Informatyka techniczna i sztuczna inteligencja
Technical informatics and Artificial Intelligence
Engineer of XXI Century'2024

Ensuring information systems security based on database servers

Roman Syrotynskyi [1], Ivan Tyshyk [2,*]

[1] *Lviv Polytechnic National University, postgraduate student of Information Protection Department, roman.m.syrotynskyi@lpnu.ua*
[2] *PhD, Lviv Polytechnic National University, Associated Professor of Information Protection Department, ivan_tysh@i.ua*
* *Corresponding author, ivan_tysh@i.ua*

Abstract: Ways to increase the security of database servers when operating on a network and using the database as the core of an enterprise security system were explored. Mathematical models and algorithms for information protection in relational database management systems have been developed. A mechanism for restricting access using modified SQL queries is proposed. Methods of request modification were analyzed and classified.

Keywords: Secure database, Information protection, Information system, Unauthorized access, Security policy, Database management system, Data integrity;

Zapewnienie bezpieczeństwa systemów informatycznych na podstawie serwerów baz danych

Roman Syrotynskyi [1], Ivan Tyshyk [2,*]

[1] *Lviv Polytechnic National University, postgraduate student of Information Protection Department, roman.m.syrotynskyi@lpnu.ua*
[2] *PhD, Lviv Polytechnic National University, Associated Professor of Information Protection Department, ivan_tysh@i.ua*
* *Corresponding author, ivan_tysh@i.ua*

Streszczenie: Przeanalizowano sposoby poprawy zabezpieczenia serwerów baz danych podczas pracy w sieci oraz wykorzystania bazy danych jako jądra systemu bezpieczeństwa przedsiębiorstwa. Opracowano modele matematyczne i algorytmy ochrony informacji w relacyjnych systemach zarządzania bazami danych. Zaproponowano mechanizm ograniczania dostępu z wykorzystaniem zmodyfikowanych zapytań SQL. Dokonano analizy i klasyfikacji metod modyfikacji zapytań.

Słowa kluczowe: Bezpieczna baza danych, Ochrona informacji, System informatyczny, Nieautoryzowany dostęp, Polityka bezpieczeństwa, System zarządzania bazami danych, Integralność danych;

1. Introduction

Financial-economic and managerial activities are transitioning from non-automated methods and tools to information systems. This transition simplifies data accounting and analysis, saves costs, etc. It is necessary that the security of information in IS against theft and forgery should be at least no less than in management systems based solely on paper document flow. Ideally, the change in technology and information processing tools should enhance data security.
Considering the widespread use of relational databases (DB) [5, 6], the large volume of information accumulated in existing DBs from various subject areas, and the existence of standardized tools for database management systems (DBMS), the protection of DBMS and DBs is an important task for information security specialists. This work focuses on modeling the security system specifically for relational databases, addressing the protection of other IS components as necessary.

DOI: https://doi.org/10.53052/9788367652292.07

An analysis of the current state of research in the fields of computer science, DBMS theory, decision-making theory, and information security highlights several unresolved issues. Among them are:

- the absence of mathematical models for relational DB security systems that take into account the need for statistical information protection, quotas, optimization of the security system's resource consumption, auditing at the tuple and tuple attribute levels, and ensuring the ability to reproduce the history of data changes for any period;
- an insufficiently developed methodology for designing relational DB schemas considering information security requirements;
- existing models inadequately address authorization tasks for DBMS users [6] and several special DB security tasks (statistical protection, protection from excessive information acquisition, etc.) [7].

Unauthorized destruction, modification, copying, and blocking of authorized access to data are considered threats to information. Security breaches of information systems (IS) are viewed as the realization of threats to information. Depending on the purpose of the IS, the importance of protection characteristics changes. For example, in economic IS, data integrity and availability are of the highest importance, while in military and government IS, confidentiality and information authenticity are among the main requirements for IS.

The tasks of an IS are to provide information support for decision-making processes [4]. Various technologies for storing and processing information, such as OLAP, OLTP, DSS, Data Warehouse, expert systems, etc., are used to solve this task. For many economically focused IS, the goals are to increase profitability and reduce costs through decision support processes (DSP).

