APPLYING SQL DATABASE QUERY TO ACCESS SQL SERVER
2019 – VISUAL STUDIO 2019

Vladimir Šimović1,*, Matija Varga2

1 The University of Applied Sciences Baltazar Zaprešić, Vladimira Novaka 23, 10290 Zapresic, Croatia
2 University NORTH, Trg Žarka Dolinara 1, 48000 Koprivnica, Croatia

*Corresponding author:
E-mail address: vladimir.simovic@bak.hr

ABSTRACT: The paper presents: relational data model, which is the basis for the development of relational database management system (RDBMS) and the development of query languages for working with databases, relational schema, complex SQL queries and subqueries and the results obtained based on queries in relations. MS SQL Server 2019 accesses data in the database and can extend MS SQL Server 2019 application modules (eg. MS Visual Studio) that are created as a graphical interface for end users, with the addition of new capabilities that will enhance the execution of processes for participants within the company and other application modules of the organization. Each organization’s database contains information of great importance, so queries and backups will be made using relational algebra operators and warned of potential threats and dangers of database corruption.

Key words: RDBMS, relational data model, relational schema, SQL, data manipulation language, dana management, relational algebra operators, database protection

INTRODUCTION

Working with the database is about retrieving data, adding data, modifying, deleting and preserving the integrity of the data, backing up the database, allowing simultaneous access to the database for multiple users, verifying the identity of users, etc. The database in Croatian has been created entirely by author named: "Applying SQL Complex Queries to Access Data in a Database Using MS SQL Server 2019".

The goals of the paper are: (1) to show certain capabilities of MS SQL Server 2019, (2) to show ways of querying using aggregate functions, (3) to show how to set nested queries i.e. subqueries, (4) to warn about possible database corruption, (5) present the ability to collect data from multiple relationships, (6) show and explain, by way of example, basic concepts in databases such as relation, relational schema, RMP model, etc., (7) present more complex queries to quickly find specific data in a database, to (8) demonstrate how to delete data on MS SQL Server 2019, (9) explain ways to protect data from unauthorized and simultaneous access to data.
RELATIONAL DATA MODEL

The relational data model forms the basis for the development of the RDBMS¹ and the development of query languages for working with databases. RDM (Relational Data Model) is a solid theoretical apparatus that has a dominant influence on the further development of theory and practice in the field of databases. The RMP was constructed based on the EVA² model [4]. Also, model creation represents the second stage in data modelling. RDM is created automatically so that each object type from the ELA model is converted to a RDM relation. The object properties become attributes of the relational schema, and the primary object key becomes the primary relational key. An attribute in a relational schema is called a foreign or foreign key if it has (has) a primary key function in another relational schema. The relational model shows everything that the ELA model does.

SCOPE OF PROJECTS

The scope of the project depends on three key elements: quality to be delivered, time to be spent and resources to be engaged. These three key elements are the basic pillars of project management. Project quality can be defined as a series of activities that must be carried out for a project to meet agreed specifications. Specifically, the keyword here is an activity for the simple reason that quality control enters the activity. When planning, activities that are part of the quality assurance system should be envisaged. If these activities are neglected, the quality may be unsatisfactory.

The time it takes to complete project activities is defined in the planning process. However, it is in this segment that there can be gross omissions. In such a case, the recovery of the project can be problematic.

Resources are not just about money; this segment is far broader and includes other hard-to-define moments in addition to financial moments. Knowledge and experience are typical project resources that are difficult to define. They are abstract in themselves, but the members of the project team in charge of carrying out the project are the bearers of knowledge and experience.

All told so far applies to software projects, especially in the area of human resource management. The software is mainly a product of the experience and knowledge of the project participants. On the other hand, when the software becomes too complex then the division of work can take place.

RELATIONAL SCHEME

A relational schema is a data model based on set theory that represents the form by which data is stored. Each relational scheme has meaning and content [2].

