Vorwort

Fast vier Jahre sind nun vergangen (wenn man von dem »BI-Release« SQL Server 2008 R2 absieht), bis nun endlich wieder eine wirklich neue Version von SQL Server erschienen ist. Von vielen bereits für 2011 erwartet, hat es doch bis Ende März 2012 gedauert, bis die finale Version von SQL Server »Denali« (so der Arbeitstitel) fertiggestellt war und nun unter dem Namen SQL Server 2012 erhältlich ist. Wenn man eher ein ungeduldiger Typ ist, kann man sich fragen, warum es so lange gedauert hat. Ich bin eher der Ansicht, dass gerade ein Produkt wie ein relationales Datenbank-Management-System, das auch vielfach für geschäftskritische Anwendungen produktiv eingesetzt wird, immer eine gewisse Zeit zum Reifen braucht. Microsoft hat einen extremen Entwicklungs- und Testaufwand betrieben, bis das Produkt nach verschiedenen Community Technology Previews und Release Candidates nun letztendlich ausgereift genug ist, um den bereits ebenfalls sehr stabilen und bewährten SQL Server 2008 R2 abzulösen. Dazu wurden große Mengen an neuen Features in nahezu allen Bereichen des komplexen Produkts integriert, die ich hier nicht vorwegnehmen will.

Und trotz der langen Zeit und den vielen Änderungen wurde SQL Server nicht komplett neu entworfen, sondern stattdessen konsequent weiterentwickelt, sodass man auch als langjähriger Anwender auf bestehendes Wissen aufbauen und dieses um die Kenntnis der neuen Features erweitern kann.

Genauso wie SQL Server immer weiterentwickelt wurde, wird auch dieses Buch ständig weiterentwickelt, das bereits in ähnlicher Form für SQL Server 2008 Express und SQL Server 2008 R2 Express verfügbar war. Ich habe mich mit dem Verlag darauf geeinigt, das Wort »Express« diesmal bewusst aus dem Buchtitel wegzulassen, da alle beschriebenen Features und Beispiele auch für die größeren Editionen von SQL Server anwendbar sind und der Text sich damit als Einstieg für alle Editionen eignet.

Trotzdem finden Sie auf der Buch-DVD wieder die Express Edition des aktuellen SQL Server, sodass Sie alle Beispiele gleich ausprobieren können, auch wenn Sie bisher keinen SQL Server installiert hatten.

Damit Sie auch wissen, wer dieses Buch geschrieben hat, möchte ich mich kurz vorstellen:

Ich beschäftige mit seit etwa 1995 mit dem SQL Server (damals noch in der Version 6.0) und habe das Produkt seitdem in seinen vielfältigen Facetten kennengelernt. Anfangs in Form von Performance-Untersuchungen für meine Diplomarbeit zum Thema »Optimierung von Datenbankanwendungen«, mit der ich im Jahr 1996 mein Informatik-Studium an der FH Darmstadt erfolgreich abschloss. Anschließend aus der Sicht als Datenbankadministrator für ein großes Unternehmen der Reisebranche. Darauf wechselte ich in ein mittelständisches Softwarehaus, in dem mich der SQL Server sowohl aus der Sicht eines Administrators als auch aus der eines Anwendungsentwicklers stets begleitete. Seit 2007 bin ich nun als Senior Consultant für Logica – ein großes international agierendes Beratungsunternehmen – tätig und bin spätestens seit diesem Zeitpunkt völlig auf den SQL Server fokussiert. Sofern es die Projekte zulassen, bin ich auch immer wieder mal als Sprecher auf diversen Fachkonferenzen (z.B. BASTA! & SQLCON) sowie in der offiziellen SQL Server User Community PASS (Professional Association for SQL Server) aktiv.

Und nach all dieser Zeit fasziniert es mich nach wie vor, dass es immer wieder Neues beim SQL Server zu entdecken gibt. Diese Begeisterung möchte ich Ihnen mit diesem Buch weitergeben.

Robert Panther: Richtig einsteigen: Datenbanken entwickeln mit SQL Server 2012

Bevor ich aber mit dem eigentlichen Buchtext beginne, will ich noch ein Dankeschön an verschiedene Personen loswerden:

Beginnen möchte ich mit Sylvia Hasselbach von Microsoft Press, die dieses Buch betreut und Verständnis für so manche Terminverzögerung aufgebracht hat. Ein weiteres Dankeschön geht an Sandra Michel, die dasselbe für die vorangegangenen Fassungen des Buchtextes getan hat. Ebenso gilt mein Dank Rainer G. Haselier, der mich sowohl als Fachlektor unterstützt hat als auch für den Satz des Buchtextes verantwortlich war.