Considering the purpose of the IS, the security objectives are:

1. Timely provision of reliable information for decision-making processes based on the data entered into the IS.
   1.1. Sufficient data completeness for DSP.
   1.2. Accuracy of output data for decision-making processes (during data storage and DSS).
   1.3. Sufficient speed of decision-making.
      1.3.1. Availability of decision-making mechanisms.
      1.3.2. Guaranteed response after receiving a request (even "unknown" should arrive on time).
2. Ensuring competitiveness (antagonistic business activities, avoiding the creation of more favourable conditions for competitors due to the use of the IS).
   2.1. Preservation of confidentiality (not creating more favourable working conditions for competitors by supporting their DSP with one's own data).
   2.2. Disinformation of the adversary (violating points 1.1 and 1.2 in the information protection system of a competitor using industrial espionage).

An information security system (ISS) is a hardware and software complex designed to solve the tasks of IS protection. The tasks presented by the author are universal for information system security and apply to ISS regardless of the domain, specifics, and technologies of the IS.

Accordingly, the above ISS tasks for IS are projected into the tasks of database security systems and IS based on relational databases

2. Analysis of research and publications

According to data from the Computer Security Institute (USA) [13], the causes of IS security breaches are as follows: 3% – viruses, 2% – unauthorized access to IS from other IS (including from the Internet), 20% – technical failures and hardware malfunctions, 55% – errors and unqualified actions by legitimate IS users and service personnel, and 20% – intentional actions by legitimate IS users aimed at harming the IS and violating security (Fig. 1).

Thus, the most frequent cause (75%) of IS security breaches is related to incorrect actions by legitimate IS users, i.e., users who have successfully passed authentication. These actions (55% accidental and 20% intentional) cannot be prevented by additional authentication tools [9, 11], firewalls, cryptographic protection of communication lines, or VPN products.

DOI: https://doi.org/10.53052/9788367652292.07

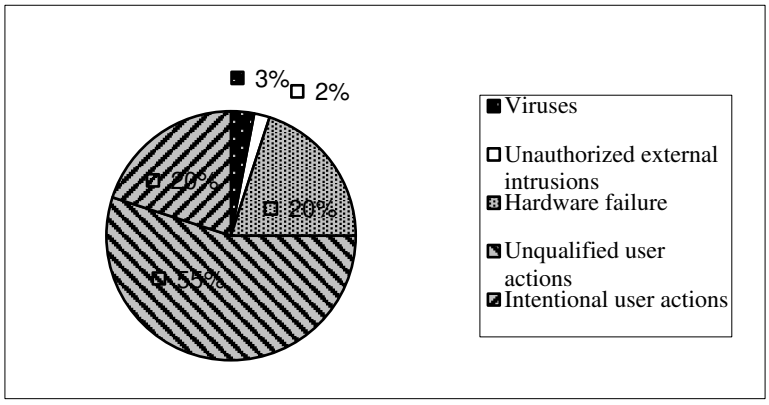

Figure 1. Causes of information security breaches

Secure Databases. Given the widespread use of database technology in IS, the large volumes of critical information stored in databases, and the statistical data on the causes of 75% of security breaches, as shown in Figure 1, several tasks arise [12]:

1. Creating a secure environment for database users, ensuring data integrity, and preventing unauthorized disclosure or modification of data by other database users.
2. Minimizing users' access to unnecessary data, both for reading and for creating, modifying, and deleting data.
3. Ensuring data accuracy and integrity in IS.
4. Developing efficient tools for administering database security policies.
5. Integrating the information security system (ISS) into the IS at the database schema design stage.

A database that addresses these tasks is called a secure database. The concept of a secure database (SDB) is introduced in this work for the first time. Unfortunately, in practice, some security tasks cannot be fully implemented using commercial DBMS, including:

- ensuring confidentiality (including statistical protection);
- ensuring data integrity (including context-dependent protection);
- auditing;
- complexity of administering database security policies.