---

¹ RDBMS – Relational Database Management System.
² ELA – Entities, Links, and Attributes.
Figure 1 shows the relational schemas. In the lower part of the image, the ‘Results’ option contains the attributes of relations, that is, the set of attributes that make up the domain elements. Figure 1 also shows the relational schema and the queries invoking the relational schema and showing its attributes. Such a data model is based on set theory. For example, we can denote a relation by $R$, while a set of attributes can be denoted by $A$. $R(A)$ is a set of attributes of a relation $R$. Example: $R(A_1, A_2, A_3, \ldots, A_n)$ be a relational scheme and let each attribute $A_i$ be an element of the relation $R$ that can represent a domain (depending on which level we are observing), given by the corresponding domain $D_i$. Let $D$ be the union of all domains $D_i$ ($i = 1, 2, \ldots, n$). The relation $R$ on the relational scheme is a finite set of $n$-vertices obtained by mapping from $R$ to $D$ with the constraint that for every $n$-torque value to which the attribute $A_i$ is mapped, there must be an element of domain $D_i$ ($i = 1, 2, \ldots, n$). The set of attributes that participate in the formation of key candidates are called key attributes, because they are key in defining relationships, and other attributes can be said to be non-key attributes [4].

RELATIONAL DATABASE MANAGEMENT SYSTEM – RDBMS

A ‘Relational Database Management System’ (RDBMS) is a computer system with software to create, update, search and maintain a database whose structure is described by a relational model. The properties of RDBMS are: functionalities of RDBMS, which provides easier administration of data, provides a better understanding of data integrity requirements, enables a more efficient data redundancy control process, supports more complex database lookup procedures, enables data relationships to be displayed. In fact, RDBMS has a rigorous mathematical basis established on the concept of relation and set theory, etc. [4]. The relational system in use here is a database management system called ‘MS SQL Server 2019’.

---

3 Isti podatci koji su pohranjeni više puta (zalihost). Nastoji se izbjeći.
4 Relational Database Management System” = “RDBMS”.
Figure 2. View of the RDM (Relational Data Model) in ‘MS SQL Server 2019’
(Source: Authors)

Figure 2 shows the ‘Relational Data Model’ (RDM) and database entry commands. This is a data model because the data for each attribute within the created relation is displayed. Figure 2 also shows the relational database. Relational databases consist of at least two tables or relations. Five of the seven relationships can be seen from Figure 2. From all relationships, we can print or call data using queries contained in the database. The relational schemas shown show that attributes represent the domain (depending on which level we are looking at).

STANDARD STRUCTURED QUERY LANGUAGE (SQL)

The abbreviation SQL means ‘structured query language’. SQL (Structured Query Language) is a commercial query language for working with relational databases. It is widely known that SQL was developed by IBM [8]. SQL is the most popular, well-known and commonly used query language. SQL is used to manipulate and manage the data contained in the database. The SQL language was created in the 1970s in the development of the ‘System R’ development research project within IBM [8]. The project manager and chief designer of SQL was Donald Chamberlin. The language was gradually refined, and its refined version appeared in today’s IBM [8] RDBMS called DB2. In the spread of SQL during the 1980s, Oracle Corporation, a software company that incorporated SQL into its DBMS, played an important role, making it accessible and popular on all major computer platforms. Other DBMS vendors, such as Digital Equipment Corporation (DEC) and Informix Inc., have closed acceptable SQL on the market and abandoned their own languages [3]. The relational database provided the greatest flexibility in decomposition data, which provides a major advantage in scheduling the distribution of data across individual system nodes. They can divide relationships vertically and horizontally so that they can also be reunited by joining operations.

For example, ‘Customer’ relationship (in Figure 1) has a set of attributes: ID1, Name, Address, Giro_Account, Telephone_number. The relational schema and attribute set can be
displayed as an example: Customer(ID, Name, Address, Giro_Account, Phone_Number). No matter what the attributes in the 2019 MS SQL Server are sorted in the order they are in the ELA model, in the relational schema the order of the attributes does not matter. In MS SQL Server 2019 they are arranged in this way for the sake of transparency. Example of a query to call the displayed relationships by name Buyer, Seller, Document_Label, Goods_Services, Inventory, Other_Document_Part, Document_Item in Figure 1.

