International Journal of Computer Applications (0975 - 8887)
Volume 185 - No.2, April 2023

Consequences of Violating Database Integrity Rules in Data Management Systems

Ametovi Koffi Jacques Olivier
Department of Computer Science and Applications
Koneru Lakshmaiah Education Foundation
Green Fields, Vaddeswaram, Guntur, Andhra Pradesh, India

ABSTRACT

Nowadays every action we do produces data like most computer applications. To this point moreover, some of these are reduced to managing and consulting data: our bank accounts, registrations, insurance, etc. This data is stored in what is called a file and is structured into records containing details such as our surnames, first names, date of birth, blood type, etc. More complex applications require data whose structure is also complex. The data is classified in several files according to the objects they describe: customer file, product file, order file, invoice file, etc. There are links between the files that reflect the relationships between the objects described. It also appears that the data necessary for an application could be useful to other applications, or even to other users. This data then constitutes what is called a database. Managing such data is no longer within the reach of elementary software. Guaranteeing the quality of the recorded data (can we find what was recorded?), their consistency (is the customer of each order listed?), protecting them in the event of an incident, allow several users to access them simultaneously, while strictly controlling access to confidential data, offering good access performance to all applications, in particular those that are interactive, are functions that require powerful and complex software like the database management systems, or DBMS such as SQL SERVER, MYSQL, ORACLE SQL. This type of software is one of the fundamental tools for the development of large computer applications thanks to these rules of integrity, namely the integrity of entities and referential integrity which allow the solidity and robustness of data, But these rules which must guarantee the essential security of the database for our various organizations are often violated by inattention or ignorance of these rules by the new developers and even by the old ones thus being able to involve many consequences on the whole of direct or indirect actors
Keywords: Data, Integrity rules, DBMS

Keywords

Data, Integrity Rule Violations,Entity Integrity, Referential Integrity, DBMS

1. INTRODUCTION

Integrity constraints are rules that lay down the properties that the fields of the database must respect (the fields correspond to the attributes of the relational model) in order to guarantee their consistency, their relevance and their validity [2]. These constraints therefore represent properties that must be satisfied by all instances of a database. The first research works in this field proposed the expression of the constraints by formulas of the first order logic, but so that the management systems of databases can check them automatically in an effective way. These researches have moved towards the study of more restricted classes, commonly called dependencies. For example, saying that a student's number (StdNo) Table 1 is the key to a Student relationship is an integrity constraint that enforces the existence and uniqueness of StdNo values throughout the Student relationship.
Key constraints are a special case of more general constraints called functional dependencies. An example of a functional dependency that is not necessarily a key is the constraint that a student can only have one ID number. Include dependencies are another very important type of integrity constraint. For example, assuming that the Student relationship includes each student's Department DeptName and that a Department Table 3 stores information for each university Department, an inclusion constraint might dictate that any DeptID value stored in Student Table 1 must exist in Department Table 3. In this way the information system remains consistent and in this respect the integrity constraints are an essential aspect of data quality. Additionally, these constraints ensure that no operation violates the integrity between data elements.
In these DBMS's, referential integrity constraints are not provided because these were not designed to maintain relationships between data, in part due to the denormalized and decentralized nature of data storage [5]. Instead, the responsibility for maintaining referential integrity is delegated to the application. However, such an approach involves a significant additional cost for the applications because they must themselves integrate the integrity validation repository. Not to mention that these DBMSs commonly handle large amounts of interconnected, dependent, and widely replicated data that make validation more difficult. Therefore, it becomes a critical and problematic element for applications to manage such data where dependencies must be properly maintained and maintained.
The integrity rules that guarantee an essential level of validity for the rows stored in the tables can be violated in our various database constructions on a daily basis, which can lead to enormous consequences and failure of our applications [6]. Consider a university with a database linked to an application allowing the management of these students as well as all the staff. If we take each row of the registration table which contains the number of a valid Student and by violation of the rules of integrity (entity integrity and referential integrity) in the database some registrations may be meaningless

which can cause a Student to be refused registration because non-existing Students have taken their place.