Limitations of Authorization Models in Commercial DBMS. DBMS can restrict access to database objects at the table, view, or individual attribute level [1-5, 10, 12]. Access is differentiated for reading, modifying, deleting, or writing new data. In practice, it is necessary to grant users (e.g., from different departments) access to specific tuples (or fields within some tuples) while restricting access to other tuples in the same relation. This cannot be achieved using standard commands like CREATE ROLE…, GRANT…, REVOKE… [4, 13]. There are two approaches to ensuring detailed protection at the tuple level:

1. Using a multilevel relational database (MRDB).
2. Using a secure database (SDB).

The implementation of a multilevel relational database (MRDB) is based on augmenting the existing database schema with an additional attribute L for storing security labels or their equivalents. This essentially transforms the relational database into an MRDB [12-14]. Several specialized DBMS (e.g., Trusted ORACLE, INGRES/Enhanced Security) implement multilevel security systems and mandatory access control models. DBMS with multilevel security systems support data multiversioning, where users receive ‘versions’ of documents based on their access level. However, using DBMS with the MRDB model presents several challenges:

- The access level hierarchy in specialized DBMS is not detailed enough to reflect the distribution of authority according to functional responsibilities.
- Classic multiversioning mechanisms are redundant for IS of a financial-economic (and, in general, civil) nature and impose additional computational resource burdens.
- The use of specialized DBMS is also limited by the unavailability of relevant software in the Ukrainian market (due to high cost and export restrictions to countries, including Ukraine).

Work is underway on DBMS that restrict access to individual table records by modifying users' SQL queries. These products [15] supplement SQL queries with simple additional constraints, partially implementing a mechanism similar to security label categories.

3. Database Server Protection in Network Operations

The security server (center) manages the configuration of various components of the corporate information system (collection, processing, and storage of system information) in accordance with a unified IS security plan.
Active agents of the server are installed on workstations and servers, ensuring access rights separation, user activity logging, and cryptographic data protection. Additionally, integration with security analysis and attack detection tools is provided. Security analysis tools allow checking known security breach paths, identifying weak passwords, generating reports, and monitoring the operational systems' activity statistics. The best analysis tools constantly update the database of security "vulnerabilities" and have built-in expert systems for security and administration. Using corporate databases to store data collected by security servers allows for long-term monitoring of security parameters and auditing operations. Using industrial database servers enables the application of powerful OLAP and DSS systems for effective analysis of the collected data [15].
Figure 2 shows the structure of a corporate network utilizing security and certification servers. Certification server, security server, security system operation analysis, database server, corporate data, security policy data, IS audit results, protected corporate services, user, user, firewall, corporate services, uncontrolled communication channels, corporate database information, DBMS protection tools, OS user authentication tools, lower-level network protocols, workstation software, technical protection, organizational measures, database user, server OS user.

