by which plaintext is transformed into ciphertext is referred to as encryption. (geeksforgeeks.org, 2022)

3.2.1.1 Symmetric Key Encryption

In cryptography algorithms, there are a few different tactics that may be applied. In order to perform encryption and decryption procedures, some algorithms make use of a unique key. In these kinds of operations, the unique key has to be protected due to the fact that the system or individual who holds the key possesses full authentication in order to decode the message for reading. The term "symmetric encryption" refers to this encryption method, which is used in the field of data encryption. (geeksforgeeks.org, 2022)

A screenshot of a computer code

Description automatically generated

The figure above shows how Symmetric Key Encryption is set up for data protection. Frist, a master key is created to protect the certificates and symmetric keys which are stored in the database. It serves as the primary defence when securing the encryption hierarchy within the SQL server. Next, a certificate will be created to manage and secure the symmetric key that encrypts passwords. It adds an additional layer of complexity on top of the encryption keys, which makes the system more secure. Lastly, a symmetric key is created for password encryptions in the database. As shown in the figure, AES-256 has been chosen as the encryption algorithm as Advanced Encryption Standard (AES) 256 usus 256-bit key converter to encrypt plain texts into cipher texts, which makes it an almost uncrackable symmetric encryption algorithm. (Kananda, 2022)

A screenshot of a computer code

Description automatically generated

The figure above shows an example of how Symmetric Key Encryption takes action, focusing on sensitive data encryption such as passwords before storing it in the database. The symmetric key ‘*PasswordEncryptionKey’* is opened using the ‘*AISServerCert’* certificate before using it for encryptions which is done using the command ‘*OPEN SYMMETRIC KEY PasswordEncryptionKey’* followed by *‘DECRYPTION BY CERTIFICATE AISServerCert’*. Next, the *‘EncryptByKey’* command is used to encrypt the temporary password with the symmetric key opened previously. The *‘(Key\_GUID('PasswordEncryptionKey')’* is used as a function call to retrieve the unique key. Lastly, the symmetric key is closed using the command *‘CLOSE SYMMETRIC KEY PasswordEncryptionKey’* after the encryption and data insertion are completed to secure it.

A screenshot of a computer

Description automatically generatedAs shown in the figure above, the passwords have been encrypted into ciphertext therefore securing the integrity of the user’s data.

3.2.1.2 Transparent Data Encryption

Transparent Data Encryption (TDE) is a security solution that encrypts data at the storage layer. This protects sensitive data that is stored in database files that are stored on disc. The data is encrypted and decrypted on the fly using TDE while it is being written to or read from the storage. This is accomplished without the need for any changes to be made to the application's code. Not only does this ensure that data is encrypted while it is stored, but it also provides an essential layer of protection against unauthorized access, which is especially important in circumstances when physical security systems are ineffective. (Ahmed, 2022)

A screenshot of a computer

Description automatically generated

The figure above shows the process of how TDE is set up. First, the command *‘CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'MzF!0B$r3H6v&dK'’* is used to create a master key in the master database, securing the encryption hierarchy in the SQL server. This key will be used for the encryption of certificates and symmetric keys stored. Next, a certificate is then created using the command *‘CREATE CERTIFICATE TDEAISCertificate WITH SUBJECT = 'TDE AIS Certificate'’* which will be used specifically for TDE only, which is responsible for protecting the Database Encryption Key (DEK). The command on line 826 to 828 shown in the above figure is used to create the DEK to encrypt the database using the TDE certificate created previously. Lastly, the command on line 832 and 833 as shown in the above figure is used to activate the TDE on the AIS database.

After enabling the TDE, we can then verify that TDE is enabled by right-click on the database and then clicking on Properties selection. Next, click on Options after Properties windows is opened, then scroll down to State section and look for Encryption Enabled. (Filip Holub , 2020) We can confirm that our TDE is in fact working as Encryption Enabled is indicated as True as shown in the following figure.

