Trennen eines Strings in SQL Server mit PARSENAME

Kathegorie
SQL Server

Simon
Schlaipfer

Problem

Datentabellen und -ansichten wie der Name einer Person oder ihre Adresse werden in SQL Server normalerweise entweder in einer verketten Zeichenfolge, also einem String, oder als einzelne Spalten für jeden Teil des gesamten Werts gespeichert. Zum Beispiel: 55131 – Schusterstraße 5, Mainz, RP. Diese Informationen können sowohl in einer einzelnen Spalte, als auch mit Hilfe der Funktion **PARSENAME** in beliebig vielen Spalten, dargestellt werden.

Aus diesem Grund muss ein Datenbankadministrator solche Werte zwangsläufig verketten oder analysieren, um sie an die Bedürfnisse der Benutzer anzupassen.

Um den Umgang mit **PARSENAME** zu verdeutlichen, wollen wir im folgenden, aufbauend auf dem obigen Beispiel, eine Adressspalte erzeugen, in der die vollständige Adresse (Name, Hausnummer, Stadt und Bundesstaat) durch Kommas in einer verketteten Spalte getrennt ist. Anschließend soll diese erzeugte Spalte weiter in die Bestandteile Straße, Stadt und Bundesland aufgeteilt werden. Hierbei soll für jedes Attribut eine eigene Spalte zur Verfügung stehen.

Lösung

Anknüpfend an das oben gegebene Beispiel erstellen wir eine Testdatenbank und eine Testtabelle für das Analysebeispiel:

```
USE master;
GO

CREATE DATABASE TestDB;
GO

USE TestDB;
GO
```


Mainzer Datenfabrik GmbH
Rotterdamer Platz 2, 55120 Mainz

info@madafa.de
madafa.de

+49 6131 3331612

Anschließend erstellen wir eine Tabelle in der neuen (Test-)Datenbank mit nur zwei Spalten – einer ID-Spalte und einer Adressspalte:

```
CREATE TABLE dbo.customerAddress(
     colID INT IDENTITY PRIMARY KEY
   , myAddress VARCHAR(200)
   );
GO
```

Nun füllen wir die Tabelle mit einigen Daten. Hier ist zu beachten, dass die Adresse ein Zeichenfolgewert in einer Spalte ist:

```
INSERT INTO dbo.customerAddress(myAddress)
VALUES('55131 - Schusterstraße 5, Mainz, RP')
    , ('55246 - Lilienweg 9, Wiesbaden, HE')
    , ('66299 - Ackerstraße 17, Friedrichsthal, SL')
    , ('01099 - Lagestraße 1, Dresden, SN')
GO
```

Mit der folgenden SELECT-Anweisung verifizieren wir das erfolgreiche Erstellen und Befüllen der Tabelle:

```
SELECT *
FROM dbo.customerAddress;
GO
```

Als Ergebnis erhalten wir zwei Spalten. Die ID-Spalte und die Adressspalte:

	colID	myAdress
1	1	55131 - Schusterstraße 5, Mainz, RP
2	2	55246 - Lilienweg 9, Wiesbaden, HE
3	3	66299 - Ackerstraße 17, Friedrichsthal, SL
4	4	01099 - Lagestraße 1, Dresden, SN

Der nächste Schritt besteht nun darin, die drei einzelnen Teile der Adresse, die durch ein Komma getrennt sind, zu analysieren.

```
SELECT
     REVERSE(PARSENAME(REPLACE(REVERSE(myAddress), ',', '.'), 1)) AS [Street]
   , REVERSE(PARSENAME(REPLACE(REVERSE(myAddress), ',', '.'), 2)) AS [City]
   , REVERSE(PARSENAME(REPLACE(REVERSE(myAddress), ',', '.'), 3)) AS [State]
FROM dbo.custAddress;
GO
```

Da wir die ID-Spalte in der obigen SELECT-Anweisung nicht aufgerufen haben, enthält die Ausgabe nur die in drei Spalten unterteilte Adress-Zeichenfolge.

	Street	City	State
1	55131 - Schusterstraße 5	Mainz	RP
2	55246 - Lilienweg 9	Wiesbaden	HE
3	66299 - Ackerstraße 17	Friedrichsthal	SL
4	01099 - Lagestraße 1	Dresden	SN

Natürlich sind Datenbankadministratoren in der Praxis mit wesentlich komplexeren Tabellen oder Ansichten konfrontiert. Obwohl das obige Beispiel schon verdeutlicht, wie das Analysieren eines Zeichenfolgewerts funktioniert, wollen wir im folgenden Abschnitt eine komplexere Situation betrachten.