COMPLEX QUERIES AND SUBQUERIES

A simple query is set with the "select" command, with the help of which we can get information from a specific table (relation). A simple subquery is set with the 'SELECT' command which is nested within the commands SELECT, SELECT ... INTO, INSERT ... INTO, DELETE, UPDATE, or within other subquery. In the subquery, we use the SELECT command to derive a set of specific values to evaluate in a WHERE or HAVING condition statement [6]. Encoding the first complex query (below, in Croatian) the relation with the following data was obtained (Figure 3): the name of the goods and the amount of the total price with VAT for all the goods purchased (pipes and counters):

```
select Naziv_Robe_i_Usluga, (sum(st.Kolicina * ru.Cijena)*23)/100+sum(st.Kolicina * ru.Cijena) as 'Iznos' from Stavka_Dokumenta st,Robe_Usluge ru
where st.Broj_Robe_Usluge2=ru.Broj_Robe_Usluge group by Naziv_Robe_i_Usluga
```

![Figure 3. View of the relation obtained after the first complex query is set (Source: Authors)](image)

The name of the goods and services and the amount represent the attributes of the displayed relation that we can view as a domain with the elements: counters 298.33 and pipes 87.26. When asked for a second complex query (below, in Croatian), information about the buyer's name, address, buyer's phone number and seller's name, seller's address and seller's phone number from the buyer and seller relations was requested.

```
```

---

5 In Croatian: Kupac(ID, Naziv, Adresa, Žiroaračun, Telefonski_broj).
6 In Croatian: Kupac, Prodavatelj, Oznaka_dokumenta, Robe_Usluge, Zalihe, Ostali_Dio_Dokumenta, Stavka_Dokumenta.
With the third complex query (below, in Croatian), the price of material per unit of product above HRK 30 was requested and obtained using comparison operators: greater than (>) and equal (\(=\)) (Figure 5).

```sql
select Naziv_Robe_i_Usluga, Cijena as 'Cijena po jedinici proizvoda' from Robe_Usluge where Cijena >= 30
```

**Figure 5.** View of the relation obtained after the third complex query is set (Source: Authors)

With the set fourth complex query (below, in Croatian), the price information of all products beginning with the letter R was requested and obtained (Figure 6).

```sql
select Naziv_Robe_i_Usluga, Cijena as 'Naziv robe ili usluga' from Robe_Usluge where Naziv_Robe_i_Usluga like 'R%'
```

**Figure 6.** View of the relational representation of the price information of all products beginning with the letter R obtained after the fourth complex query is set (Source: Authors)

With the set of the fifth complex query (below, in Croatian), the price information of the horizontal pressure gauge 0-10 bar, connection "\(\frac{1}{4}\)" was requested and obtained (Figure 7).
select Naziv_Robe_i_Usluga as 'Naziv robe ili usluga', Cijena from Robe_Usluge where Naziv_Robe_i_Usluga like 'MANOMETAR HORIZONTALNI 0-10 BARA PRIKLJUČAK ¼'''

**Figure 7.** View of the relational representation of the price information of the horizontal pressure gauge 0-10 bar, connection “¼” obtained after the fifth complex query is set
(Source: Authors)

With the sixth complex query (below, in Croatian), the date of delivery and the name of the product delivered (the pipe) were requested and obtained (Figure 8).

```sql
select Rok_isporuke,Isporuka_Robe_Na_Naslov from Ostali_Dio_Dokumenta where Rok_isporuke = '2009-03-20'
```

**Figure 8.** View of the relational representation of the date of delivery and the name of the product delivered (the pipe) obtained after the sixth complex query is set (Source: Authors)

With the seventh complex query (below, in Croatian), information on the name of the goods and the amount of the price of the goods excluding VAT were requested and obtained (Figure 9). The query uses the HAVING operator, which is used for filtering, that is, it limits the groups that appear in the query. The HAVING condition has similarities to the WHERE condition. WHERE comes usually before GROUP BY, while HAVING comes after GROUP BY and executes after grouping records.

```sql
select Naziv_Robe_i_Usluga, sum(st.Kolicina * ru.Cijena)as 'Iznos' from Stavka_Dokumenta st,Robe_Usluge ru
where st.Broj_Robe_Usluge2=ru.Broj_Robe_Usluge group by Naziv_Robe_i_Usluga
having sum(st.Kolicina*ru.Cijena)>45.23
```
In SQL, commands are often used to delete data, such as (below, in Croatian):

`delete from Stavka_Dokumenta where Cijena=23.64`

The command deletes all the quoted prices from the relation named `Item_Document` (in Croatian: `Stavka_Dokumenta`), i.e. it destroys the information that the material price is 23.64 HRK.