A screenshot of a computer

Description automatically generated

* + 1. **Backup and Recovery**

The practice of generating and preserving copies of data that may be used as protection against the loss of data is known as backup and recovery or operational recovery. The process of recovering data from a backup often entails restoring the data to the spot where it was originally stored, or to a different location where it may be utilised in lieu of the data that was lost or destroyed. (NetApp.com, 2023)

A backup is created with the intention of producing a copy of the data that can be retrieved if the main data fails to be recovered. There are a number of factors that may lead to primary data failures. These include failures in hardware or software, data corruption, or events that are induced by humans, such as an attack of virus or malware, or even the accidentally loss of data. The ability to restore data from an earlier point in time is made possible by backup copies. (NetApp.com, 2023)

3.2.2.1 Encrypted Backup

Encrypted Backup is the action of encrypting a backup file. It is important to encrypt backup files which are important and need a high level of integrity to ensure that no one will be able to access the data in the event of an attack or unauthorized access to the backup files.

A screenshot of a computer program

Description automatically generated

The figure above shows the creation of encrypted backups of certificate and private keys. These backups are crucial in the event of disaster recovery or moving encryptions to another server. The first block of SQL commands which is on line 840 to 848 in the figure above is used to backup the certificate used for the TDE. The next block of SQL commands which is on line 854 to 862 in the above figure is used to backup the certificate used for password encryptions in the AIS database.

3.2.2.2 Backup Automation

Automated Backup is also implemented in the AIS database with the help of SQL Server Agent. As shown in the following figure, the command on line 865 to 871 is used to create a new job in the SQL Server Agent named “Automated\_Backup” which will then be assign tasks to run scripts later. Next, a job step will be added. The command on line 874 to 882 is used to add a step in the job, which specifies the actual backup command. A specified file path will be directed to the SQL Server to backup the AIS database and is also configured to retry backups up to 5 times in an interval of 5 minutes if the backup fails. The next step is to schedule the job. As shown on line 885 to 893 in the figure below, the command is used to set up a schedule to run the job once every 6 hours every single day. After creating a schedule, it will then be link to the job to ensure the job runs according to the schedule created as shown on line 896 to 899. Lastly, the job will be enabled and start running along side with the SQL Server Agent as shown on line 902 to 907, and then added to the SQL Server Agent’s active job list using the command shown in line 910 to 912.

A screenshot of a computer

Description automatically generated

In conclusion, each of the steps that were mentioned above addresses different risks, such as unauthorised data access, data breaches, and unauthorised data alteration, and assures compliance with regulations and standards related to data protection. As a result of these solutions, the AIS database is made more secure against both internal and external attacks, while simultaneously preserving the data's integrity and confidentiality.

1. **Auditing**
   1. **Audit Matrix**

Auditing, a process referred to tracking and logging any kind of events, actions, and activities that occur within a database server environment. Those kind of events, actions, and activities may include data modifications, access attempts, schema changes, login attempts, and more (ManageEngine, 2022).

In cases of a security incident or data breach, the database that has implemented auditing will be able to identify unauthorized access attempts or suspicious activities within the database. By monitoring and recording database activities, the database’s admin can detect and prevent security breaches or unauthorized access to sensitive data. There are also situations where industries and organizations have regulatory requirements that mandate the auditing of database activities, making it easy for the authority to trace back the activities from the auditing logs, if something ever goes wrong or if malicious attacker accessed or modified the data unauthorized (SatoriCyber, 2023).

|  |  |  |
| --- | --- | --- |
| Audit Area | Audit Objectives | Audit Solution |
| Data Modifications | Capture data changes from the tables: Student, Lecturer, Subject, and Result | Create triggers for auditing data changes |
| Structural Modifications | Capture structural changes from the database objects: Tables, Procedures, Views, and Functions | Create trigger for auditing structural changes |
| Permission Modifications | Capture permission changes from the roles: DBAdmin, Student, and Lecturer | Create trigger for auditing permission changes |
| Login And Logout | Capture and generate login and logout event | Create trigger for login history table and stored procedures |

In summary, the database security audit matrix above highlights the key audit areas of database activity that should be closely monitored for security, compliance, and troubleshooting purposes. Our group suggest that each of the audit area has their own specific auditing mechanisms to effectively capture and log relevant information, such as triggers for data and structural modifications, as well as stored procedures and login history tables for tracking access attempts.