Des Weiteren bedanke ich mich natürlich bei allen Kunden, Kollegen und Freunden, die mich immer wieder in Gesprächen oder in Form von Lesermails mit interessanten Ideen und Anregungen für dieses Buch versorgt haben.

Vor allem aber bedanke ich mich bei meiner Frau Birgit dafür, dass sie wieder einmal Verständnis dafür aufbringen konnte, dass ich für einen gewissen Zeitraum einen großen Teil meiner Freizeit für die Bearbeitung dieses Buchtextes aufbringen musste.

Robert Panther,
Königstein im April 2012

Kapitel 11

Benutzer, Rollen und Rechte

In diesem Kapitel lernen Sie

- wodurch sich Anmeldungen und Benutzer voneinander unterscheiden
- welche Authentifizierungsmodi es gibt
- wie Sie Benutzern Rechte erteilen oder entziehen
- wie Sie die Rechtevergabe durch Rollen vereinfachen können
- wozu die neuen Contained Databases verwendet werden können
- wie Sie Schemas für einfachere Rechtevergabe und bessere Übersicht in der Datenbank verwenden

11.1 Das SQL Server-Rechtesystem

Bisher haben wir ausschließlich mit maximalen Berechtigungen auf dem gesamten Datenbankserver gearbeitet, da der Benutzer verwendet wurde, der den Datenbankserver auch installiert hat. Das ist sicherlich völlig ausreichend, wenn Sie die Datenbank bzw. die dazugehörende Anwendung nur von einem Benutzer verwendet wird. Auch während der Entwicklung einer Anwendung kann man eine Zeit lang so verfahren. Aber spätestens dann, wenn mehrere Benutzer aktiv mit der Datenbank arbeiten oder die Datenbankanwendung sogar an die Endanwender verteilt wird, sollte man sich ernsthaft Gedanken über ein ausgefeilteres Sicherheitskonzept machen.

SQL Server unterscheidet dabei zwischen Anmeldungen (Login) und Datenbankbenutzern (Database User). Beide zusammen (insbesondere aber die Anmeldungen) ergeben die Authentifizierung, mit der die Frage geklärt wird, wer der Benutzer ist, bzw. durch Mechanismen wie Passwortabfragen etc. sichergestellt wird, dass der Benutzer auch der ist, für den er sich ausgibt.

Darauf aufbauend erfolgt dann später die Autorisierung, also die Verwaltung der Rechte für Anmeldungen und Benutzer.

11.2 Anmeldungen und Authentifizierung

Der eigentliche Verbindungsaufbau zum SQL Server geschieht über Anmeldungen. Dabei können zwei Arten der Authentifizierung genutzt werden, die von verschiedenen Systemen verwaltet werden. Bei der Windows-Authentifizierung werden Benutzer und Passwörter vom lokal installierten Betriebssystem (bzw. in Netzwerkdomänen von einem Server, der als Domänencontroller fungiert) verwaltet. Dadurch

Robert Panther: Richtig einsteigen: Datenbanken entwickeln mit SQL Server 2012

entfällt die Notwendigkeit einer separaten Anmeldung am SQL-Server, da das Betriebssystem automatisch die Information an den SQL Server weitergibt, welcher User angemeldet ist.

Bei der SQL Server-Authentifizierung dagegen werden Benutzer und deren Passwörter von SQL Server verwaltet. Die entsprechenden Daten werden verschlüsselt in Systemtabellen abgelegt. Standardmäßig ist ein Benutzer mit Namen *sa* (die Abkürzung steht für Systemadministrator) eingerichtet, der volle Rechte auf dem gesamten SQL Server hat. Das Passwort für diesen User haben Sie selbst bei der Installation von SQL Server festgelegt.

Ob die SQL Server-Authentifizierung überhaupt verfügbar ist, hängt davon ab, ob Sie bei der Installation die reine Windows-Authentifizierung oder den gemischten Modus gewählt haben. Wenn Sie sich an die in *Abschnitt 3.2, Installation* beschriebene Installationsanweisung gehalten haben, sind beide Authentifizierungsmodi verfügbar, sodass Ihnen hier alle Möglichkeiten offenstehen.