With the eighth complex query (below, in Croatian), average data calculated on the basis of the prices of all products recorded in the database were requested and obtained (Figure 10). AVG's function is a commonly known function for calculating averages. Numerous authors in addition to the AVG function and the min, max, count and sum sums are aggregate functions. So instead of AVG it is possible to combine in queries the basic functions: min, max, count and sum.

```sql
select avg(cast(Cijena as real)) as Realna_prosjecna_cijena_svih_cijena_u_skupu from Robe_Usluge
```

**Figure 10.** View of the relational representation of the query results for the average price calculated based on all the prices entered for each product obtained after the eighth complex query is set (Source: Authors)

With the ninth complex query (below, in Croatian), information on the name of the goods and the price per unit of product were requested and obtained, but only at the price other than HRK 23.64 (Figure 11). The operator `!=` other than. Was used for this query. Other comparison operators that can be combined in this type of query are: equal (=), greater than (>), less than (<), equal to or greater than (=>), equal to or less than (=<), different from (<>), not greater than (!=>), not less than (!<).
select Naziv_Robe_i_Usluga,Cijena as 'Cijena po jedinici proizvoda' from Robe_Usluge where Cijena != 23.64.

**Figure 11.** View of the relational representation of the results for query requesting information on the name of the goods and a price different from 23.64 obtained after the ninth complex query is set (Source: Authors)

With the 10th complex query (below, in Croatian), information on the name and price of the goods was requested and obtained, costing between 25 HRK and 208 HRK (Figure 12). The **between** operator specifies the ranges we are testing, and it is combined with the logical operator **and**.

select ru.Naziv_Robe_i_Usluga,ru.Cijena from Robe_Usluge ru where ru.Cijena between 25 and 208

**Figure 12.** View of the relational representation of the information on the name and price of the goods costing between 25 HRK and 208 HRK (operator **between** was combined with the logical operator **and**) obtained after the 10th complex query is set (Source: Authors)

With the eleventh complex query (below, in Croatian), the requested and obtained information about the customer’s name and customer’s address are sorted by descending by name. The query defines that the user of RDBMS wants only the first five customers, which is defined by the **top** command and the number (**value**) 5 (Figure 13).

select top 5 Naziv, Adresa FROM Kupac ORDER BY Naziv desc
Figure 13. View of the relational representation of the information about the customer’s name and customer’s address (which were sorted by descending by name) and only for the first five customers obtained after the 11th complex query is set (Source: Authors)

When asked for the twelfth complex query (below, in Croatian), the information about the customers which left their phone number was requested and obtained, and their number was entered into the database (Figure 14). The query used the where condition. In where condition, aggregate values or groups cannot be referenced. Where condition applies to single rows.

1st example: SELECT * FROM Kupac WHERE Telefonski_broj <> " AND Telefonski_broj != 'nema';

2nd example: SELECT * FROM Kupac WHERE NOT (Telefonski_broj=" OR Telefonski_broj = 'nema');

Figure 14. View of the relational representation of the information about the customers which left their phone number obtained after the 12th complex query is set (Source: Authors)

With setting up the thirteenth complex query (below, in Croatian), the values of the two attributes were combined into one column (Figure 15).

select Naziv+’ ’+Adresa as 'Naziv tvrtke i adresa' from Kupac
The fourteenth complex query (below, in Croatian) shows cities that do not have the letter S in their name (Figure 16).

\[
\text{select * from Mjesto where Naziv not like 's%'}
\]

Figure 16. View of the relational representation of the information about the cities that do not have the letter S in their name obtained after the 14\textsuperscript{th} complex query is set (Source: Authors)

For the fifteenth complex query (below, in Croatian), the cities whose name does not begin with the letters V and Z are printed. The LIKE operator is used to print the results, which is used to compare strings, i.e. characters (Figure 17).

\[
\text{select * from Mjesto where Naziv like '^[^VZ]%'}
\]

\[
\text{select * from Mjesto where Naziv like '_s%'}
\]
Figure 17. View of the relational representation of the information about the cities whose name does not begin with the letters V and Z, and the cities which has the second letter S in the name and contains the first of any letters in the name obtained after the 15th complex query is set (Source: Authors).