* 1. **Auditing Data Modifications**

Auditing for data change in the AIS database involves the creation of four separate triggers, one for each of the following tables: “Student”, “Lecturer”, “Subject”, and “Result”. These triggers serve the same purpose of auditing data modifications, and the SQL queries used to create them are also almost identical.

By having separate triggers for each table, the auditing process can capture data changes specific to each table, providing a comprehensive audit trail for the entire database. This auditing approach ensures that any unauthorized or unintended modifications to the data can be tracked and investigated, enhancing the overall security and integrity of the AIS database.

All the 4 triggers follow the same logic and structure as the “AuditDataChanges\_Student” trigger in the figure below, with minor variations to accommodate the different table structures and columns. The core functionality remains the same, which are determining the type of DML operation including “INSERT”, “UPDATE”, or “DELETE”, capturing the relevant data and inserting a log entry into the centralized “AuditLog” table.

A screenshot of a computer program

Description automatically generated

The trigger determines the type of DML operation that invoked it by checking the presence of rows in the “inserted” and “deleted” virtual tables. If rows exist in both “inserted” and “deleted” tables, it means an “UPDATE” operation occurred. The “@Data” variable is populated with the inserted and deleted rows, along with an “[@Action]” attribute indicating whether the row was inserted or deleted. If only the “inserted” table has rows, it means an INSERT operation occurred, and the “@Data” variable is populated with the inserted rows and an “[@Action]” attribute set to “inserted”. If there are no rows in the “inserted” table, it means a DELETE operation occurred, and the “@Data” variable is populated with the deleted rows and an “[@Action]” attribute set to “deleted” (Malhotra, 2022).

The “@SqlStatement” variable is set to a descriptive string indicating the DML operation and the table it was performed on. Finally, a new row is inserted into the “AuditLog” table, capturing the event type, event date and time, username, schema name, object name, SQL statement, and the changed data in XML format. The “CAST(‘’ AS XML) AS SystemPwd” section in the “SELECT” statements is used to avoid capturing the encrypted password data in the audit log, as the “SystemPwd” column stores encrypted passwords, and it is excluded from the audit log to maintain data confidentiality (Richardson, 2019).

A screenshot of a computer

Description automatically generated

Based on the figure above, this trigger effectively captures all data modifications such as “INSERT”, “UPDATE”, and “DELETE” operations on the “Student” table and logs the relevant information in the centralized “AuditLog” table for auditing purposes.

* 1. **Auditing Structural Modifications**

The auditing mechanism for structural modifications in the AIS database is implemented through the “AuditStructuralChanges” trigger. This trigger is defined at the database level and captures events related to creating, altering, or dropping tables, procedures, views, and functions.

A screen shot of a computer

Description automatically generated

Based on the figure above, the “AuditStructuralChanges” trigger uses the “EVENTDATA()” function to retrieve the event data in XML format. The captured event data will contain information about the DDL operation that fired the trigger, including the SQL statement, event type, object name, and schema name. The trigger extracts the relevant information from the event data XML and declares variables to store these values.

Once the necessary information is extracted just like in the figure below, the trigger will insert a new row into the “AuditLog” table. This row captures the event type, event date and time, username, schema name, object name, SQL statement, and the complete event data in XML format. By logging these details, the trigger provides a comprehensive audit trail for any changes made to the database schema.

A screenshot of a computer

Description automatically generated

* 1. **Auditing Permissions Modifications**

The auditing mechanism for permission modifications in the AIS database is implemented through the “AuditPermissionChanges” trigger. This trigger is defined at the database level and captures events related to granting, denying, or revoking database-level permissions, as well as adding or removing role members. It is executed whenever any of these permission-related operations occur.

