Tutorial:
Anbindung einer SQL-Datenquelle

Version: 1.1

Autor: David Haasler

Inhaltsverzeichnis

Voraussetzungen

Der Entwickler benötigt für dieses Tutorial den Zugriff auf die web.config der onion.net Render Engine, sowie einen Zugriff auf einen SQL-Server.

Beschreibung

Dieses Tutorial beschreibt die Anbindung einer Datenbank als SQL-Datenquelle an die onion.net Render Engine.

Zeichenerklärung

Text, der grün umrandet und mit dem Pfeil-Symbol gekennzeichnet ist, enthält konkrete Anweisungen, was als nächstes zu tun ist.

Texte in solchen Kästchen enthalten Tipps und Tricks.

```
Quellcode wird in solch blauen Boxen dargestellt.
```

1 Erstellen der Datenbank

Im ersten Schritt erstellen wir eine Datenbank mit Testdaten, welche wir später als Datenquelle in unserer onion.net Render Engine konfigurieren und Daten abrufen wollen.

Das folgende Skript erstellt eine Test-Datenbank, legt eine Tabelle mit Testdaten an und erzeugt zwei gespeicherte Prozeduren.

```
CREATE DATABASE SQLTest
GO
USE SQLTest
GO
CREATE TABLE [Contacts](
[id] [int] IDENTITY(1,1) NOT NULL,
[firstname] [nvarchar](50) NOT NULL,
[lastname] [nvarchar](50) NOT NULL,
[age] [int] NOT NULL,
[male] [bit] NOT NULL
) ON [PRIMARY]
GO
INSERT INTO [Contacts] (firstname, lastname, age, male) VALUES ('Walter',
'Webb', 20, 'True')
INSERT INTO [Contacts] (firstname, lastname, age, male) VALUES
('Kristina', 'Kreativ', 25, 'False')
INSERT INTO [Contacts] (firstname, lastname, age, male) VALUES ('Max',
'Mustermann', 30, 'True')
INSERT INTO [Contacts] (firstname, lastname, age, male) VALUES ('Erika',
'Mustermann', 35, 'False')
GO
CREATE PROCEDURE [getContacts] AS
BEGIN
SELECT id, firstname, lastname FROM Contacts
END
GO
CREATE PROCEDURE [getContact] (@id int) AS
BEGIN
DECLARE @count int
SELECT @count = COUNT(1) FROM Contacts WHERE id = @id
IF @count &gt; 0
SELECT id, firstname, lastname, age, male FROM Contacts WHERE id = @id
```

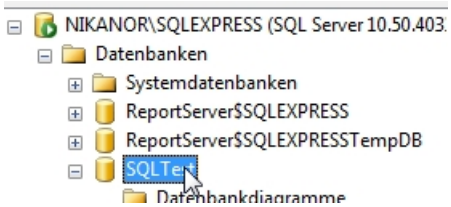
```
ELSE
RAISERROR ('no data found', 16, 1)
END
GO
```

Nach dem Ausführen des SQL-Scripts muss die Render Engine neu gestartet werden. Dies kann manuell erfolgen oder, indem man Änderungen in der web.config-Datei vornimmt (Setzen eines Leerzeichens reicht). Wird die Render Engine nicht neu gestartet, wird folgende Fehlermeldung angezeigt: **Die gespeicherte Prozedur 'table_validation' wurde nicht gefunden.**