Auch jedes Mal, wenn Sie sich mit dem SQL Server verbinden, ist auszuwählen, ob die Anmeldung über SQL Server- oder Windows-Authentifizierung erfolgen soll. Im erstgenannten Fall sind Anmeldungsname und Passwort einzugeben, bei der Windows-Authentifizierung werden Benutzernamen (gegebenenfalls mit vorangestelltem Domänennamen) angezeigt und sind nicht änderbar. Wenn Sie in diesem Authentifizierungsmodus einen anderen Windows-Benutzer verwenden möchten, müssen Sie sich zuerst vom Betriebssystem abmelden und mit dem anderen Login neu verbinden, der dann beim nächsten Verbindungsaufbau mit dem SQL Server automatisch verwendet wird.

Abbildung 11.1: Herstellung einer SQL Server-Verbindung mit Windows-Authentifizierung

Erfolgt die Verbindung zum SQL Server nicht über das SQL Server Management Studio, sondern aus einer Applikation für Endanwender, so werden die Anmeldedaten normalerweise in Form einer Verbindungszeichenfolge (hier ist der englische Begriff *Connection String* eigentlich gebräuchlicher) übertragen. Diese Verbindungszeichenfolge beinhaltet unter anderem den Namen des SQL Server, den Namen der Serverinstanz (sofern nicht die Standardinstanz verwendet wird), den Authentifizierungsmodus sowie – im Falle der SQL Server-Authentifizierung – den Anmeldenamen und das dazugehörige Passwort.

Robert Panther: Richtig einsteigen: Datenbanken entwickeln mit SQL Server 2012

Wichtig: Nie den sa-Login in einer Verbindungszeichenfolge verwenden!

Selbst wenn die Anwendung volle Rechte auf die Datenbank erhalten soll, sollten Sie nie das *sa*-Passwort in einer Verbindungszeichenfolge hinterlegen. Dies würde eine große Sicherheitslücke darstellen, da der *sa*-Login Vollzugriff auf alle Datenbanken des Servers (inklusive der Systemdatenbanken) hat. Stattdessen sollte zumindest eine eigene Anmeldung für diese Anwendung erstellt werden, die alle Rechte auf die entsprechende Datenbank erhält.

Anlegen von SQL Server-Anmeldungen

Schauen wir uns nun einmal an, wie neue Anmeldungen angelegt werden, indem wir drei verschiedene SQL Server-Anmeldungen erstellen, die später unterschiedliche Rechte bekommen werden.

1. Stellen Sie im SQL Server Management Studio eine Verbindung zur lokalen Serverinstanz *SQL2012EXPRESS* her.
2. Klicken Sie nun mit der rechten Maustaste auf den Eintrag *Sicherheit/Anmeldungen* und wählen Sie die Option *Neue Anmeldung*.
3. Es erscheint das Dialogfeld zum Erstellen einer neuen Anmeldung. Im linken Bereich können Sie eine von fünf möglichen Seiten mit Einstellungen auswählen: *Allgemein*, *Serverrollen, Benutzerzuordnung, Sicherungsfähige Elemente* und *Status*. Im Moment benötigen wir davon allerdings nur die erste Seite.