A screen shot of a computer

Description automatically generated

Based on the figure above, the “AuditPermissionChanges” trigger is extremely alike to the “AuditStructuralChanges” trigger. This similarity in structure and implementation promotes code reusability and consistency within the auditing mechanisms of the AIS database. By leveraging a common approach, the database administrators can ensure a uniform auditing process across different types of events, simplifying the maintenance and future enhancements of the auditing system.

A screenshot of a computer

Description automatically generated

Based on the figure above, by implementing this auditing mechanism, the AIS database enhances its overall security and compliance by providing a detailed record of who made what changes to permissions and when, enabling effective monitoring and investigation of potential security incidents or policy violations.

* 1. **Auditing Login and Logout**

The last auditing mechanism for the login and logout events in the AIS database is implemented through a combination of a table and stored procedures. This approach ensures that login and logout activities are properly recorded and monitored, providing valuable information for security and compliance purposes.

A screenshot of a computer program

Description automatically generated

Firstly, a “LoginHistory” table is created to store the login and logout records for all the users in the AIS database. In the figure above, the table contains columns to store either the student’s ID or the lecturer’s ID, the login time, the logout time, and a “bit”, to indicate whether the login attempt was successful or not (Gigoyan, n.d.).

A screenshot of a computer code

Description automatically generated

A screen shot of a computer program

Description automatically generated

Then 2 procedures were created to store the record of login and logout, respectively in both “RecordLogin” and “RecordLogout” stored procedures in the AIS database. These 2 procedures in the figures above share a similarity in their implementation and work together to capture login and logout events for users of the system, with “RecordLogin” responsible for recording login attempts and “RecordLogout” for updating the logout time when a user logs out.

In the case of “RecordLogin”, a new record is inserted into the “LoginHistory” table, capturing the user ID, current date and time as the login time, and the success status. On the other hand, “RecordLogout” updates the logout time for the most recent login record in the “LoginHistory” table, setting the “LogoutTime” column to the current date and time.

A screenshot of a computer program

Description automatically generated

Finally, the figure above is the “GenerateLoginLogoutReport” stored procedure, it is designed to generate a report of login and logout activities for the current day. This procedure checks if the executing user is a member of the “DBAdmins” role or has the “sysadmin” server role. If the user has the required permissions, it retrieves and displays all login and logout records from the “LoginHistory” table for the current date, just like in the figure below.

A screenshot of a computer

Description automatically generated

By implementing this auditing mechanism, the AIS database can keep track of all login and logout activities, including successful and failed login attempts. The centralized “LoginHistory” table serves as a repository for this information, while the stored procedures provide a structured way to record and retrieve login and logout data.

1. **Summary**

In the case for an educational institution like a school, the database will certainly contain sensitive data about students, lecturers, and academic records. If a database system did not utilize any kind of security measurement, malicious attackers could grant access to the data, which would lead to privacy breaches, data corruption, or even legal implications. Hence, the objective of this project is to ensure that the AIS database system can protect the sensitive data and the users can only access and change information based on their authorized roles and permissions.

This led to the need for implementing security features for the fundamental principle of the CIA triad. By implementing role-based access control (RBAC) and permission controls in the database, the system ensures that users can only perform operations that are within the scope of their assigned roles. The database also implements both symmetric key encryption and transparent data encryption, as well as enabling secure backup and recovery mechanisms to safeguard against data loss or unauthorized access.

The incorporation of features like temporary passwords for new accounts, password changes requirement upon first login, and audit logging further enhances the security of the database. By implementing multiple layers of security controls in the database, such as access control, encryption, auditing, and monitoring, the system can effectively mitigate various threats and vulnerabilities, ensuring the confidentiality, integrity, and availability of sensitive data within the database.

