

Ministerul Educației al Republicii Moldova
Universitatea Tehnică a Moldovei
Facultatea Calculatoare, Informatică și Microelectronică
Ingineria Software

REPORT

Laboratory work Nr.4

SELECT Transact-SQL statement

Realized by: Popa Eugeniu
FAF-202

Chișinău, 2021

The theoretical part

1. What features does the SQL Transact-SQL Query Editor component offer?

- View data in text or grid form, using SQL commands.
- Analysis of the query execution plan.
- Online help of the Transact-SQL language.

2. What are DDL, DML, DCL and TCL?

DDL (data definition language) is a part of the SQL language that is used for definition data and objects of a data store. When the instructions in this group are used, in the respective data entries of the SQL Server system are made. With the help of these instructions there can be created tables and (or) modified the specified tables, inserted columns (fields) in tables, defined integrity constraints, constructed indexes, etc.

DML (data manipulation language) is a part of the SQL language that is used for data manipulation, consultation, insertion of lines (records) in tables, exclusion of lines and line modification.

DCL (data control language) is used to create roles, permissions and their integrity referential. It is also used to manage access to data stores and resources server.

TCL (Transactional Control Language), is used to manage transactions that take place in a database.

3. Which operators offer the Transact SQL language?

Operatorii utilizați în construirea expresiilor Transact-SQL se pot clasifica astfel:

Operatori aritmetici	Reprezentare	Operatori de comparare	Reprezentare
Împărțirea	/	Diferit de	<>
Modulul	%	Egal cu	=
Înmulțirea	*	Mai mare sau egal cu	>=
Adunarea	+	Mai mare ca	>
Scăderea	-	Mai mic sau egal cu	<=
		Mai mic ca	<
Operatori de caracter	Reprezentare	Nu e egal cu	!=
Concatenarea	+	Nu e mai mare ca	!>
		Nu e mai mic ca	!<

Operatori logici	Reprezentare
Adevărat, dacă pentru orice element din mulțimea definită de expresie este adevărat	ALL
Adevărat, în cazul când operandul este din intervalul fixat	BETWEEN
Adevărat, în cazul când o subinterogare extrage cel puțin un rând	EXISTS
Adevărat, când un element se găsește într-o mulțime de elemente	IN
Adevărat, în cazul când operandul găsește un element ce se potrivește cu un model	LIKE
Inversează valoarea booleană	NOT
Adevărat, în cazul când vreun element din mulțimea definită de expresie are valoarea adevărat.	SOME
Adevărat, în cazul când vreun element din mulțimea definită de expresie are valoarea adevărat.	ANY
Adevărat, dacă ambele expresii în structura instrucțiunii au valoarea adevărat	AND
Adevărat dacă cel puțin una din expresiile constante în structura comenzilor are valoarea adevărat	OR

Operatori la nivel de biți	Reprezentare
Operatorul <i>AND</i> pe biți	&
Operatorul <i>NOT</i> pe biți	~
Operatorul <i>OR</i> pe biți	
Operatorul exclusive <i>OR</i> pe biți	^

Operatorul *LIKE*, compară datele cu un model de tip caracter:

Caracter generic	Descriere operator <i>LIKE</i>
	Un singur caracter oarecare
%	Orice secvență de zero sau mai multe caractere
[ab...]	Un caracter din lista <i>ab...</i>
[a-z]	Un caracter din intervalul <i>a-z</i>
[^ab...]	Un caracter din afara listei sau din afara intervalului specificat.

4. What is the basic syntax of the SELECT statement?

```
SELECT [ALL|DISTINCT] <column list>
      FROM <table list>
      [WHERE <search condition>] [<second_clauses>]
```

5. What functions does the Transact SQL language offer (including aggregation functions)?

A function in the Transact-SQL language is nothing but a routine that performs a specific operation and returns a result. Functions are an important feature of language and are used to:

- perform calculations on data
- change data
- manipulate groups of records
- change the data format
- convert different types of data

There are several types of functions on a single line: character, numeric, data, conversion, etc.

For example, the most important character functions are:

UPPER (<expression>)

UPPER converts alpha characters from lowercase to uppercase.

LOWER (<expression>)

LOWER converts alpha characters from uppercase to lowercase.

SUBSTRING (<expression>, m [, n])

SUBSTRING returns a string of n characters starting with the character at position m.

LEN (<expression>)

LEN returns the number of characters in an expression.

REPLACE (<text>, <search_string>, <replacement_string>)

REPLACE searches for a specific text in a string and, if it finds it, replaces it.

CONCAT (<expression1>, <expression2>)

CONCAT function is the equivalent of the concatenation operator (+).

Group (aggregation) functions process sets of records (rows) of the table and return a certain result. Most of these functions usually act on all rows of the table. They support the following parameters:

- ALL - makes the function consider all records, including duplicates. This option is default.
- DISTINCT - makes the function consider only records that do not have a duplicate value (in the specified fields).

The aggregation functions include:

SUM ([DISTINCT | ALL] <expression>)

The SUM function calculates the sum of a set of numeric values.

AVG ([DISTINCT | ALL] <expression>)

The AVG function returns the arithmetic mean of the expression <expression>.

{MAX | MIN} ([DISTINCT | ALL] <expression>)









The MAX / MIN function returns the maximum or minimum value of the expression.

COUNT (* | [DISTINCT | ALL] <expression>)

The COUNT function returns the number of null lines returned by the query. Here, <expression> represents a field or expression.