For the sixteenth complex query (below, in Croatian), all buyers that do not have ID1 equal to 1, 3, 5, 7 are printed (Figure 18).

```sql
select * from Kupac where ID1 NOT IN (1,3,5,7)
```

Figure 18. View of the relational representation of the information about all buyers that do not have ID1 equal to 1, 3, 5, 7 obtained after the 16th complex query is set (Source: Authors).

When asked for the seventeenth complex query (below, in Croatian), all information about the company name, the address of the company headquarters, the delivered material and the method of dispatch of the goods are printed. The query used an internal join to retrieve records from multiple tables and to provide one set of records as the final query solution (Figure 19).

```sql
select Naziv as 'Naziv tvrtke',Adresa as 'Adresa sjedišta tvrtke', Isporuka_Robe_Na_Naslov as Isporuceni_Materijal, Nacin_otpreme from Kupac inner join Ostali_Dio_Dokumenta ON Kupac.ID1=Ostali_Dio_Dokumenta.Broj
```
Figure 19. View of the relational representation of all information about the company name, the address of the company headquarters, the delivered material and the method of dispatch of the goods obtained after the 17th complex query is set (Source: Authors)

With the eighteenth complex query (below, in Croatian), the number of the seller having a different name in the relation called the seller is printed. COUNT returns the number of elements in the group. COUNT returns an integer value, while DISTINCT considers different values (Figure 20).

```
select COUNT(DISTINCT Naziv) as Broj_prodavatelj from Prodavatelj
```

Figure 20. View of the relational representation of all information about the number of the seller having a different name in the relation called the seller obtained after the 18th complex query is set (Source: Authors)

When asked for the nineteenth complex query (below, in Croatian), the average grade data is printed as a real number. Grades from the fourth exam date, 2011, were averaged. The query used the aggregate function AVG (Figure 21).

```
select AVG(CAST(ocjena as float)) as Prosjek from MV_predmet
where DATEPART(D, ispitni_rok)=4
AND DATEPART(YY,ispitni_rok)=2011
```

Figure 21. View of the relational representation of all information about the average grade (as a real number) obtained after the 19th complex query is set (Source: Authors)
When asked for the 20th complex query (below, in Croatian), all information about the all courses which are completed and take more hours than the average time for teaching all the subjects individually (Figure 22).

```
select naziv as 'Kolegij', sati_predavanja from MV_predmet
where sati_predavanja > (select AVG(sati_predavanja) from MV_predmet)
```

**Figure 22.** View of the relational representation of all information about the all courses which are completed and take more hours than the average time for teaching all the subjects individually obtained after the 20th complex query is set (Source: Authors)

With the 21st complex query (below, in Croatian), all grades above the average grade and subject name are printed (Figure 23).

```
select ocjena, naziv from MV_predmet where ocjena > (select AVG(ocjena) from MV_predmet)
```

**Figure 23.** View of the relational representation of all grades above the average grade and subject name obtained after the 21st complex (sub)query is set (Source: Authors)

The 22nd complex query (below, in Croatian) shows the number of faculties with more than 15,000 students enrolled. The aggregate `count` function and the comparison operator more than (`>`) were used (Figure 24).

```
select count(*) from College
where enrollment > 15000;
```

[7]
Figure 24. View of the relational representation of the number of faculties with more than 15,000 students enrolled obtained after the 22\textsuperscript{nd} complex (sub)query is set (Source: Authors)

With the 23\textsuperscript{rd} complex query (below, in Croatian), the union operation was applied with the \textit{union} command (Figure 25).

\begin{verbatim}
select cName as name from College
union all
select sName as name from Student;
\end{verbatim}