2. LITERATURE REVIEW

The consistency of the database is preserved by imposing integrity constraints on this interdependent data. The actions of updating the relational database can lead to violations of integrity rules that increase the reliability of data by defining field properties, linking tables, and applying data integrity rules. This problem of database integrity rule violation alerts various researchers including San et al. [6]. from the University of Computer Studies and Kyaing Tong who proposes the implementation of an integrity constraint checker for healthcare using the Constraint Planning Algorithm (CPA). Whenever a table update declaration event in a database is initiated, a consistency check is first performed by the CPA. Only if there is no constraint violation the update statement is performed. Otherwise, the update instruction is rejected.
As a rule, in relational DBMS's, individual records (fields and cells) are not specially protected, although there are known examples of practice when this is necessary. Therefore, in order to protect the database against the violation of the consistency of the data stored in it, researchers such as Yesin et al. [7] propose two methods that guarantee the integrity of data in databases by combining the Clark-Wilson model and the Universal Basis of Relations (UBR). The Clark-Wilson integrity model is based on triples: "The subject transaction does not violate the integrity object." Subjects, according to this model, do not have direct access to objects. Objects can only be accessed via the transformation procedure(TP). TPs are the only procedures allowed to modify a constrained data element whose integrity is checked by a VPI verification procedure (Integrity Verification Procedure). VPI is a procedure that scans data elements and confirms their integrity. Data whose integrity is not checked by the security model is called unconstrained data elements (UDIs). UBR shows a schema for using declarative and procedural support techniques for constraints such as entity integrity, referential integrity, required (non-zero) data, and domain constraints to ensure data integrity in the database.

3. DATABASE DATA MODELS

Data in a database is organized as one or more tables. A table contains a collection of rows stored on an external medium, usually a disc. A line is itself a sequence of (one or) several values, each of a specific type. Generally, a line gathers information about an object, an individual, an event, etc., i.e. a real-world concept (external to computing), which we will sometimes call an entity or fac [1].
The Table 1 below shows nine rows, each describing a Student's data. There are nine values representing respectively StdNo for Student Number, StdFirstName for Student First Name, StdLastName for Student Last Name, StdCity for Student City, StdState for Student State, StdPinCod for Student Pin Code, DeptID for Student Department ID, StdSection for Student Section, and StdGPA for Student CGPA. One of these lines represents the following fact: a student named GOSSE CARMEN EMMANUELLA, residing in AP State, VISAKPATNUM, 10520 Pin Code, from the Department ID 5 which corresponds to the FINANCE in the Department Table 3, Section E, and CGPA 9. All rows in a table have the same format or structure. This property means that in the Student Table, all rows consist of a value StdNo, StdFirstName, StdLastName, StdCity, StdState, StdPinCod, DeptID, StdSection and StdGPA, the order of the lines is irrelevant. The set of values of the same type corresponding to the same property of the described entities is called a table column. For example the StdFirstName column or the StdSate column of the Student table 1. Rows can be added to and deleted from a table. It is also possible to modify the value of a column of a line, or more generally of a subset of the lines.

4. ROLES OF A COLUMN IN A DATABASE

A row is assumed to group information about a real-world entity or fact. In this view, the value of a column represents a property of that entity. However, not all columns play the same role with respect to the entities represented by the rows of a table. A more precise analysis of the example in the figure below reveals three types of columns in this database, according to the role they play in it.

4.1 Identifiers (Primary Keys)

This column is used to identify an entity, and therefore also the row that represents it in the table. This is the case of OfferNo for the rows of the Offering Table 2 and OfferNo + StdNo (an identifier is a column or a combination of columns with unique values in each row) for those of the Enrollment Table 4. Such a column is called the table identifier. Declaring OfferNo to be the identifier of the Offering Table means that at all times the rows of this table must have values distinct from OfferNo.

4.2 Foreign Keys

A column of the second type is a copy of another table's identifier. Each of its values acts as a reference to a row of this table. This will be called a reference column, or in standard terminology, a foreign key. The Enrollment Table 4 contains two foreign keys: OfferNo, which is a reference to an offering entity (and therefore also to an Offering Table 2 row), and StdNo, which is a reference to a student (and also to a Student Table 1 row). It is by foreign keys that we can relate rows in separate tables. For each enrollment, represented by an Enrollment line, it is possible to know the information concerning the offering (via OfferNo) and those concerning a student (via StdNo). Note that the name of a column forming a foreign key is independent of that of the identifier it references. It will often happen that a foreign key is given the name of the target identifier (OfferNo in Offering Table), but it could also be given that of the referenced table (CourseNo).

