Transact-SQL

სტრუქტურირებული მოთხოვნების ენა (Structured Query Language, SQL) 1970 წელს შეიმუშავა IBM კორპორაციამ, როგორც მონაცემთა რელაციური ბაზების მართვის ენა. შემუშავებული იქნა ამ ენის რამდენიმე ვერსია. ბოლოს, 1992 წელს სტანდარტების ამერიკის ეროვნულმა ინსტიტუტმა (American National Standard Institute, ANSI) შეიმუშავა სტანდარტი - SQL-92, რომელიც აღწერს SQL Server-ის ქცევას და ახდენს მისი მუშაობის ძირითადი წესების განსაზღვრას. SQL Server-ზე რეალიზებულია Transact-SQL-ის (T-SQL) ის ვარიანტი, რომელიც უზრუნველყოფს ANSI SQL-92 სტანდარტის შესაძლებლობების დიდ ნაწილს.

SQL ენას აქვს ინსტრუქციების რამდენიმე კატეგორია, რომლებიც მოიცავს: მონაცემების აღწერის ენას (Data Definition Language, DDL), მონაცემებით მანიპულირების ენასა (Data Manipulation Language, DML) და მონაცემების მართვის ენას (Data Control Language, DCL):

- DDL ენას საქმე აქვს განსაზღვრებებთან და შეიცავს ისეთ ბრძანებებს, როგორიცაა: CREATE, ALTER და DROP;
- DML ენა იძლევა მონაცემების მოთხოვნისა და ცვლილების შესაძლებლობას და შეიცავს ისეთ ბრძანებს, როგორიცაა: SELECT, INSERT, UPDATE, DELETE და MERGE;
- DCL ენა დაკავშირებულია მიმართვის უფლებებთან ან უფლებამოსილებებთან და შეიცავს ისეთ ბრძანებებს, როგორიცაა: GRANT და REVOKE.

N G D

Transact-SQL ენის სინტაქსის წაკითხვის წესები

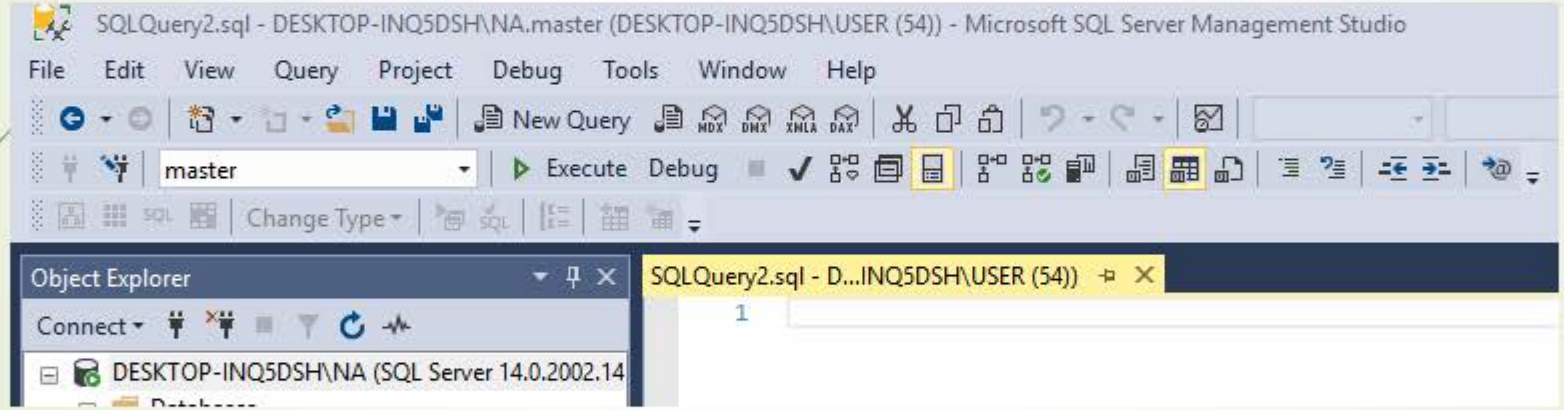
T-SQL ენის თითოეულ ბრძანებას აქვს საკუთარი **სინტაქსი** ანუ სწორად დაწერის წესი. განვიხილოთ CREATE VIEW ბრძანების სინტაქსი, რომელიც წარმოდგენის შესაქმნელად გამოიყენება:

```
CREATE VIEW [ სქემის_სახელი. ] წარმოდგენის_სახელი
[ ( სვეტის_სახელი [ , . . . n ] ) ]
[ WITH <წარმოდგენის_ატრიბუტები> [ , . . . n ] ]
AS
SELECT-ბრძანება [ WITH CHECK OPTION ]
<წარმოდგენის_ატრიბუტები> ::= { ENCRYPTION | SCHEMABINDING | VIEW_METADATA }
```