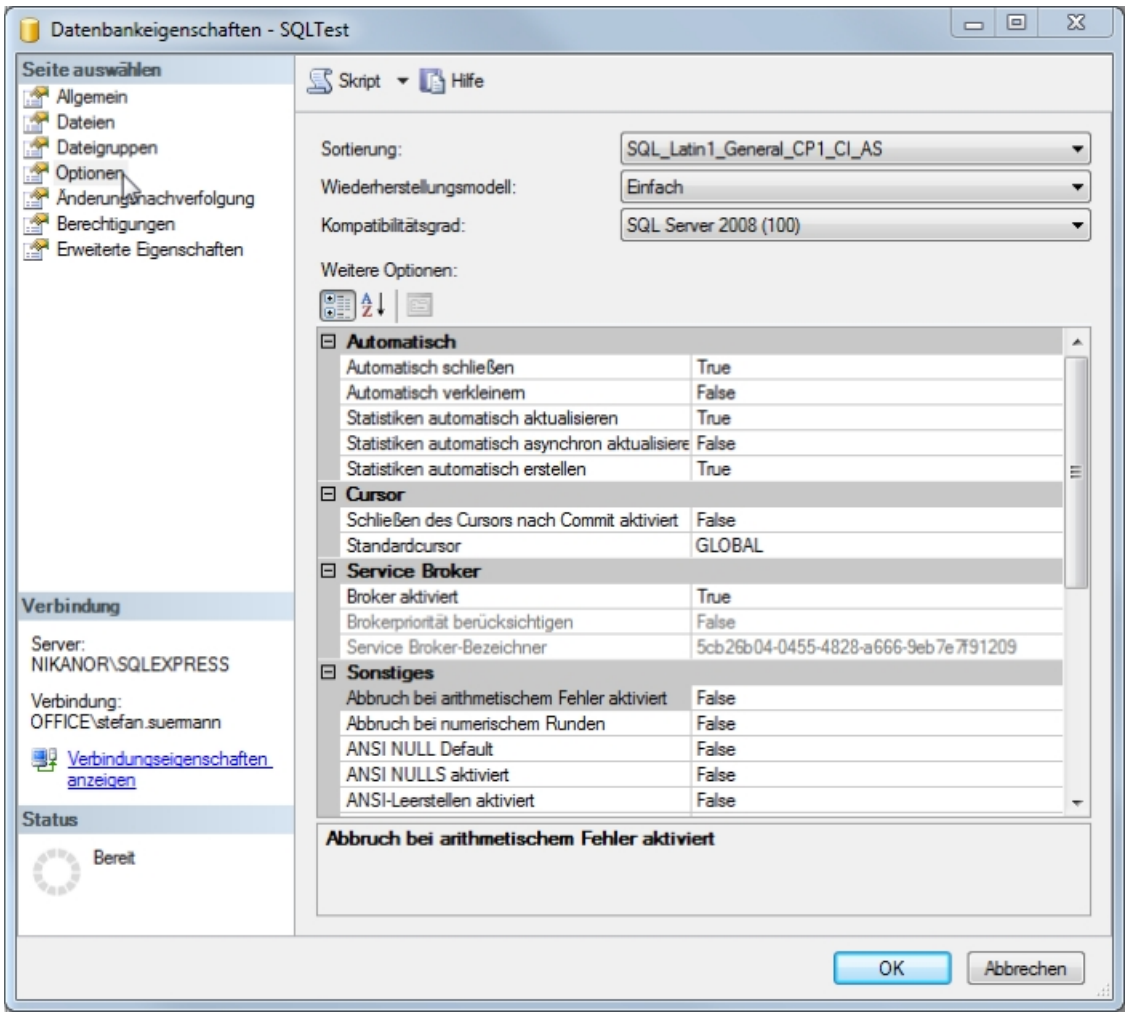
1.1 Fehlerbehebung bei Falsche Syntax in der Nähe von 'HOLDLOCK'

Sollte eine Fehlerseite mit folgender Fehlermeldung kommen ***Falsche Syntax in der Nähe von 'HOLDLOCK'...*** muss wie folgt vorgegangen werden:

Öffnen Sie dazu ein Programm zur Interaktion des DBMS (Database Management System) (z.B. Microsoft SQL Server Management Studio) und melden Sie sich am Datenbankserver an. Navigieren Sie zur neu erstellten Datenbank *SQLTest*.



Rechtsklicken Sie auf die Dantnbank und wählen Sie *Eigeschaften*. Wählen Sie im Strukturbum links den Punkt *Optionen* . Ändern Sie nun den Kompatibitätsgrad auf *SQL Server 2000 (80)*.



Klicken Sie nun auf den Button *OK*, damit sich der Dialog schließt. Laden Sie nun die Seite erneut.

2 Konfigurieren der onion.net Render Engine

Nachdem die Testdaten im SQL-Server erstellt worden sind, können wir nun diese Datenbank als Datenquelle in der web.config unserer onion.net Render Engine konfigurieren. Der folgende Konfigurationsabschnitt wird innerhalb des Elements dataSources eingefügt.

```
<source xlinkPrefix="sql"
type="Onion.RenderEngine.CommonDataSources.MsSqlServerRepository,
Onion.RenderEngine.CommonDataSources">
  <server connectionString="user
id=username;password=password;server=localhost;initial catalog=SQLTest"
typeIdentifier="SQL">
    <commands>
      <command name="getContacts" contentType="ContactList">
        <dataViews>
          <list
                    query="exec getContacts"
                    fields="id firstname lastname"
                    resultType="sql"
                    dependencies="Contacts"
          />
        </dataViews>
      </command>
      <command name="getContact" contentType="Contact">
        <dataViews>
          <detail
                    query="exec getContact @id"
                    fields="id firstname lastname age male"
                    resultType="sql"
                    dependencies="Contacts"
          />
        </dataViews>
        <queryParameters>
          <parameter name="id" type="int" />
        </queryParameters>
      </command>
    </commands>
  </server>
</source>
```

Wie Sie im Konfigurationsabschnitt evtl. erkennen, wird eine weitere Datenquelle definiert. Als Prefix wurde nun sql konfiguriert.