Ein etwas komplexeres Beispiel

Auch hier erstellen wir erneut eine Beispieltabelle in unserer Testdatenbank mit zusätzlichen Spalten.

```
CREATE TABLE colleagueData(
     colID INT IDENTITY PRIMARY KEY
   , empID INT
   , empName VARCHAR(50)
   , empAddress VARCHAR(200)
   , empPhone VARCHAR(12)
   , jobClass VARCHAR(50)
   );
GO
```


Mainzer Datenfabrik GmbH
Rotterdamer Platz 2, 55120 Mainz
info@madafa.de
madafa.de
+49 6131 3331612

Nun füllen wir die Tabelle mit Werten:

```
INSERT INTO colleagueData(empID, empName, empAddress, empPhone,
jobClass)
VALUES (1, 'Richard, M, Müller', '55131 - Schusterstraße 5, Mainz,
RP', '06131-635793', 'Programmierer')
     , (2, 'Rainer, S, Winkler', '55246 - Lilienweg 9, Wiesbaden,
HE', '06122-692058', 'Designer')
     , (3, 'André, K, Weißlinger', '66299 - Ackerstraße 17,
Friedrichsthal, SL', '06897-394052', 'Verkäufer')
     , (4, 'Anton, W, Meyer', '01099 - Lagestraße 1, Dresden, SN',
'0351-384752', 'Manager')
GO
```

Wir erstellen nun eine gemischte SELECT-Anweisung, um die Spalten empName, empAddress und empPhone in separate Spalten aufzuteilen.

Im folgenden Codeblock ist zu beachten, dass mit jedem Parsen in einer neuen Spalte die Zählung von vorne beginnen muss. Weiterhin sind die Zeichenfolgenwerte empName und empAddress durch ein Komma, der Zeichenfolgenwert empPhone allerdings durch einen Bindestich getrennt. Dies muss in der Funktion **PARSENAME** kenntlich gemacht werden.

```
SELECT
   colID
   , empID
   -- The following section breaks down the "empName" column into
three columns.
   , REVERSE(PARSENAME(REPLACE(REVERSE(empName), ',', '.'), 1)) AS
FirstName
   , REVERSE(PARSENAME(REPLACE(REVERSE(empName), ',', '.'), 2)) AS
MiddleName
   , REVERSE(PARSENAME(REPLACE(REVERSE(empName), ',', '.'), 3)) AS
LastName
   -- The following section breaks down the "empAddress" column into four
columns.
   , REVERSE(PARSENAME(REPLACE(REVERSE(empAddress), ',', '.'), 1))
AS Street
   , REVERSE(PARSENAME(REPLACE(REVERSE(empAddress), ',', '.'), 2))
AS City
   , REVERSE(PARSENAME(REPLACE(REVERSE(empAddress), ',', '.'), 3))
AS State
   -- The following section breaks down the "empPhone" column into
two columns
   , REVERSE(PARSENAME(REPLACE(REVERSE(empPhone), '-', '.'), 1)) AS
Prefix
   , REVERSE(PARSENAME(REPLACE(REVERSE(empPhone), '-', '.'), 2)) AS
[Primary]
   , jobClass
FROM colleagueData;
GO
```

Die Ausgabe sollte nun so aussehen:

	colID	empID	FirstName	MiddleName	LastName	Street	City	State	Prefix	Primary	jobClass
1	1	1	Richard	M	Müller	55131 - Schusterstraße 5	Mainz	RP	06131	635793	Programmierer
2	2	2	Rainer	S	Winkler	55246 - Lilienweg 9	Wiesbaden	HE	06122	692058	Designer
3	3	3	André	K	Weißlinger	66299 - Ackerstraße 17	Friedrichsthal	SL	06897	394052	Verkäufer
4	4	4	Anton	W	Meyer	01099 - Lagestraße 1	Dresden	SN	0351	384752	Manager

Fazit

Die **PARSENAME**-Funktion ist eine praktische Ergänzung zu Ihrem T-SQL-Toolkit für Abfragen mit begrenzten Daten. Es ermöglicht das Parsen und Zurückgeben einzelner Segmente eines Zeichenfolgenwerts in separaten Spalten.

parse, parsename, split string, split string, split string
sql server, sql server, string

Mainzer Datenfabrik GmbH
Rotterdamer Platz 2, 55120 Mainz
info@madafa.de
madafa.de
+49 6131 3331612