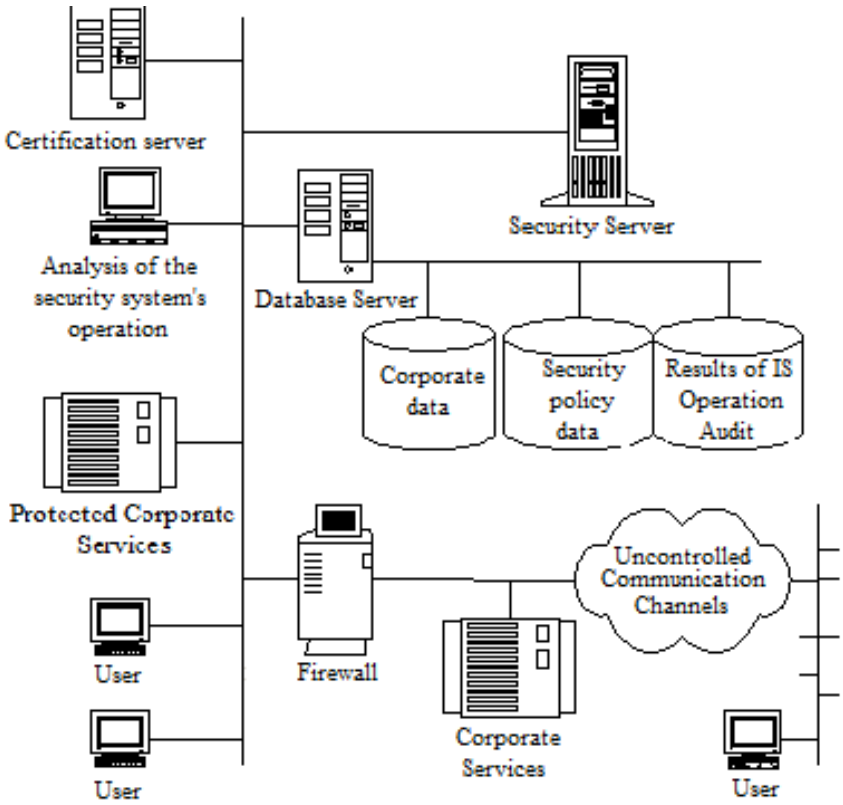

Figure 2. Corporate network using security and certification servers

To account for information regarding IS security policies, the practical implementation of security policies, and to manage and process IS user data, a database server is used. Storing this information in a database allows the design of a distributed security system utilizing well-established mechanisms of distributed computing and ensuring information integrity within the DBMS.
Database servers generally form the backbone of automated information processing in corporate IS. The high concentration of information in databases makes them a focal point for competitors and other interested parties. A breach in database security threatens the loss of valuable information, the inability to make correct decisions based on inaccurate data, and more.
The issue of attacks on database servers does not receive sufficient attention in publications. When assessing the security of a network database server, the following model is used: Database Security = Client Security + Network Security + Server OS Security + DBMS Security. In this model, DBMS security focuses on user identification and authentication, as well as protecting the “DB – authenticated user” information channel.

In reality, a database server can be attacked by a legitimate network and server OS user without the use of special client-side software. The client/server technology allows access to the DBMS even without prior identification as a corporate network and server OS user. A legitimate network and server OS user can access the database bypassing DBMS data exchange protocols. These access schemes are illustrated in Figure 3. The paths for database and OS users to access database information shown in Fig. 3 are controlled by the following data protection mechanisms:

1. Cryptographic protection of database files using DBMS tools.
2. Data protection at the database user level (cryptographic protection of database objects, access and operation rights distribution for database objects, semantic data protection, etc.).
3. DBMS user authentication and authorization, checking for allowed working hours, etc.
4. Cryptographic protection of information, subscriber encryption, etc.
5. Data protection at the OS user level (cryptographic file protection, access separation to system resources, access rights and operation distribution for resources, environment parameter configuration, etc.).
6. Server OS user authentication and authorization, user session initiation procedures, etc.
7. Ensuring data integrity and authenticity, collecting data for authentication procedures.
8. Performing checks at the user's workstation, presenting data in the required format.
9. Meeting general security requirements for IS users.