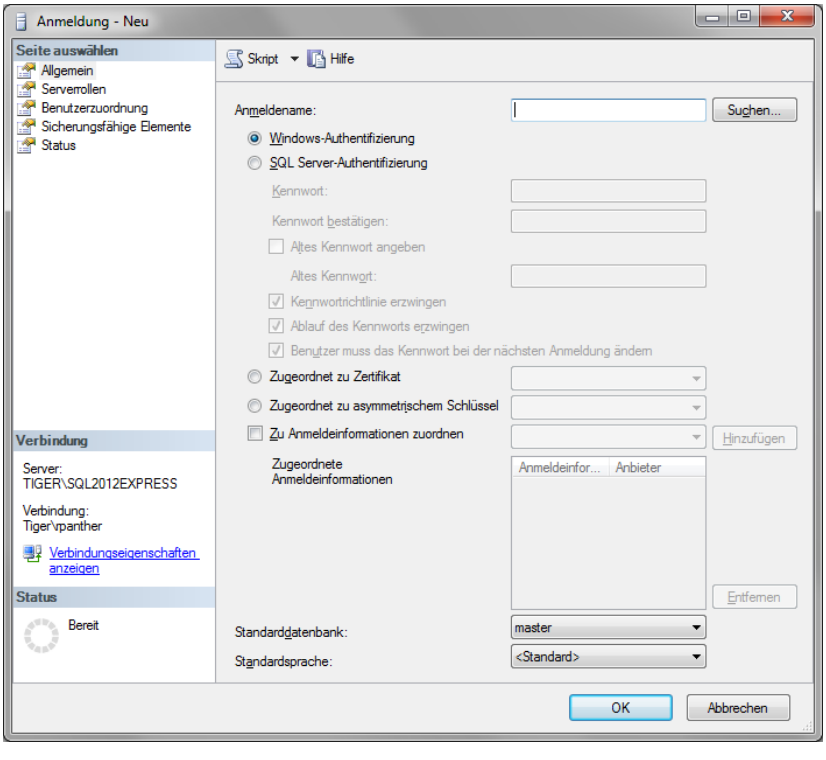

Abbildung 11.2: Das Dialogfeld zum Erstellen einer neuen Anmeldung

Robert Panther: Richtig einsteigen: Datenbanken entwickeln mit SQL Server 2012

4. Nehmen Sie auf der Seite *Allgemein* folgende Einstellungen vor:
   - *Anmeldename*: **MediaBaseReadWrite**
   - *SQL Server-Authentifizierung*
   - *Kennwort*: **mbrw**
   - *Kennwortrichtlinie erzwingen*: nein
   - *Standarddatenbank*: master
   - *Standardsprache*: <Standard>

   Schließen Sie dann das Dialogfeld über die *OK*-Schaltfläche und die Anmeldung wird angelegt.
5. Legen Sie auf dieselbe Weise Anmeldungen mit Namen *MediaBaseReadOnly* (Kennwort: *mbro*) und *MediaBaseAudioOnly* (Kennwort: *mbao*) an.

Wichtig: Keine zu einfachen Passwörter verwenden

Der Einfachheit halber wurden für die Beispielanmeldungen in diesem Kapitel sehr kurze Passwörter verwendet. In der Praxis sollte man dies natürlich nicht tun, sondern stattdessen längere Passwörter verwenden, die neben Kleinbuchstaben auch Großbuchstaben und eventuell ein paar Sonderzeichen oder Ziffern enthalten.

Das Erstellen von Anmeldungen ist natürlich auch mit SQL-Anweisungen durchführbar. Dazu wird die `CREATE LOGIN`-Anweisung der sogenannten Data Control Language (DCL) verwendet. Zum Erstellen der oben genannten drei Anmeldungen wäre das folgende SQL-Skript auszuführen:

```
USE [master]
GO
CREATE LOGIN [MediaBaseReadWrite]
WITH PASSWORD=N'mbrw', DEFAULT_DATABASE=[master], CHECK_EXPIRATION=OFF, CHECK_POLICY=OFF
GO
CREATE LOGIN [MediaBaseReadOnly]
WITH PASSWORD=N'mbro', DEFAULT_DATABASE=[master], CHECK_EXPIRATION=OFF, CHECK_POLICY=OFF
GO
CREATE LOGIN [MediaBaseAudioOnly]
WITH PASSWORD=N'mbao', DEFAULT_DATABASE=[master], CHECK_EXPIRATION=OFF, CHECK_POLICY=OFF
GO
```

Bevor wir uns damit befassen, diesen Anmeldungen auch Datenbankbenutzer zuzuordnen, werfen wir noch einen Blick auf die Windows-Authentifizierung.

Windows-Benutzer und -Gruppen als Anmeldungen anlegen

Damit Windows-Benutzer als Anmeldungen verwendbar sind, müssen diese auch als solche angelegt werden. Das Vorgehen dazu entspricht weitgehend dem zum Anlegen von Anmeldungen für die SQL Server-Authentifizierung. Um auch dies auszuprobieren, legen wir nun einen lokalen Windows-Benutzer im Betriebssystem an[1] und erstellen anschließend dafür eine SQL Server-Anmeldung:

[1] Die Beschreibung für das Anlegen eines Betriebssystembenutzers orientiert sich am Betriebssystem Windows 7. Bei Verwendung einer anderen Windows-Variante kann das Verfahren leicht abweichen.

Robert Panther: Richtig einsteigen: Datenbanken entwickeln mit SQL Server 2012
Copyright 2012 O'Reilly Verlag GmbH & Co. KG, ISBN 978-3-86645-219-0

Hinweis: Lokale Admin-Rechte erforderlich

Damit das Anlegen eines Betriebssystembenutzers möglich ist, müssen Sie über die entsprechenden Betriebssystemrechte verfügen. Dies ist beispielsweise dann der Fall, wenn Ihr Betriebssystembenutzer der Gruppe lokale Administratoren angehört.