- ყოველგვარი ფრჩხილების გარეშე პირდაპირი ტექსტი - აუცილებელი არგუმენტი;
- კვადრატულ ფრჩხილებში (“[” და “]”) - არა აუცილებელი არგუმენტი;
- კუთხურ ფრჩხილებში (“<” და “>”) - კონსტრუქცია, რომელსაც თავისი სინტაქსი აქვს;
- ფიგურულ ფრჩხილებში (“{” და “}”) მოთავსებული არგუმენტებიდან ერთ-ერთი უნდა ავირჩიოთ, ხოლო “|” ნიშნავს "ან" ოპერატორს;
- [, . . . n] - არგუმენტის გამეორება, რომლებიც მძიმით იქნებიან გამოყოფილნი.

SQL Server Management Studio (SSMS)-ში ბრძანებების განხორციელება

MS SQL Server-ში SQL Server Management Studio-ს (SSMS) გამოყენებით მონაცემთა ბაზებში ნებისმიერი კოდინგის შეტანა წარმოებს ღილაკი New Query -ის გამოყენებით. ღილაკის მაუსით გააქტიურებისას მონიტორის ეკრანზე გამოისახება შესაბამისი ფანჯარა, რომელშიც მომხმარებელს ეძლევა შესაძლებლობა შეიტანოს ბრძანება/ბრძანებები:



ბრძანება/ბრძანებების შეტანის დასრულების შემდეგ მომხმარებელმა უნდა გააქტიუროს მაუსით ღილაკი Execute , რათა განახორციელოს ბრძანებ(ებ)ის შესრულება, რომლის შედეგსაც SQL Server-ი გამოსახავს იქვე ბრძანებ(ებ)ის შეტანის ფანჯრის ქვედა მიდამოში.

Transact-SQL-ის ელემენტები

Transact-SQL-ის ენას საფუძვლად უდევს შემდეგი ელემენტები:

- **იდენტიფიკატორები (identifiers)** - გამოიყენება კონკრეტულ ობიექტთან მიმართვისათვის (ცვლადი, ცხრილი, წარმოდგენა, სვეტი, ინდექსი, ტრიგერი და სხვ.). ამრიგად, იდენტიფიკატორი არის ობიექტის სახელი.
- **კომენტარები (comments)** - ტექსტი, რომელიც მოთავსებულია ბრძანების კოდში და შეიცავს მის განმარტებას. Transact-SQL-ის მიერ კომენტარის დამუშავება არ ხორციელდება. არსებობს ორი ტიპის კომენტარი - სტრიქონული და ბლოკური:
  - სტრიქონული კომენტარი „**--**“ სიმბოლოებით იწყება და სტრიქონის ბოლომდე ივრცობა;
  - ბლოკური კომენტარები „/*” სიმბოლოებით იწყება და „*/” სიმბოლოებით მთავრდება. შესაძლებელია რამდენიმე სტრიქონზე განივრცოს.
- **დარეზერვებული საკვანძო სიტყვები (reserved keywords)** - გამოიყენება SQL Server-ის მუშაობის მართვისთვის, რომლებიც არ არიან დამოკიდებული რეგისტრზე და არ შეიძლება წარმოადგენდნენ იდენტიფიკატორებს.

SQLQuery2.sql - D...INQ5DSH\USER (56))* | SQLQuery1.sql - D...INQ5DSH\USER (54))

```
1  -- ამოვიღოთ მონაცემები თანამდებობებზე
2  SELECT tanamd_id, tanamd_dasaxeleba -- აქ მითითებულია სვეტების სახელები
3    FROM Market.dbo.tanamd
4  /*
5  ამ ბრძანებაში FROM-ში გამოყენებულია ბაზის სახელის
6  მითითებაც და, ამიტომ, SELECT ბრძანების წინ
7  აღარ მიუთითეთ ბაზის სახელი */
```

NGD

Transact-SQL-ის გამოსახულებები

გამოსახულება (expression) არის იდენტიფიკატორების, ფუნქციების, არითმეტიკისა და ლოგიკის ოპერაციების ნიშნების, მუდმივებისა და სხვა ობიექტების კომბინაციას. გამოსახულება შედგება ოპერანდებისა (მონაცემებისა) და ოპერატორებისაგან (ოპერაციის ნიშნებისაგან). არსებობს შემდეგი ტიპის ოპერანდები:

- **მუდმივები** - მუდმივი სიდიდეებია, რომელთა მნიშვნელობები არ იცვლება. მუდმივა შეიძლება იყოს მთელი რიცხვი, წილადი, სტრიქონი და სხვ.;
- **ცვლადები** - გარკვეული ზომის მქონე მეხსიერების სახელდებული უბანი, რომელშიც მონაცემები იწახება;
- **სვეტების სახელები** - შეიძლება გამოვიყენოთ როგორც ოპერანდი გამოსახულებაში. ამ დროს სვეტის სახელის ნაცვლად SQL Server-ი ავტომატურად ჩასვამს შესაბამის მნიშვნელობას. ცხრილის ერთი სტრიქონიდან მეორეზე გადასვლის დროს სვეტის სახელს შესაბამისი მნიშვნელობა ენიჭება;
- **ფუნქციები** - სახელდებული, მცირე ზომის პროგრამები, რომლებიც ახდენენ მონაცემების დამუშავებას და შედეგის დაბრუნებას. ფუნქციას შეიძლება გააჩნდეს ან არ გააჩნდეს პარამეტრები;
- **ქვემოთხოვნები** - გამოსახულება შეიძლება შეიცავდეს ქვემოთხოვნას, რომელიც ახდენს მონაცემების შესაბამისი ნაკრების მომზადებას.

Transact-SQL-ის იდენტიფიკატორები

არსებობს იდენტიფიკატორების ორი ტიპი:

- **სტანდარტული (regular identifiers)**;
- **შეზღუდული (delimited identifiers)**.

სტანდარტული იდენტიფიკატორების განსაზღვრისას დაცულია იდენტიფიკატორების შექმნის წესები, ხოლო შეზღუდული იდენტიფიკატორები (რომლის განსაზღვრისას არ არის დაცული იდენტიფიკატორების შექმნის წესები) უნდა მოვათავსოთ კვადრატულ ფრჩხილებში **[ ]** ან ბრჭყალებში “ ”.

სტანდარტული იდენტიფიკატორის განსაზღვრის წესები:

1. პირველი სიმბოლო უნდა აკმაყოფილებდეს შემდეგ წესებს:
   - უნდა შეესაბამებოდეს Unicode Standard 2.0 სტანდარტს, რომლის მიხედვით ეს სიმბოლო უნდა იყოს ნებისმიერი ლათინური სიმბოლო a-დან z-მდე ან ეროვნული ანბანის სიმბოლო;
   - შეიძლება იყოს ხაზგასმის სიმბოლო, @ ან # სიმბოლო (როგორც უკვე იქნა განხილული იდენტიფიკატორი, რომელიც # სიმბოლოთი იწყება, აღნიშნავს ლოკალურ დროებით ობიექტს, ხოლო იდენტიფიკატორი, რომელიც ## სიმბოლოებით იწყება, აღნიშნავს გლობალურ დროებით ობიექტს. ასევე, იდენტიფიკატორი, რომელიც @ სიმბოლოთი იწყება, აღნიშნავს ცვლადს).

(გაგრძელება შემდეგ სლაიდზე)

Transact-SQL-ის იდენტიფიკატორები

2. იდენტიფიკატორის მომდევნო სიმბოლოები შეიძლება იყოს:
   - Unicode Standard 2.0 სტანდარტით განსაზღვრული სიმბოლო;
   - ციფრი (0-9);
   - @, $, _, # სიმბოლო.
3. იდენტიფიკატორის შიგნით დაუშვებელია სპეციალური სიმბოლოების გამოყენება: ~, !, %, ^, &, - , (,), {, }, “, ., \, ' და ინტერვალი;
4. იდენტიფიკატორი არ უნდა იყოს დარეზერვებული სიტყვა და უნდა იყოს უნიკალური;
5. იდენტიფიკატორი არ არის დამოკიდებული რეგისტრზე.

წესების გვერდის ასავლელად იდენტიფიკატორის სახელი ბრჭყალებში ან კვადრატულ ფრჩხილებში თავსდება ანუ გარდაიქმნებიან შეზღუდულ იდენტიფიკატორებად.

მაგალითი:

სწორი იდენტიფიკატორებია: **t_id, r4kod#, [own file], [FROM]**

ყალბი იდენტიფიკატორებია: **t id, r4^kod%, own file, FROM**

(გაგრძელება შემდეგ სლაიდზე)

NGD

Transact-SQL-ის იდენტიფიკატორები

ნებისმიერი ობიექტი გარკვეული მომხმარებლის მიერ იქმნება და კონკრეტულ მონაცემთა ბაზას ეკუთვნის. თავის მხრივ, მონაცემთა ბაზა მოთავსებულია კონკრეტულ SQL Server-ზე. SQL Server-ის, მონაცემთა ბაზის, სქემისა და ობიექტის სახელის საფუძველზე იქმნება ობიექტის **სრული სახელი (complete name)** ან სრულად განსაზღვრული სახელი **(full qualified name)**, რომელსაც აქვს შემდეგი სინტაქსი:

[[[სერვერის_სახელი.][მონაცემთა_ბაზის_სახელი].][სქემის_სახელი].]ობიექტის_სახელი

SQL Server-ის ნებისმიერ ობიექტს უნდა ჰქონდეს უნიკალური, სრულად განსაზღვრული სახელი.