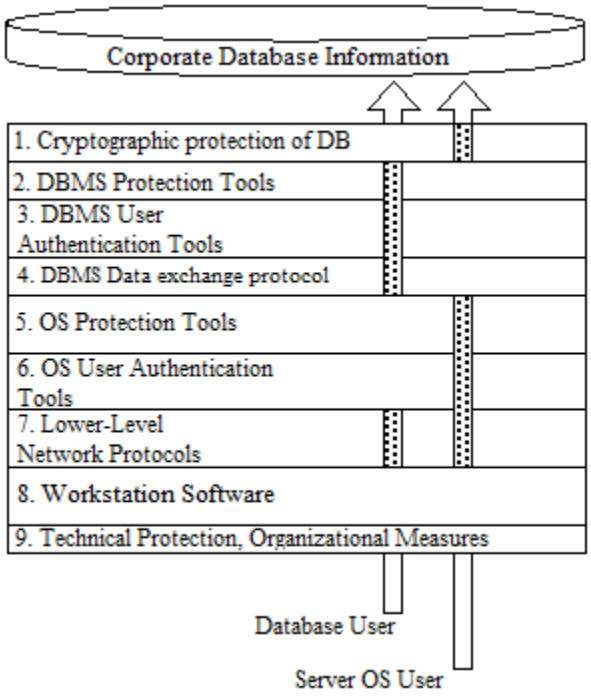

Figure 3. Paths of user access to the DBMS and server OS to the database

In Figure 3, only possible channels for unauthorized user access through the network are considered. Options such as physical access to the database server, physical information carriers, hardware emissions, and other similar methods are not described. When multiple services (DBMS, Web and mail server, file server) are running on a single network server, it becomes possible to attack one service through vulnerabilities in the protection of another. Examples of such attacks may include service blocking by resource overconsumption beyond the quota, complete computer shutdown, and direct access to the file system and virtual memory. Even the network protocols used by the server can pose a threat to services. Installing several complex-to-install and administer software products can lead to unnoticed changes in security parameters by the administrator, introduced by the new versions of the software products (restoration of OS default values, overriding event responses, absence of certain features in Beta, Trial, or Lite versions, etc.). The mutual overriding of system parameters during the installation and administration of different complex software products can result in unpredictable consequences under the specific OS conditions. An example of this is the appearance of DBMS system passwords in plain text in the OS system files. In this case, several users of the file server, on which the DBMS was running, had access to the database passwords.

Thus, it is advisable to install different services on separate servers. This is especially relevant for separating internal corporate services from those serving external users. It is also beneficial to separate services based on functionality –

separating data storage and processing services from those that display and transmit information to users. An example of this would be hosting the database server and the Web server on different machines.

Using Web servers as gateways between users and the DBMS allows deviation from the traditional authentication scheme and enables user authentication via the Web server. A user can use one of the common authentication mechanisms, such as electronic certificates. In turn, the Web server, according to its own user password data in the DBMS, initiates a session with the database using internal corporate authentication mechanisms. However, in the event of a security breach of the Web server or its authentication system being compromised, unauthorized access to the DBMS becomes possible.

In light of the aforementioned threats, DBMS are evolving towards creating their own security tools, taking into account the possibilities of data access on the computer, alternative user administration, and more.

To protect information within the database, DBMS use cryptographic protection, store objects in formats unsuitable for reverse engineering of algorithms and other intellectual property, and reduce the risk of misuse or tampering with software components, even by privileged users. Additionally, cryptographic transformation of database files is used. Cryptographic protection prevents access to information bypassing the DBMS, during DBMS downtime, or in the event of database file tampering.

Continuing in the direction of controlling system resources, DBMS are starting to include OS and file system functions. The DBMS includes the necessary minimum OS functions, thereby eliminating some potential security vulnerabilities. How practically reliable database servers with built-in OS functions within the DBMS will be remains to be seen.

Another direction of DBMS development is improving the administration tools for their own security systems. Database security administration includes: managing access to the database, managing permissions (direct and indirect user rights), resource limitations (memory volumes, work priorities, etc.), and monitoring password aging and parameters (minimum password strength requirements, password history, mechanisms for password blocking and aging).

DBMS data exchange protocols with users allow the exchange of passwords and data in a cryptographically protected form. The use of secure protocols, together with network filters, enables limiting access to the database to registered users only, and exclusively through the DBMS data exchange protocol.

To reduce losses when an attacker uses data modification operations, it is necessary to restrict the use of data modification operations by prohibiting certain operations and partially limiting the application of others. It is clear that the resulting set of operations must be complete in the sense that users can perform their functional duties using the operations from this set.

To achieve this, the following actions are required:

1. Prohibit the operation of attribute value replacement (UPDATE).
2. Restrict the right to delete tuples to prevent:
    - the destruction of historical information and audit data;
    - replacing UPDATE with a combination of DELETE + INSERT;
    - cascading deletion;
    - the destruction of traces of unauthorized interventions.

Thus, the restricted set of operations for database users includes the operations INSERT, SELECT, and DELETE (with some limitations on the use of DELETE).

For convenience, these operations can be grouped into procedures and executed via the EXECUTE command. The combination of DELETE + INSERT allows performing an update operation. However, the DELETE operation gives an attacker the opportunity to erase traces of intervention in the IS. Moreover, data deletion disrupts the system's historical record and the analysis of the basis for decision-making. Therefore, the author suggests limiting the use of the DELETE operation (or even completely eliminating it). Specifically, deleting tuples that contain historical information about protected objects and audit data, as well as tuples referenced by foreign keys, is prohibited.