4.3 Additional information

The third type of column plays no other role than that of providing additional information on the entity. This is the case with OffYear, OffLocation, OffTime and OffDays 2. From these basic elements, we will again highlight four important concepts: identifiers and multi-component foreign keys, identifiers primary, referential constraints and optional columns.

4.4 Multi-component identifiers and foreign keys

Nothing prevents the identifier of a table from being made up of more than one column. We could thus impose that the columns (OfferNo, StdNo) form the identifier of the Enrollment Table 4. This property amounts to saying that only one enrollment is recorded per offering for a specific offer, or that an offer can only be the subject of a single offering from a specific student. Consequently, a foreign key that references a table having a multi-component primary identifier is itself multi-component.

Table 1. : Student

StdNo	StdFirstName	StdLastName	StdCity	StdState	StdPinCod	DeptID	StdSection	StdGPA
120-56-7890	KPAZARA	MARIE-CLAUDE	VIJAYAWADA	AP	72000	6	A	9.00
124-56-7890	DAS	AMALEENA	VISAKPATNAM	AP	82100	2	E	9.5
234-56-7890	KUMWENE	MISHECK	VIJAYAWADA	AP	99042	1	B	9.20
250-56-7890	COULIBALY	MADIARA	VIJAYAWADA	AP	19000	4	C	8.9
300-56-7890	KUMAR	RISHU	VIJAYAWADA	AP	20052	5	A	9.7
320-56-7890	KOUASSI	KOFFI EVARISTE	VISAKPATNAM	AP	21002	3	B	8.5
350-56-7890	GOSSE	CARMEN EMMANUELLA	VISAKPATNUM	AP	10520	5	E	9.00

Table 2. : Offering

OfferNo	CourseNo	OffTerm	OffYear	OffLocation	OffTime	FacNo	OffDays
1111	CR001	SUMMER	2021	VIJAY302	10:30 AM		MWF
1234	CR003	FALL	2022	VIJAY302	10:30 AM	098-76-5432	MTW
4321	CR005	FALL	2021	VIJAY214	3:30 PM	098-76-5432	TTHS

4.5 Primary identifiers

Nothing prohibits imposing more than one identifier on a table either. For example, a table that contains the identifying information of a population with social security coverage could include, among other things, a REGISTRATION-NUMBER column and an IDENTITY-CARD-NUMBER column, each constituting an identifier. Among the identifiers of a table, one is chosen as the most representative. It will be declared primary identifier, the others being the secondary identifiers of the table. Every table has a primary identifier and any number (possibly zero) of secondary identifiers. This has the important consequence that the rows of a table are distinct, in accordance with the definition of the contents of a table as a set of rows. Note on identifiers. Any set of columns that includes an identifier is still an identifier: (StdNo, StdFirstName) 1 is a Student identifier. An identifier from which no component can be removed without losing its quality as an identifier is called a minimal identifier. It is obvious that we will seek to define only minimal identifiers. Note on foreign keys. Although, theoretically, the target of a foreign key is one of the identifiers of the referenced table, we agree to limit ourselves to the primary identifier.

4.6 Referential constraints

A value of a foreign key consisting of column C of a table (or, given the existence of multi-component identifiers, consisting of a group of reference columns) is intended to designate a row of a table T1. Concretely, we can impose that for any value of C in T2, there is a row of the table T1 identified by this value. We deduce a very important property, called referential constraint. This stipulates that the set of values of a foreign key is at all times part of the set of values of the primary identifier of the referenced table. For example, we would impose that any value of OfferNo in Enrollment Table be present in the OfferNo column of Offering Table (which we will note Enrollment.OfferNo Subset of Offering.OfferNo), and that any value of StdNo in Enrollment Table be found as the StdNo value of the Student Table. It is therefore forbidden to introduce a row such as (OfferNo: 2520, StdNo: 2022-12-0021, EnrGrade: 3.2) into the Enrollment Table, since there is no row in Offering whose primary identifier OfferNo has the value 2520.

5. INTEGRITY RULES (CONSTRAINTS)

The structural properties (identifier, referential constraint, mandatory/optional column) associated with the data must be respected at all times; they therefore constitute constraints imposed on the operations of modification of this data. Adding a row, deleting a row or changing the value of a column in a row are operations that are only allowed if these properties are still respected by the data after the operation [4]. If these properties are violated, we say that the data has lost its integrity. These properties are therefore integrity constraints. Let us briefly examine the impact of these constraints on operations to modify the contents of a database.