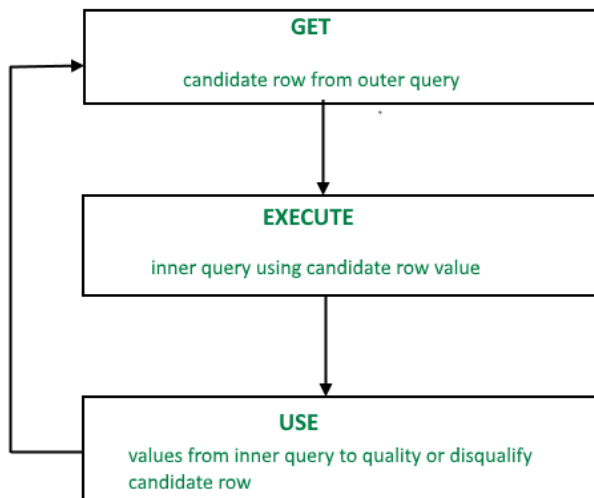
6. Queries with JOIN. Types of junctions.

JOIN is an SQL clause used to query and access data from multiple tables, based on logical relationships between those tables. In other words, JOINS indicate how SQL Server should use data from one table to select the rows from another table.

LEFT JOIN  Everything on the left + anything on the right that matches	<pre>SELECT * FROM TABLE_1 LEFT JOIN TABLE_2 ON TABLE_1.KEY = TABLE_2.KEY</pre>	ANTI LEFT JOIN  Everything on the left that is NOT on the right	<pre>SELECT * FROM TABLE_1 LEFT JOIN TABLE_2 ON TABLE_1.KEY = TABLE_2.KEY WHERE TABLE_2.KEY IS NULL</pre>
RIGHT JOIN  Everything on the right + anything on the left that matches	<pre>SELECT * FROM TABLE_1 RIGHT JOIN TABLE_2 ON TABLE_1.KEY = TABLE_2.KEY</pre>	ANTI RIGHT JOIN  Everything on the right that is NOT on the left	<pre>SELECT * FROM TABLE_1 RIGHT JOIN TABLE_2 ON TABLE_1.KEY = TABLE_2.KEY WHERE TABLE_1.KEY IS NULL</pre>
OUTER JOIN  Everything on the right + Everything on the left	<pre>SELECT * FROM TABLE_1 OUTER JOIN TABLE_2 ON TABLE_1.KEY = TABLE_2.KEY</pre>	ANTI OUTER JOIN  Everything on the left and right that is unique to each side	<pre>SELECT * FROM TABLE_1 OUTER JOIN TABLE_2 ON TABLE_1.KEY = TABLE_2.KEY WHERE TABLE_1.KEY IS NULL OR TABLE_2.KEY IS NULL</pre>
INNER JOIN  Only the things that match on the left AND the right	<pre>SELECT * FROM TABLE_1 INNER JOIN TABLE_2 ON TABLE_1.KEY = TABLE_2.KEY</pre>	CROSS JOIN  All combination of rows from the right and the left (cartesian product)	<pre>SELECT * FROM TABLE_1 CROSS JOIN TABLE_2</pre>

7. What are correlated subqueries used for?

Correlated subqueries are used for row-by-row processing. Each subquery is executed once for each row of the outer query.



A correlated subquery is one way of reading each row in a table and comparing values in each row against related data. It is used whenever a subquery must return a different result or set of results for each candidate row considered by the main query. In other words, you can use a correlated subquery to answer a multipart question whose answer depends on the value in each row processed by the parent statement.

8. How can INTERSECT and EXCEPT operations be rendered between two subqueries by nested queries?

The UNION, EXCEPT and INTERSECT operators enable to combine more than one SELECT statement to form a single result set.

EXCEPT returns distinct rows from the left input query that aren't output by the right input query. INTERSECT returns distinct rows that are output by both the left and right input queries operator. To combine the result sets of two queries that use EXCEPT or INTERSECT, the basic rules are:

The number and the order of the columns must be the same in all queries.

The data types must be compatible.

Syntax:

```
{ <query_specification> | ( <query_expression> ) }  
{ EXCEPT | INTERSECT }  
{ <query_specification> | ( <query_expression> ) }
```

9. What are the techniques for limiting the number of returned query lines?

The SQL LIMIT clause restricts how many rows are returned from a query.

The practical part

16. Furnizati numele si prenumele studentilor, care au studiat discipline cu un volum de lectii mai mic de 60 de ore, precum si profesorii (identificatorii) respectivi, care le-au predat.

```
select distinct s.Nume_Student, s.Prenume_Student, r.Id_Profesor
from studenti s
join studenti_reusita r
on s.Id_Student = r.Id_Student
join discipline d
on r.Id_Disciplina=d.Id_Disciplina
where d.Nr_ore_plan_disciplina < 60;
```

23. Sa se obtina lista disciplinelor (Disciplina) sustinute de studenti cu nota medie de promovare la examen mai mare de 7, in ordine descrescatoare dupa denumirea disciplinei.

```
select distinct d.Disciplina
from discipline as d
join studenti_reusita as r
on d.Id_Disciplina = r.Id_Disciplina
where r.Tip_Evaluare = 'Examen'
group by Disciplina
having avg(Nota) > 5
order by Disciplina desc;
```

39. Gasiti denumirile disciplinelor la care nu au sustinut examenul, in medie, peste 5% de studenti.

```
select d.Disciplina
from discipline d
join studenti_reusita r
on d.Id_Disciplina = r.Id_Disciplina
where r.Tip_Evaluare = 'Examen' and r.Id_Student = all
(select avg(Id_Student) from studenti_reusita
where Tip_Evaluare = 'Examen' and Nota <5)
group by Disciplina
having avg(Id_Student) > 3.75
order by Disciplina;
```