1. **References**

Ahmed, I. (2022, December 27). *Transparent Data Encryption (TDE)*. Percona Database Performance Blog. https://www.percona.com/blog/transparent-data-encryption-tde/

Berning, T. (2023, July 3). *What is Row-Level Security?* NextLabs. https://www.nextlabs.com/what-is-row-level-security/

Filip Holub. (2020). *Configure SQL Server Transparent Data Encryption with PowerShell*. Www.mssqltips.com. https://www.mssqltips.com/sqlservertip/6316/configure-sql-server-transparent-data-encryption-with-powershell/

GeeksforGeeks. (2020, June 5). *Advantages and Disadvantages of Using Stored Procedures - SQL*. GeeksforGeeks. https://www.geeksforgeeks.org/advantages-and-disadvantages-of-using-stored-procedures-sql/

geeksforgeeks.org. (2022, January 16). *What is Data Encryption?* GeeksforGeeks. https://www.geeksforgeeks.org/what-is-data-encryption/

Gigoyan , S. (n.d.). *SQL Server Bit Data Type*. Www.mssqltips.com. https://www.mssqltips.com/sqlservertip/6447/sql-server-bit-data-type/

Kananda, V. (2022, June 22). *What is AES 256 Encryption & How Does it Work?* Progress Blogs. https://www.progress.com/blogs/use-aes-256-encryption-secure-data

Malhotra, H. (2022, December 9). *Data Activity Tracking Using SQL Triggers*. SQLServerCentral. https://www.sqlservercentral.com/articles/data-activity-tracking-using-sql-triggers

ManageEngine. (2022, February 10). *Understanding SQL Server Audit better*. ManageEngine. https://www.manageengine.com/products/eventlog/logging-guide/understanding-sql-%20server-audit.html

NetApp.com. (2023). *What Is Backup and Recovery? - Why It’s Important | NetApp*. Www.netapp.com. https://www.netapp.com/cyber-resilience/data-protection/data-backup-recovery/what-is-backup-recovery/

Pavel, P. (2022, October 17). *Database Views: What You Need To Know*. Aristek Systems. https://aristeksystems.com/blog/database-views-what-you-need-to-know/

Ravikiran. (2021, May 10). *Stored Procedure in SQL: Benefits And How to Create It*. Simplilearn.com. https://www.simplilearn.com/tutorials/sql-tutorial/stored-procedure-in-sql

Richardson, B. (2019, October 11). *Working with XML Data in SQL Server*. SQL Shack - Articles about Database Auditing, Server Performance, Data Recovery, and More. https://www.sqlshack.com/working-with-xml-data-in-sql-server/

SatoriCyber. (2023, November 2). *Database Auditing*. Satori. https://satoricyber.com/cloud-data-governance/database-auditing/

SecurePass. (2021, March). *Role-Based Access Control Management System*. Www.thesecurepass.com. https://thesecurepass.com/blog/role-based-access-control-system

Szentgyorgyi-Siklosi, A. (2023, July 6). *What is a Data Classification Matrix?* Lepide Blog: A Guide to IT Security, Compliance and IT Operations. https://www.lepide.com/blog/what-is-a-data-classification-matrix/

Team Post. (2022, April 7). *What is a Database View? (And How Does It Help Business Intelligence)*. Dashboardfox.com. https://dashboardfox.com/blog/what-is-a-database-view-and-how-does-it-help-business-intelligence/

TutorChase. (2023). *What are the advantages and disadvantages of using a view in SQL?* Tutorchase.com. https://www.tutorchase.com/answers/a-level/computer-science/what-are-the-advantages-and-disadvantages-of-using-a-view-in-sql

Uikey, P. (2023, March 17). *What is Stored Procedure in SQL? | DataTrained*. DataTrained. https://datatrained.com/post/stored-procedure-in-sql/#:~:text=A%20stored%20procedure%20is%20a

Zahid, A. (2023, April 13). *What is Row Level Security in SQL? What are Best Practices and How to Implement them?* Medium. https://mrasimzahid.medium.com/what-is-row-level-security-in-sql-what-are-best-practices-and-how-to-implement-them-696f016718bb