5.1 Entity Integrity

Entity Integrity Figure 1 means that each table must have a primary key or identifier . A primary key cannot have NULL values in any row. Entity integrity ensures entities, people, objects, places, and events are uniquely identified in a database. For auditing, security, and communication reasons, business entities must be easily traceable and unique [5]. Examples:

– Rows in the Student Table are uniquely identified by StdNo.
– Rows in the Offering Table are uniquely identified by OfferNo.
– Rows in the Enrollment Table are uniquely identified by the combination of StdNo and OfferNo.

5.2 Referential Integrity

Referential Integrity Figure 2 means that the values of columns in a table must match the values of columns in a related table. Referential integrity ensures that a database contains valid connections [3]. Examples:

– Enrollment.StdNo references a valid StdNo value in the Student Table.
– Enrollment.OfferNo refers to a valid OfferNo in the Offering Table.

6. INTEGRITY RULE VIOLATIONS

Integrity constraint violations occur when an insert, update, or delete statement violates a primary key, foreign key, check, or unique constraint or a unique index.

Table 3. : Department

DeptID	DeptName	StdNo	HodDept	Location
1	CSA	234-56-7890	A001	———————-
2	CSE	124-56-7890	A002	———————-
3	BUSINESS	320-56-7890	A003	———————-
4	PHARMACY	250-56-7890	A004	———————-
5	FINANCE	350-56-7890	A005	———————-
6	LAW	120-56-7890	A006	———————-

Table 4. : Enrollment

OfferNo	StdNo	EnrGrade
1234	350-56-7890	3.3
1234	234-56-7890	3.5
4321	300-56-7890	3.5
4321	124-56-7890	3.2

6.1 Data Modification

Modifying the model of a database most often involves modifying data. Like, adding a column to a table containing rows is followed by modifying that column for each of the rows (setting to null or the default value). Or, the deletion of a table is preceded by the deletion of each of its rows. It is obvious that these modification operations must respect the integrity constraints at the time of their execution [3]. Here are the most common cases:

6.1.1 Adding a column to a table. If the column is optional, the operation is unconstrained. If it is mandatory (not null), then the table must be empty or the column must be accompanied by a default value (default).

6.1.2 Deleting a column from a table. This column cannot be used in the composition of an identifier or a foreign key. If necessary, these must first be removed.

6.1.3 Deleting a table. If the table is referenced by a foreign key, then a delete operation is applied to each of its rows before it is deleted. This deletion can therefore be refused if the delete mode of this key is no action and if the referencing table contains at least row whose foreign key has a non-null value.

6.1.4 Adding an identifier to a table. If the table is not empty, the rows must respect the uniqueness constraint.

6.1.5 Deleting an identifier in a table. This deletion is not subject to conditions on the data. However, this identifier must not be referenced by a foreign key.

6.1.6 Adding a foreign key to a table. If the table is not empty, the rows must respect the referential constraint.

6.1.7 Other operations. On the other hand, the addition of a table, the addition and deletion of a non-identifying index as well as the deletion of a foreign key are not subject to conditions.

6.2 Common Threats of Integrity Rule Violations

Different configuration errors in our database management software, vulnerabilities, negligence or incorrect use can lead to violations of the integrity of our data. The most common types or causes of database-related security attacks are:

6.2.1 Internal errors: . An insider threat, or insider trading, is a security threat from any of the following three sources that has privileged access to the database:

(1) A person inside or with access to inside information, with malicious intent.
(2) Someone inside who makes mistakes that make the database vulnerable to attack.
(3) An insider threat, or insider trading, is a security threat from any of the following three sources that has privileged access to the database.

Insider threats are among the most common causes of database security breaches. They are often caused by too many employees having privileged user access rights.

6.2.2 Human error. Crashes, weak passwords, password sharing and other reckless or misinformation behaviours continue to be the cause of nearly half of all reported data breaches(49%).

7. RESULTS AND DISCUSSIONS

In this section, three tables illustrated by Table 5: Student1, Table 6: Enrollment1 and Table 7: Offering1 with violating integrity constraints are implemented in order to show the consequences that a violation of these integrity constraints can cause.

7.1 Overview of Results

Looking at the Student1, Offering1 and Enrollment1 tables below, we can admit the following analyses.

