
Gespeicherte Funktionen

Funktionen

- 3 Arten von Funktionen
 - Skalarfunktionen (liefern einzelnen Wert zurück)
 - Tabellenwertfunktion (liefern Tabelle zurück)
 - Inline-Funktion (Spezialfall einer Tabellenwert-Funktion)
- Funktionen liefern immer einen Wert zurück
- Funktionen können in SQL-Anweisungen verwendet werden
- In Funktionen können viele Anweisungen nicht genutzt werden
 - z.B. Definitionsvorgänge (CREATE, ALTER...), Datenänderungen

Skalarfunktion

- Geben einen einzelnen Wert zurück
 - Skalar = mathematische Größe, die allein durch die Angabe eines Einheitswertes charakterisiert ist (einzelner Zahlenwert, einzelne Zeichenkette usw.)

Syntax:

```
CREATE FUNCTION schema.name ([@Parameter <Typ> [,...]])  
RETURNS <Typ>  
[WITH ENCRYPTION | SCHEMABINDING | EXECUTE AS]  
AS
```

Funktionsdefinition mit optionaler Parameterliste, notwendigem Rückgabetyt und optionalen Eigenschaften

```
BEGIN  
    <Anweisungen>  
RETURN <Rückgabewert>  
END
```

Funktionsrumpf durch BEGIN und END begrenzt, mit den Anweisungen und dem notwendigen RETURN-Wert

Skalarfunktion Eigenschaften

- **ENCRYPTION**
 - Text der Funktion wird in ein verborgenes Format konvertiert
- **SCHEMABINDING**
 - Von der Funktion verwendete Tabellen können nicht mehr „gegen die Funktionsdefinition“ geändert werden
- **EXECUTE AS <Benutzer>**
 - Funktion wird innerhalb des Sicherheitskontextes des angegebenen Benutzers ausgeführt

Beispiel

```
CREATE FUNCTION fnProz_arb(@ang int)
RETURNS int
AS
```

```
BEGIN
```

```
    DECLARE @anteil int
    SELECT @anteil = SUM(proz_arb)
    FROM pro_ang
    WHERE a_nr = @ang
```

```
    RETURN @anteil
END
```

Funktionskopf mit Übergabe-
parameter und Rückgabetyt

Funktionsrumpf in einem BEGIN
... END-Block mit RETURN-Wert

Aufruf mit:

```
SELECT dbo.fnProz_arb(401);
```

Aufruf mit Angabe des
Schemanamens

Inline-Funktion

- Eigenschaften
 - Gibt eine Tabelle zurück
 - Enthält nur eine SELECT-Anweisung
 - Es gibt keinen Funktionsrumpf
 - RETURNS TABLE in RETURNS-Klausel
 - RETURN-Klausel enthält SELECT-Anweisung

Beispiel

```
CREATE FUNCTION fnAngAbt(@abteilung int)
RETURNS TABLE
AS
RETURN SELECT ang.name,
             abt.abt_name
FROM ang
INNER JOIN abt
ON ang.abt_nr = abt.abt_nr
WHERE abt.abt_nr = @abteilung;
```

RETURNS TABLE ist Pflicht bei Inline-Funktion

Es gibt keinen durch BEGIN...END gekennzeichneten Funktionsrumpf; Es gibt nur ein RETURN SELECT (...)

Aufruf mit:

```
SELECT * FROM dbo.fnAngAbt(2);
```

Tabellenwertfunktion

Beispiel

```
CREATE FUNCTION fnAngData(@anr int)
RETURNS @tabelle TABLE (Name nvarchar(max),
                          Vorgesetzter nvarchar(max),
                          Untergebener nvarchar(max))
```

```
AS
BEGIN
```

```
    INSERT INTO @tabelle
```

```
    SELECT ang.name, a.name, b.name
    FROM ang
    LEFT JOIN ang as a ON ang.vorg = a.a_nr
    LEFT JOIN ang AS b on ang.a_nr = b.vorg
    WHERE ang.a_nr = @anr
```

```
    RETURN
END
```

```
SELECT * FROM dbo.fnAngData(205);
```

Definition der
Rückgabetabelle

Befüllen der Rückgabetabelle

SELECT-Statement

RETURN Rückgabe

Funktionsrumpf

- **Deterministische Funktionen**

- Geben bei gleichen Eingabewerten und gleichem Datenbankstatus immer auch den gleichen Wert zurück
- Integrierte Aggregat- und Zeichenfolgen-Funktionen
- Ergebnisse können indiziert werden

- **Nicht Deterministische Funktionen**

- Können bei gleichen Eingabewerten und gleichem Datenbankstatus unterschiedliche Werte zurückgeben
- Ergebnisse können nicht indiziert werden
- Integrierte Konfigurations-, Cursor-, Metadaten-, Sicherheits- und statistische Systemfunktionen