Innerhalb des sources-Elements wurde die Verbindung zum SQL-Server konfiguriert. In diesem Fall läuft der SQL-Server auf dem lokalen System.

Für diesen Server werden nun zwei Kommandos erzeugt. Das erste Kommando "getContacts" gibt die Ergebnisse als Datentyp ContactList zurück. Hier wird die gespeicherte Methode "getContacts" aufgerufen, welche die Spalten id, firstname und lastname zurück geben soll.

Das zweite Kommando "getContact" soll nur einen Kontakt zurückgeben, daher ist der Rückgabetyp Contact. Als gespeicherte Prozedur wird getContact aufgerufen, welche den Parameter @id übergeben bekommt. Dieser Parameter wird unter dem Abschnitt queryParameter auf einen Datentyp (im Beispiel die @id auf int) gemappt. Die gespeicherte Prozedur soll neben id, firstname und lastname auch age und male zurück geben.

3 Erstellen von XSL-Templates

Nun möchten wir die Testdaten in der Datenbank mittels der onion.net Render Engine darstellen.

Das Ziel ist es eine Übersicht und eine Detailseite für die Kontakte in der Datenbank zu erstellen.

Am Ende des Tutorials sollte der Strukturbaum wie in der Abbildung aussehen.

Dafür wird für das root-Dokument das folgende Template erstellt:

```
<xsl:stylesheet
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:sql="http://onionworks.net/2006/datasources/mssql"
  xmlns:onion="http://onionworks.net/2004/data"
  version="1.0"
>
  <xsl:param
    name="id"
    c.optional="true"
    c.type="Number"
  />
  <xsl:output
    method="xml"
    omit-xml-declaration="yes"
    indent="no"
  />
  <xsl:template match="/">
    <xsl:choose>
      <xsl:when test="$id">
       <c.literalCall id="{sql:createCommandXLink('sql', 'getContact', $id)}"
method="detail" />
      </xsl:when>
      <xsl:otherwise>
        <c.literalCall
                id="{sql:createCommandXLink('sql', 'getContacts')}"
                method="list"
                delegate="{c.xlink('default')}"
        />
      </xsl:otherwise>
    </xsl:choose>
  </xsl:template>
</xsl:stylesheet>
```

default Template

Die folgenden Templates werden nicht mehr unter der Datenquelle onion.net erstellt, sondern unter der Datenquelle SQL. In der web.config wurden contentTypes definiert, welche in der Datenquelle als Datentypen erstellt werden. Das wären die Datentypen ContactList und Contact.

Für die Übersichtsseite wird die Methode "list" für den Datentyp ContactList aufgerufen. Das Template listet die Ergebnisse der Abfrage auf.

```
<xsl:stylesheet
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:onion="http://onionworks.net/2004/data"
  version="1.0"
>
  <xsl:param name="delegate" />
  <xsl:output
    method="xml"
    omit-xml-declaration="yes"
    indent="no"
  />
  <xsl:template match="/">
    <table>
      <tr>
        <th>ID</th>
        <th>Firstname</th>
        <th>Lastname</th>
      </tr>
      <xsl:for-each select="result/row">
        <tr>
          <td>
            <xsl:value-of select="@id" />
          </td>
          <td>
            <a href="{c.literalUri($delegate, 'id', @id)}">
              <xsl:value-of select="@firstname" />
            </a>
          </td>
          <td>
            <a href="{c.literalUri($delegate, 'id', @id)}">
              <xsl:value-of select="@lastname" />
            </a>
          </td>
        </tr>
      </xsl:for-each>
    </table>
  </xsl:template>
</xsl:stylesheet>
```

Zum Schluss erstellen wir noch das Template für die Detailansicht. Die Methode "detail" wird nun für den Datentyp Contact aufgerufen.

```
<xsl:stylesheet
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:onion="http://onionworks.net/2004/data"
  version="1.0"
>
  <xsl:output
    method="xml"
    omit-xml-declaration="yes"
    indent="no"
  />
  <xsl:template match="/result">
    <xsl:for-each select="row">
      <h1>
        <xsl:value-of select="concat(@firstname, ' ', @lastname)" />
      </h1>
      <ul>
        <li>Age:
          <xsl:value-of select="@age" />
        </li>
        <li>Male:
          <xsl:value-of select="@male" />
        </li>
      </ul>
    </xsl:for-each>
  </xsl:template>
</xsl:stylesheet>
```