Cascading data deletion is particularly undesirable. In the event of the deletion of an object, cascading deletion often removes tuples from several relations that reference on this one (frequently without additional warnings).

Of course, the database administrator is allowed to use all operations, including UPDATE and DELETE. Sometimes, the use of UPDATE and DELETE significantly simplifies database administration tasks.

Deleting data for "cleaning the database" is a database administration task. Therefore, the rights to delete data for "cleaning" should be granted only to database administrators. Given the continuous decrease in the cost of computing resources and storage devices, the need for "cleaning the database" is constantly diminishing.

To maintain IS functionality without using UPDATE and DELETE operations, it is necessary to ensure the system’s historical record. This requires properly classifying data and designing the database schema considering the information protection requirements and the existing DBMS audit system.

4. Concept Of User Access to Secure Databases (SDB)

To access the database data, an employee must register on the database server using their DBMS username (login) and password. The reliability of the authentication procedure is ensured by ORACLE DBMS tools. The result of registration is access to the database resources authorized by the administrator. Each SDB user belongs to one of the departments and represents the access rights of a specific official to the SDB data.
A department may have multiple (or no) SDB users.
All users within a department have the same rights to access the SDB data. The list of access rights is based on the functional responsibilities of the department. In the database, user access rights are stored as roles.
If necessary, the database administrator can change the access rights of any user or group of users. To implement data access restrictions defined by the security policy, roles, data views, synonyms, and snapshots are used. Roles are applied in accordance with the distribution of functional responsibilities and permissions at automated workstations in the SDB. Roles grant users the rights to connect to the database, read, write, and delete data from tables, views, synonyms, snapshots, sequences, execute procedures, etc.
For flexibility in administration, roles provide access rights not directly to tables, but to table synonyms and views. If necessary, the administrator can redefine the synonym, thus providing users with access to other SDB objects (tables, views, snapshots, procedures, sequences) without changing the user's software.
Depending on the task, synonyms are created in the PUBLIC schema, the SDB administrator's schema, or the specific user's schema.
A *view* is a virtual table whose data is obtained from the base database tables and other views. A view can be seen as a named query stored in the database. A view is a database schema object and has attributes such as a unique name, a set of named attributes, corresponding domains, etc.
Views can (with some limitations) be manipulated in the same way as real database tables.
Views are used to filter data from the base database tables, simplify working with complex queries, improve data protection and ensure data integrity, and provide logical data independence in the database.
To define a view, SQL92 contains the command:

```
`CREATE VIEW <view_name> [(attribute_list)] AS <select_expression> [WITH [CASCADED | LOCAL]
CHECK OPTION]`
```

The `<select_expression>` parameter contains a SELECT command, the result of which forms the "body" of the view with the name `<view_name>`. The view name must be unique within the database schema. The attribute names of the view can be defined using either optional parameters (attribute_list) or the names (or aliases) of attributes in the `<select_expression>` query.
The `<select_expression>` can be based not only on "real tables" but also on other views, snapshots, etc. According to the SQL92 standard, the `<select_expression>` cannot contain dynamic parameters or an ORDER BY clause.
Privileges to create and work with views, as well as with tables, must be granted using the GRANT command.
The optional *WITH [CASCADED | LOCAL] CHECK OPTION* parameter applies to views used for updating data. It requires data update operations in the view to check if their results meet the view's (select expression's) conditions. The *LOCAL* parameter enforces checking compliance with the select expression, while *CASCADED* also checks the results' compliance with the conditions of all views on which the select expression is based. Using the *LOCAL* parameter may violate data integrity constraints imposed by nested views, reducing the logical independence of the data.
When update commands are applied to a view, all changes affect the base tables of that view. The inserted or modified record in the view must satisfy the data selection conditions (such as WHERE) and the constraints on the base tables. If such actions are possible, the view is called an updatable view. The SQL92 standard requires updatable views to meet the following criteria:

- The view must be based on one and only one table (or updatable view), meaning the view must not contain table joins.
- The view may contain only one query (without UNION, EXCEPT, INTERSECT).
- The view's attributes must be attributes of the base table (expressions and functions on multiple attributes are not allowed). Attributes from the base table cannot appear more than once in the view's attributes.
- The SELECT query of the view does not contain GROUP BY, HAVING, DISTINCT.