(1) Missing value for primary key(identifier) StdNo in Student1 Table (6th row).
(2) Orphan row in Enrollment1 Table: 5th row with OfferNo 8000.
(3) Missing value of a part of a primary key StdNo (identifier) for Enrollment1 Table (6th row).

Regarding the analysis of the Student1, Enrollment1 and Offering1 Tables below, it can be seen that all the two major rules (Entity Integrity and Referential Integrity) that guarantee an essential level of validity for the rows stored in the tables have been violated and this violation lead to the presence of unregistered data that make the database vulnerable and insecure to insertion, deletion, modification by anyone.

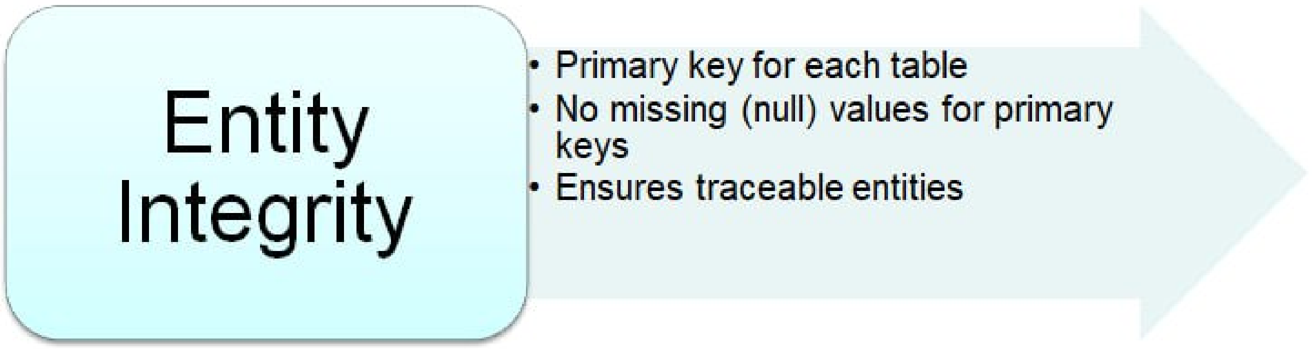
Entity Integrity
• Primary key for each table
• No missing (null) values for primary keys
• Ensures traceable entities

Fig. 1: Entity Integrity.

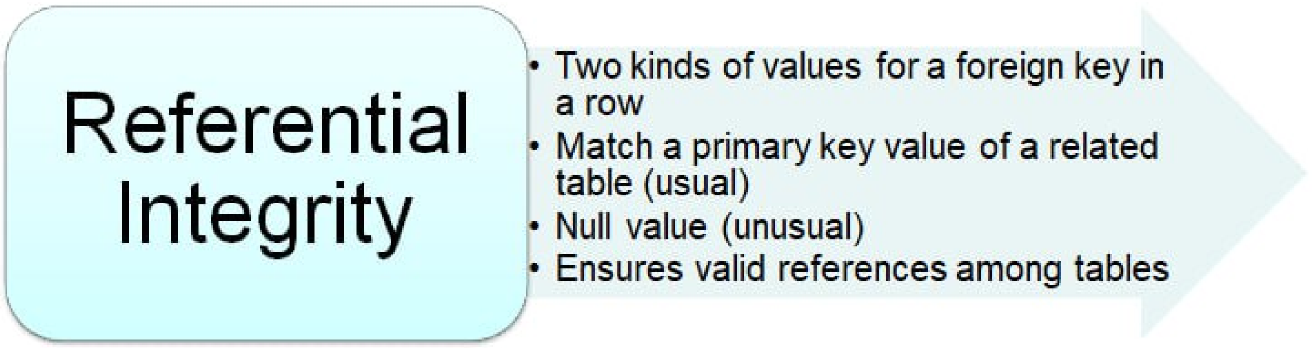
Referential Integrity
• Two kinds of values for a foreign key in a row
• Match a primary key value of a related table (usual)
• Null value (unusual)
• Ensures valid references among tables

Fig. 2: Referential Integrity.