Figure 25. View of the relational representation of the view operation (relational algebra) \textit{union} with attribute data \textit{sName} and \textit{sName} from College and Student relations obtained after the 23\textsuperscript{rd} complex (sub)query is set (Source: Authors)

The existing \textit{exists} operator was applied to the set 24\textsuperscript{th} complex query (below, in Croatian). Operator \textit{exists - true} if the subquery contains at least one record (Figure 26).

\begin{verbatim}
select sID, sName, sizeHS
from Student S1
where exists (select * from Student S2 where S2.sizeHS < S1.sizeHS);
\end{verbatim}
Figure 26. View of the relational representation of the view the subquery where the `exists` operator is used obtained after the 24th complex (sub)query is set (Source: Authors)

With the 25th complex subquery set (below, in Croatian), the data is written to a new relation that does not have data for the `sID` attribute the same as the records in the `Apply` relation (Figure 27).

```sql
select * from Student
where sID not in (select sID from Apply);
```

```sql
select * from Apply [7]
```

Figure 27. View of the relational representation of the view the subquery using the `not` and `in` operators obtained after the 25th complex (sub)query is set (Source: Authors)

---

7 *Not to negate other logical operators. True if the operand is in the default list of values.*
Figure 28 shows a file called “SradaMatijaVarga.sql” that created the database and shows the application of the DDL, DQL, DML and DCL structural languages\(^8\), and also shows the appearance of the “Visual Studio 2019” user interface.

**POSSIBLE CASE OF DAMAGE AND RENEWAL OF DATABASE**

Database corruption can occur for a variety of reasons, such as a failure of the computer hardware partition (disk and other storage media) or the occurrence of a system software error. The database may be damaged by malicious or accidental behavior. Regardless of the causes and causes of the damage, the database must return to a state of preserved physical integrity. In order for the database to be restored, it is necessary to first back up the data from the database (which must be protected by cryptographic encryption) to some medium outside the database. Therefore, the entire database is backed up to a separate medium periodically, and any changes to the database are recorded in the change log [5]. For business companies and other organizations, it is usually recommended that the backup of data from the database to the media be done on average every five days. But, backing up your data every five days is too rare. The question is what would happen if the big company remained without important information for five days. Any amount less than the loss of data would pay off to back up your data more often. It’s safer to back up your data every day. The recommendation for non-IT companies is to hire their own IT manager to handle the database administration, i.e. the database administrator, who can take care of the backups and be responsible for all data and information (or knowledge) in the database (which must be

\(^8\) According to source [9]: GeeksforGeeks. URL: https://www.geeksforgeeks.org/sql-ddl-dql-dml-dcl-tcl-commands/ (3/2/2020), “DDL, DQL, DML and DCL structural languages are abbreviation for SQL commands, which are mainly categorized into next four categories: DDL - Data Definition Language, DQL - Data Query Language, DML - Data Manipulation Language, DCL - Data Control Language.”
protected by cryptographic encryption). The “(Object) Relational Database Management Systems” (abbreviation: (O)RDBMS) and RDBMS in business companies (and other organizations) need to be available every minute, so backups are enabled for 24 hours (for every day, week and month during the whole year).

Backups can be made during operation. In order to prevent malicious intrusion of companies into the information system, it is necessary to change passwords minimally every month. Also, for each part of the application there must be a separate password and authority to access individual modules. Only certain employees have the right to access certain program modules. All passwords in business companies (and other organizations) should be set to contain a combination of uppercase and lowercase letters and numbers. To protect all the data in the database (which must be protected by cryptographic encryption), business companies (and other organizations) minimally use one (light) server (net) antivirus protection application programme and different (heavier) antivirus protection application programme for client computers (Personal Computer – PC, or notebook/laptop/tablet) and central firewall system, also. All antivirus software updates must be up-to-date, and central firewall system (also). Each time when computers (server and clients) are used, all system user must update the antivirus programmes. Also, disk scan is required at least once a week, i.e. a complete scan of the disk.