Updating data via views involves the potential for transaction competition over records in the base tables of views. Therefore, DBMSs use record locking in base tables that have undergone UPDATE and DELETE operations. Records

created by INSERT commands are unavailable to other transactions until the COMMIT command is executed. Updatable views can also be used in cursors that lock records.

In an SDB, data views are used to limit access to data from different departments for users of other departments, implementing a hierarchy of departments at the level of user access rights to SDB information.

The database administrator grants access rights to the necessary department users for the view (or view synonym) instead of granting access to the table with the required data. The view imposes access restrictions, such as:

- a list of fields needed by users;
- a list of users allowed to access specific records;
- a list of constraints on the values of individual fields.

If these restrictions are violated, the user will not receive the record's data.

The list of users allowed to access specific records is determined by the user's affiliation with a particular department.

To store information about the author of a record and the time of record creation in database tables, two system fields, CRTUSER and CRTDATE, are created. The values of these fields are the database server system username (login) who entered the record and the system time of record entry (again, from the database server). SDB users do not have rights to enter information into the CRTUSER and CRTDATE fields and cannot influence the audit data storage.

Due to the use of the SDB in the IS based on ORACLE DBMS, with restrictions on DELETE and UPDATE operations, audit data cannot be deleted. The only exception is the database administrator's rights to perform any operations on the database.

Thus, in the SDB-based IS, the *Author* function is implemented by storing the user's identifier and the record creation date within the record itself. The *Author* function's reliability is ensured solely by using a limited set of operations for working with relational databases.

The field *NOTE* is used to provide details on the reasons and circumstances of the record creation. When creating a record, the user can save their comments about the newly created record. The NOTE field implements *extrainfo*.

To monitor user actions, the database administrator can use the standard audit mechanisms of the database server (operator-level audit, object-level database audit) and specially developed reports (based on the CRTUSER, CRTDATE, and NOTE fields) for detailed analysis of the use of database objects and user actions.

The main goals of auditing an SDB-based IS are:

- tracking database object authorship;
- collecting data on user working times;
- detecting attempts to perform prohibited actions;
- identifying software errors;
- verifying the correctness of user rights.

For detailed auditing of individual database objects (e.g., the SC_USER table), the administrator can use database triggers, procedures, etc.

The use of the proposed mathematical SDB model in the IS made it possible to implement logistics accounting tasks with the storage of a significant amount of data while ensuring its integrity, confidentiality, and auditability.

Using the SDB model for IS implementation improved data security (Figure 4).

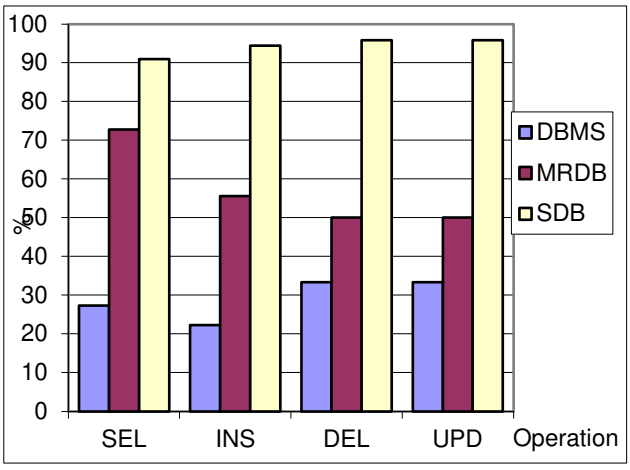

Figure 4. Comparison of the effectiveness of ISS models in terms of database operations

Figure 4 shows the ratio of threats from database operations blocked by the following means: the information security system of a relational DBMS; MRDB; and the secure database (SDB) developed based on the mathematical models proposed in this work.

Conclusion

In the context of a secure database, mechanisms for ensuring data integrity and access control have been examined. Specific threats to information confidentiality in database management systems were also considered. A new mathematical model of the protected object in relational databases has been proposed. This mathematical model is used for designing the schema of a secure database and implementing a comprehensive information security system for relational databases. Methods for implementing the mathematical model of a secure database using the relational model were defined. Methods and algorithms for implementing the mathematical model of a comprehensive information security system for relational databases using standard tools of industrial database management systems were developed. These include methods and algorithms for implementing an enhanced user authorization model and an improved user activity audit model in relational databases.