Table 5. : Student1

StdNo	StdLastName
350-56-7890	CARMEN EMMANUELLA
234-56-7890	MISHECK
250-56-7890	MADIARA
—————-	YACE MICHELLE

Table 6. : Enrollment1

StdNo	OfferNo
124-56-7890	1234
250-56-7890	1234
124-56-7890	4321
234-56-7890	4321
250-56-7890	8000
—————-	4321

Table 7. : Offering1

OfferNo.	CourseNo
1234	CR003
4321	CR005

7.2 The Consequences Produced By The Violation Of The Integrity Rules Of A Database

By definition, a data breach is an inability to maintain the confidentiality of data in a database. The damage inflicted on your business by a data breach depends on a number of consequences or factors:

(1) Compromise of intellectual property: Your intellectual property - trade secrets, inventions, proprietary practices - often plays a crucial role in allowing you to maintain a competitive advantage in your market. If your intellectual property is stolen or disclosed, it may be difficult or impossible to retain or recover your competitiveness.
(2) Brand Reputation Damage: Your customers or partners may be reluctant to purchase your products or services (or do business with your company) if they believe you are unable to protect your own or their data.
(3) Business Continuity (or Business Continuity Failure): Some businesses cannot continue to operate until a breach is resolved.
(4) Costs of Remediating Breaches and Notifying Customers: In addition to the cost of communicating a breach to customers, the victim business must incur many other costs: digital forensic investigation and investigation costs, handling the crisis, triage, restoration of affected systems, and so on.

The consequences of database integrity rule violations are multiple and diverse depending on actions such as:

(1) Valid reference problems.
(2) Orders without customer or incorrect customer.
(3) Order without shipment.
(4) Missing reference values represent lines valid (Internet order without employee) or data entered later (offer with instructor reference later).
(5) Often we find two taxpayers or customers have the same government ID or customer ID, an order sent to a customer associated with the wrong order.
(6) The costs debited from our various credit cards without valid reason.
(7) The receipt of certain personal emails which are not intended for us.
(8) not registered who find themselves on the lists of electoral campaigns or certain universities.

8. CONCLUSION

This research has shown the importance of integrity constraints as an essential tool for ensuring the accuracy, consistency and security of data in a database. They help maintain data integrity by defining rules that the data must adhere to and are enforced by the database management system (DBMS). By using integrity constraints, database performance can be improved. This is because the DBMS can quickly validate data and reject invalid data, rather than having to process and then check for errors at the application level. Integrity constraints play a critical role in maintaining data integrity in a database and are an essential aspect of database management. They help ensure data consistency, data integrity, and error prevention. They also play an important role in improving database performance and security. Integrity constraints can be used to enforce business logic, validate data and improve data security. They also serve as self-documenting rules that help document the rules that govern data in the database.
In addition, integrity constraints also play a critical role in maintaining relationships between tables. Primary key, foreign key, unique constraints, and referential integrity constraints ensure that data is consistent and accurate. They also prevent errors from occurring during data entry or modification that cause consequences that violate the major integrity rules of the database. Violations of these integrity rules can be addressed through best practices by establishing appropriate rules and controls for accessing the database such as: Administrative controls that manage the installation, modification and configuration management of the database. Preventive controls that govern access, encryption, semantic marking and masking.Detection controls that monitor database activity and data loss prevention tools. These best practices identify abnormal, suspicious activity and issue alerts using the mechanisms provided by ADVANCED SQL queries (Triggers, Procedures, functions etc.) to maintain data integrity and consistency and avoid potential integrity constraint violations.

9. REFERENCES

[1] TALENT DEVELOPMENT CENTER, 2017 ,“Responding to Data Manipulation via Triggers,” in MICROSOFT SQL SERVER (MS-SQL-102).
[2] Raja, H. (2012). Referential Integrity in Cloud No-SQL Databases.
[3] Jean-Luc Hainaut, 2000 ,4e Edition, “BASES DE DONNÉES ET MODÈLES DE CALCUL”.
[4] Michael Mannino, 2nd Edition,” DATABASE MANAGEMENT ESSENTIALS”,EISBN13: 9780983332473.
[5] Michael Mannino, 8e Edition, 2022,” Database Design, Query Formulation, and Administration”, EISBN13: 9781948426947.
[6] San, K. K. Implementation of Integrity Constraint Checker for Healthcare Database System (Doctoral dissertation, MERAL Portal).
[7] Yesin, V., Karpinski, M., Yesina, M., Vilihura, V., & Warwas, K. (2021). Ensuring data integrity in databases with the universal basis of relations. Applied Sciences, 11(18), 8781.