1. Öffnen Sie das Windows-Startmenü und rufen Sie dort die *Systemsteuerung* auf.
2. Klicken Sie in der *Systemsteuerung* auf den Punkt *Benutzerkonten*.
3. Es erscheint ein Dialogfeld, in dem Sie diverse Einstellungen für den aktuell angemeldeten Benutzer vornehmen können. Im unteren Bereich befindet sich jedoch auch eine Option *Anderes Konto verwalten*, die Sie nun anklicken.
4. Klicken Sie im folgenden Dialogfeld auf die Option *Neues Konto erstellen*.
5. Geben Sie nun den Benutzernamen *mmustermann* ein. Den bereits ausgewählten Kontotyp *Standardbenutzer* können Sie beibehalten.

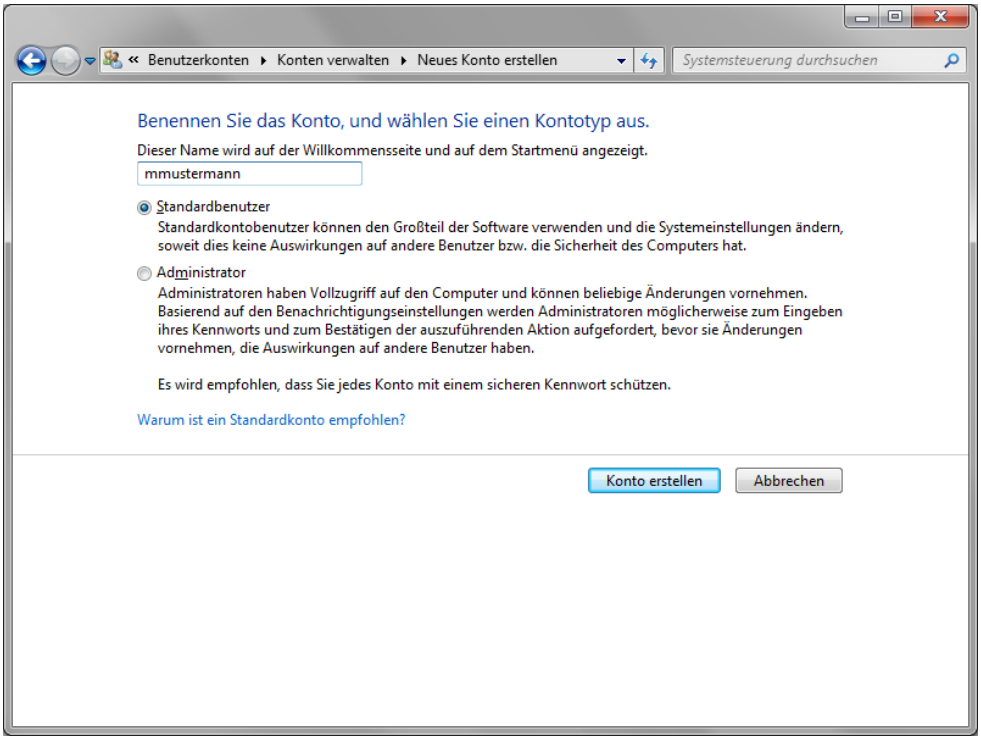

Abbildung 11.3: Das Dialogfeld zum Erstellen eines neuen Betriebssystembenutzers

6. Klicken Sie anschließend auf *Konto erstellen,* um den Benutzer anzulegen.

   Je nach Betriebssystem können Sie auf ähnlichem Weg auch Benutzergruppen erstellen, um später auch auf Gruppenebene Zugriff erteilen zu können. Bei einigen Editionen von Windows 7 ist dies jedoch nicht mehr ohne Weiteres möglich, sodass wir an dieser Stelle darauf verzichten.
7. Anschließend können Sie die Kontoverwaltung wieder schließen.

Nachdem der Betriebssystembenutzer erstellt ist, wird noch eine SQL Server-Anmeldung benötigt.