Using the example of the SDB-based IS, the implementation of the proposed mathematical models was examined, specifically: mechanisms for storing data about SDB users, the concept of user access to SDB data, the use of roles to restrict access, the use of data views to limit access, and auditing user actions.

The simulation results show that the implementation of the mathematical model of a secure database has significantly improved the efficiency of the information security system.

References

1. Murach's Oracle SQL and PL/SQL for Developers (2nd Edition) by Joel Murach 18 chapters, 648 pages, 272 illustrations Published October 2014. ISBN 978-1-890774-80-6
2. Chris J. Date. SQL and Relational Theory: How to Write Accurate SQL Code. Symbol-Plus, Series: High Tech. ISBN 978-5-93286-173-8, 978-0-596-52306-0; 2010.
3. Murach's SQL Server 2019 for Developers by Bryan Syverson and Joel Murach 19 chapters, 674 pages, 291 illustrations Published April 2020. ISBN 978-1-943872-57-2.
4. Duc Bui, V. Phuong Vu, H. Phuong Nguyen, H. Techno-economic assessment and logistics management of biomass in the conversion progress to bioenergy Sustain. Energy Technol. Assessments. 2023; 55, 102991.
5. Dotsenko S.I. Organization and Database Management Systems: Textbook. Kharkiv: UkrDUZT, 2023. – 117 pages, 92 figures, 3 tables.
6. Pasichnyk V.V., et al. Global Information Systems and Technologies: Models for Efficient Data Analysis, Processing, and Protection. Monograph / V.V. Pasichnyk, P.I. Zhezhnych, R.B. Kravets, A.M. Peleshchyshyn, D.O. Tarasov. Lviv: Lviv Polytechnic Publishing House, 2006. 348 pages. ISBN: 966-553-578-1.
7. Formal models of information security systems for relational databases. Modern Information Technologies and Innovation Methodologies of Education in Professional Training Methodology Theory Experience Problems, 47, 218-221. //URL: https://vspu.net/sit/index.php/sit/article/view/2887(accessed: 2021).
8. Ivanov, T.; Pergolesi, M. The impact of columnar file formats on SQL-on-hadoop engine performance: A study on ORC and Parquet. Concurr. Comput. Pract. Exp. 2019, 32, e5523.
9. Dmytro Matveev, Daria Fedorenko. "The Problem of Personal Data Protection on the Internet" // ΛΌΓΟΣ.ONLINE: International scientific e-journal 2019. No. 4. 63. URL: https://ojs.ukrlogos.in.ua/index.php/2663-4139/article/view/530/545 (accessed: 24.04.2021).
10. Peter P. Chen. "Entity-Relationship Modeling: Historical Events, Future Trends, and Lessons Learned." URL: http://bit.csc.lsu.edu/~chen/pdf/Chen_Pioneers.pdf (accessed: 20.04.2021).
11. Managing Claims and Authorization with the Identity Model. https://learn.microsoft.com/en-us/dotnet/framework/wcf/feature-details/managing-claims-and-authorization-with-the-identity-model (accessed: 06.01.2023).
12. Security Testing: SQL Injections. https://training.qatestlab.com/blog/technical-articles/security-testing-sql-injection (accessed: 02.04.2020).

13. Popadyuk V. V. "Encryption in SQL SERVER Databases" // Cybersecurity in the Modern World: Materials of the II All-Ukrainian Scientific and Practical Conference (Odesa, November 20, 2020).
14. Basic Security Practices for SQLite: Safeguarding Your Data. https://dev.to/stephenc222/basic-security-practices-for-sqlite-safeguarding-your-data-23lh (accessed: 03.02.24).
15. Oracle Audit Vault and Database Firewall. https://docs.oracle.com/en/database/oracle/audit-vault-database-firewall/20/sigdv/what-is-oracle-audit-vault-and-database-firewall (accessed: June 2022).