**PROTECTION OF THE DATABASE FROM UNAUTHORIZED ACCESS**

Databases must be protected against unauthorized access by malicious (physical and/or legal) persons. In addition to physical protection, there is also a software security method, in which certain software is embedded in some (O)RDBMS or RDBMS. It restricts the use of the database to users who have the ability to physically access the computer (or computer terminals), in fact to server and to client computers (PC, or notebook/laptop/tablet). There are several ways to protect the encrypted database: user identification, various security mechanisms, and authorizations. When it comes to protecting a database by identifying users, it is generally known that each user of the database has his (or her) *username* and *password*. The user must identify himself/herself with his/her *username* and *password* in some (O)RDBMS or RDBMS, thus proving his/her identity. If the user does not enter a *username* and *password* corresponding to the list in some (O)RDBMS or RDBMS, that system refuses to give his/her permission to operate the database.

Many database management systems can be managed over the Internet/intranet/extranet. Also, it is common knowledge that a database can be accessed from a remote computer (PC, or notebook/laptop/tablet) that can be located anywhere in the world. How to protect a database that can be accessed online (over the Internet/intranet/extranet)? We can temporarily protect it by turning off the network capabilities of the some (O)RDBMS or RDBMS, or to restrict options so that only local users (developers) can access the database (i.e. local network clients), and to restrict options so that only some external users may be allowed access to the database, but only with client identification, by using encrypted communication such as SSL/SSH, duplicate keys, etc. When it comes to *views*, the user can be *granted access* to only a certain part of the data, what means that other parts of the database are unavailable to a specific user.
The protection of the database by granting authority determines what the user can do with the database information at his (or her) disposal, and where security permissions can be: read/select, update, insert, and delete. The (O)RDBMS or RDBMS must “remember”, i.e. have a stored list of authorizations for each user and each relation from a specific view. If the user tries to perform an activity that is not authorized, the (O)RDBMS or RDBMS will not execute it, but will print a warning that the user is not authorized to perform the specified activity. In most cases, a developer named “Database Administrator” is concerned about database protection. The database administrator has a user list, provides view options, and regulates authorizations. Only the database administrator has the ability to regulate authority, which ordinary users do not have. In order to protect the database, it is necessary to properly monitor, analyse changes in the data by period and record the actions of the person who can access the database at the moment when he (or her) intentionally (or not) wishes to cause harm.

The (O)RDBMS or RDBMS can be vulnerable, also. The overall functionality of some (O)RDBMS or RDBMS can also be an indicator of its security. By backup (system) we can protect data in the database, also. Today, systems can be used to automatically back up and recover the system, providing the ability to back up (archived) data for up to 5 years and eliminate the costs caused by human errors associated with data storage. This kind of protection provided by systems to automatically back up and recover the system is better than backing up the entire database, because the system restores only the data that has been changed. The (O) RDBMS or RDBMS must provide and guarantee serial use (serial capability or batch execution)\(^9\), whereby serial capability is achieved by special control during the execution of transactions.

**CONCLUSIONS**

SQL query language (opposite to “Query by Example” (QBE) language) consists of English words that can be divided minimally into three groups, which are (three basic groups of SQL): DCL, DML, and DDL [1] [9], where DCL is a Data Control Language, DML is a Data Manipulation Language, DDL is a Data Definition Language for defining data, and is in some ways almost a fundamental part of these languages because it serves to define the structure of an object in a database without which the database would not exist [1] [9]. The manuals or papers entitled something like this (theme) “Applying SQL Complex Queries to Access Database Data Using MS SQL Server 2019” usually lists the most commonly used DML keywords, and the most commonly used SELECT query, where SELECT is used to retrieve rows from relationships to find the desired data. Also, the most commonly associated with the key word SELECT are these (sub)keywords: from, where, group by, having and order by. Also, usually at the end of these kind of manuals or papers are detail explanations about the controls over databases and mentions about the protection of the database against unauthorized access to the data, etc.

\(^9\) Have several transactions run concurrently in a multiuser database so that certain parts of those transactions are timed mixed, and if the ultimate effect of their execution is the same as if they were executed batch wise or sequentially, we say that it is a batch or serializable execution of transactions.
REFERENCES


951de6e94e2a70cmaasassetID=HA001231513&CorrelationId=10e79c01-2685-4c9a-9de7-4328db4c503&ui=hr-HR&rs=hr-HR&ad=HR. (3/2/2020).