ობიექტის შექმნისას შეგვიძლია არ მივუთითოთ SQL Server-ის, მონაცემთა ბაზისა და სქემის სახელი. თუ ვასრულებთ მოთხოვნას, რომელიც ბევრ ობიექტთან მუშაობს, მაშინ უმჯობესია ობიექტების სრული სახელების გამოყენება პრობლემების თავიდან აცილების მიზნით. თუ ობიექტთან მიმართვისას არ არის მითითებული SQL Server-ის სახელი, მაშინ აიღება მიმდინარე SQL Server-ის სახელი. თუ მითითებული არ არის მონაცემთა ბაზის სახელი, მაშინ აიღება მიმდინარე მონაცემთა ბაზა. თუ მითითებული არ არის სქემის სახელი, მაშინ აიღება **dbo** სქემა. ის ავტომატურად იქმნება თითოეულ მონაცემთა ბაზაში, და გამოიყენება, როგორც ნაგულისხმევი სქემა იმ შემთხვევაში, როცა მომხმარებელი აშკარად არ არის დაკავშირებული სხვა სქემასთან. თუ მითითებული ობიექტი არ მოიძებნა, მაშინ გაიცემა შეტყობინება შეცდომის შესახებ.

Transact-SQL-ის იდენტიფიკატორები

მაგალითები:

1. **Server1.Baza1.dbo.Cxrili1** – ობიექტის სახელის სრული მითითება;
2. **Baza1.dbo.Cxrili1** – ობიექტის სახელის მითითება SQL Server-ის მითითების გარეშე (ძირითადად გამოიყენება ერთი SQL Server-ის პირობებში);
3. **Cxrili1** – ობიექტის სახელის მითითება მხოლოდ ცხრილის სახელის გამოყენებით (ამ შემთხვევაში ბაზის სახელი წინასწარ უნდა იქნეს განსაზღვრული USE-ს გამოყენებით).

იმისათვის, რომ მივმართოთ ცხრილის ან წარმოდგენის რომელიმე სვეტს, აუცილებელია სრულ სახელში სვეტის სახელის დამატება:

[[[სერვერის_სახელი.] [მონაცემთა_ბაზის_სახელი].] [სქემის_სახელი].]
ობიექტის_სახელი.სვეტის_სახელი

მაგალითი:

Server1.Baza1.dbo.Cxrili1.Sveti1 – ობიექტის სახელის სრული მითითება და მიმართავს Server1 სერვერის Baza1 მონაცემთა ბაზის dbo კლასის Cxrili1 მონაცემთა ცხრილის Sveti1 სვეტს.

(გაგრძელება შემდეგ სლაიდზე)

Transact-SQL-ის იდენტიფიკატორები

შეზღუდული იდენტიფიკატორების გამოყენების წესები დამოკიდებულია QUOTED_IDENTIFIER პარამეტრის მნიშვნელობაზე, რომლის მნიშვნელობა შესაძლებელია განისაზღვროს SET ბრძანებით. მისი სინტაქსია:

SET QUOTED_IDENTIFIER { ON | OFF }

თუ არის QUOTED_IDENTIFIER ON, მაშინ შეზღუდვა ხორციელდება შემდეგი წესებით:

- იდენტიფიკატორების შეზღუდვისათვის გამოიყენება ბრჭყალები “ ” და კვადრატული ფრჩხილები **[ ]**;
- სიმბოლური სტრიქონები უნდა მოთავსდეს აპოსტროფებში ‘ ’ (არა ბრჭყალებში “ “, რომელიც იდენტიფიკატორის შეზღუდვისათვის გამოყენება).

მაგ.: **SELECT * FROM [MY TABLE] WHERE “MY COLUMN" = N‘Nukri‘.**

თუ არის QUOTED_IDENTIFIER OFF, მაშინ შეზღუდვა ხორციელდება შემდეგი წესებით:

- იდენტიფიკატორების შეზღუდვისათვის გამოიყენება კვადრატული ფრჩხილები **[ ]**;
- სიმბოლური სტრიქონები უნდა მოთავსდეს აპოსტროფებში ‘ ’ ან ბრჭყალებში “ “ (თუ სტრიქონი აპოსტროფს შეიცავს, მაშინ იგი შეგვიძლია ბრჭყალებში მოვათავსოთ).

მაგ.: **SELECT * FROM [MY TABLE] WHERE [MY COLUMN] = N”O’Bama”.**

NGD