howest
hogeschool

Syllabus

Databases

Mileto Di Marco
Kristien Roels
Bart Soete

Inhoudsopgave

H10 - User-defined functions

1 Inleiding

User-defined functions (UDFs) of gebruikersfuncties zijn functies die parameters accepteren, een actie uitvoeren, zoals een complexe berekening, en het resultaat van die actie afleveren als een scalaire waarde of als een resultaatset. De user-defined functions worden opgeslagen in de databank op de server.

Merk op dat user-defined functions noch de data noch de databankstructuur kunnen aanpassen. INSERT, UPDATE, DELETE, CREATE, ALTER en DROP zijn dus niet mogelijk binnen een functie.

User-defined functions bieden een aantal opportuniteiten:

- Herbruiken van code. De functie wordt éénmaal gemaakt en kan vele keren opgeroepen worden.
- Betere performantie. Net zoals bij stored procedures wordt de functie gecompileerd als ze de eerste keer opgeroepen wordt en wordt er een execution plan bewaard in de cache van de Database Engine. Dit execution plan wordt gebruikt bij volgende oproepen.

We behandelen 2 soorten user-defined functions:

- Scalar functions of scalar-valued functions of scalaire functies
- Inline table-valued functions

Interessant is dat een scalaire functie rechtstreeks in de SELECT- en de WHERE-clausule van een SELECT-statement kan gebruikt worden en dat een table-valued functie rechtstreeks in de FROM-clausule van een SELECT-statement kan gebruikt worden. Anders gezegd: door het gebruik van user-defined functions kan complexe logica binnen een query ingesloten worden.

De UDFs zijn terug te vinden onder de databanknode, Programmability, Functions. Rechtsklik op de gewenste gebruikersfunctie en kies Modify.

Relevante Help-pagina's:
https://docs.microsoft.com/nl-nl/sql/relational-databases/user-defined-functions/user-defined-functions
https://docs.microsoft.com/nl-nl/sql/t-sql/statements/create-function-transact-sql

De voorbeelddatabank die in dit hoofdstuk gebruikt wordt is de GRADSchool-databank.

2 Scalar functions

Het aanmaken en wijzigen van een scalaire functie:

```
CREATE [OR ALTER] FUNCTION functienaam(@parameter type [=default], …)
RETURNS type
AS
BEGIN
   SQL-statement […]
   RETURN expressie
END
```

De statements ALTER FUNCTION en DROP FUNCTION [IF EXISTS] worden gebruikt om een scalaire functie respectievelijk te wijzigen en te verwijderen.

Een scalaire functie levert 1 waarde af met het RETURN-statement.
Scalaire functies kunnen gebruikt worden binnen expressies die 1 waarde accepteren.

Scalaire functies moeten deterministisch zijn. Dit betekent dat ze steeds dezelfde waarde moeten afleveren bij dezelfde inputparameters. Daardoor kunnen niet-deterministische functies zoals newid() en rand() niet gebruikt worden binnen scalaire functies.

Een scalaire functie wordt opgeroepen met de tweedelige naam schemanaam.functienaam. Het is ook mogelijk om de drie- of vierdelige naam op te geven: [[servernaam].databanknaam].schemanaam.functienaam.

Schrijf een scalaire functie fLeeftijd die de leeftijd berekent uit een geboortedatum.

Gebruik deze functie om een lijst te leveren van alle studenten met hun leeftijd.

```
create or alter function fLeeftijd(@geboortedatum date)
returns int
as
begin
declare @leeftijd int
declare @vandaag date = getdate()

set @leeftijd = year(@vandaag) - year(@geboortedatum)

if ( (month(@geboortedatum) > month(@vandaag))
   or (month(@geboortedatum) = month(@vandaag) and day(@geboortedatum) > day(@vandaag)) )
begin
set @leeftijd = @leeftijd - 1
end

return @leeftijd
end
```

Opmerking: de voorwaarde in de IF kan ook anders geschreven worden:
if (DATEPART(dayofyear, @vandaag) < DATEPART(dayofyear, @geboortedatum))

```
select *, dbo.fLeeftijd(geboortedatum) as leeftijd
from studenten
```

3 Inline table-valued functions

Het aanmaken en wijzigen van een inline table-valued function:

```
CREATE [OR ALTER] FUNCTION functienaam(@parameter type [=default], …)
RETURNS TABLE
AS
RETURN (
   SELECT-statement
)
```

De statements ALTER FUNCTION en DROP FUNCTION [IF EXISTS] worden gebruikt om een inline table-valued function respectievelijk te wijzigen en te verwijderen.

Een inline table-valued function levert een tabel af, het resultaat van één SELECT-statement. Inline table-valued functions kunnen gebruikt worden binnen de FROM-clausule van een SELECT-statement en zijn vergelijkbaar met views.

Een inline table-valued function wordt opgeroepen met de tweedelige naam schemanaam.functienaam.

Schrijf een functie fMeestBetaaldPerGeslacht die een tabel aflevert met per geslacht de studenten die het meest betaald hebben.

Gebruik deze functie in de FROM-clausule van een SELECT-statement.

```
create or alter function fMeestBetaaldPerGeslacht()
returns table
as
return (
select *
from studenten s1
where betaald = (select max(betaald)
                 from studenten
                 where geslacht =  s1.geslacht)
)
```

```
select *
from dbo.fMeestBetaaldPerGeslacht()
```

4 User-defined functions versus stored procedures

Als er nood is aan een berekende waarde, definieer dan een scalaire functie.

Als er wijzigingen aan de data moeten aangebracht worden of wijzigingen aan de structuur van de databank, schrijf dan een stored procedure.

Functies kunnen rechtstreeks in een query gebruikt worden.
Stored procedures worden uitgevoerd met EXEC en kunnen niet rechtstreeks in een query gebruikt worden.

Als er nood is aan een resultaatset die verder moet gebruikt worden in een query, schrijf dan een table-valued function:

- Een table-valued function kan namelijk in de FROM-clausule van een SELECT-statement opgenomen worden.
- Een stored procedure kan niet rechtstreeks in de FROM-clausule van een SELECT-statement opgenomen worden. Bij een stored procedure moet de resultaatset eerst in een tijdelijke tabel geplaatst worden, van waaruit dan verder kan gewerkt worden.