1. Stellen Sie dazu im SQL Server Management Studio eine Verbindung zur lokalen Serverinstanz *SQL2012EXPRESS* her.
2. Klicken Sie mit der rechten Maustaste auf den Eintrag *Sicherheit/Anmeldungen* und wählen Sie den Befehl *Neue Anmeldung*.
3. Behalten Sie die Voreinstellung *Windows-Authentifizierung* bei und klicken Sie auf die hinter dem Eingabefeld *Anmeldename* stehende Schaltfläche *Suchen*.
4. Es erscheint ein weiteres Dialogfeld, bei dem der aktuelle Rechnername als Suchpfad voreingestellt ist. Geben Sie als Objektname **mmustermann** ein und klicken Sie auf die Schaltfläche *Namen überprüfen*. Wenn Sie sich nicht vertippt haben, erscheint im Eingabefeld dann die Kombination aus Rechnernamen und Windows-Benutzernamen (bei Ihnen wird anstelle des Rechnernamens *TIGER* sicherlich eine andere Bezeichnung stehen).

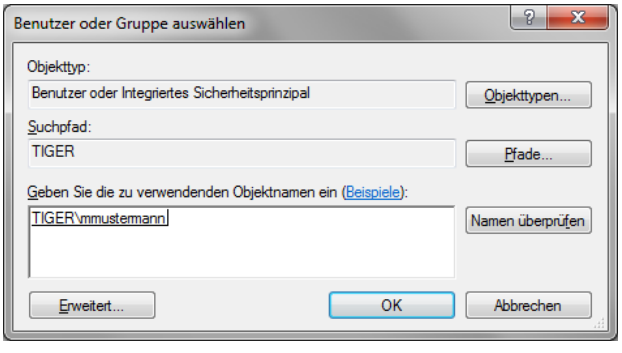

Abbildung 11.4: Eingabe eines Betriebssystembenutzers

5. Klicken Sie anschließend zweimal auf *OK* und die neue Anmeldung ist erstellt.

Das Anlegen der Anmeldungen für Windows-Benutzer und Windows-Benutzergruppe können Sie alternativ auch mit den folgenden SQL-Anweisungen durchführen (beachten Sie, dass Sie dazu den Rechnernamen `TIGER` durch den eigenen Rechnernamen ersetzen):

```
USE [master]
GO
CREATE LOGIN [TIGER\mmustermann] FROM WINDOWS WITH DEFAULT_DATABASE=[master]
GO
```

Anmeldungen testen

Nun können Sie die neu angelegten Anmeldungen einmal ausprobieren. Um die auf Windows-Authentifizierung basierenden Anmeldungen zu testen, gehen Sie wie folgt vor:

1. Schließen Sie alle offenen Anwendungen und melden Sie sich über das Windows-Startmenü vom Betriebssystem ab (oder nutzen Sie die Option *Benutzer wechseln*).
2. Melden Sie sich nun neu an und verwenden Sie dazu den lokalen Betriebssystembenutzer *mmustermann*. Falls Sie sich beim vorigen Mal mit einem Netzwerkbenutzer angemeldet haben, müssen Sie im Feld *Anmelden an* (das eventuell noch über die Schaltfläche *Optionen* eingeblendet werden

Robert Panther: Richtig einsteigen: Datenbanken entwickeln mit SQL Server 2012

muss) den lokalen Computer auswählen. Alternativ können Sie auch vor dem Benutzernamen (mit einem Backslash getrennt) den Namen des lokalen Rechners angeben. Auf meinem Datenbankentwicklungsrechner wäre dies beispielsweise **TIGER\mmustermann**. Dadurch stellen Sie sicher, dass der Benutzer *mmustermann* auf dem lokalen Rechner und nicht in der Domäne gesucht wird, wo er sicherlich nicht existiert (er wurde ja nur lokal auf dem Rechner angelegt).

3. Öffnen Sie das SQL Server Management Studio und stellen Sie eine Verbindung zur lokalen Serverinstanz *SQL2012EXPRESS* her. Benutzen Sie dabei die Windows-Authentifizierung.
4. Probieren Sie nun im Objekt-Explorer aus, auf welche Bereiche Sie Zugriff haben. Im Zweig *Datenbanken* sehen Sie zwar die Datenbank *MediaBase*, wenn Sie diesen Zweig allerdings weiter aufklappen möchten, erhalten Sie eine Fehlermeldung, die besagt, dass der Zugriff auf die MediaBase-Datenbank nicht möglich ist. Wenn Sie über die rechte Maustaste versuchen, das *Eigenschaften*-Fenster der Datenbank zu öffnen, wird die Fehlermeldung noch etwas deutlicher.

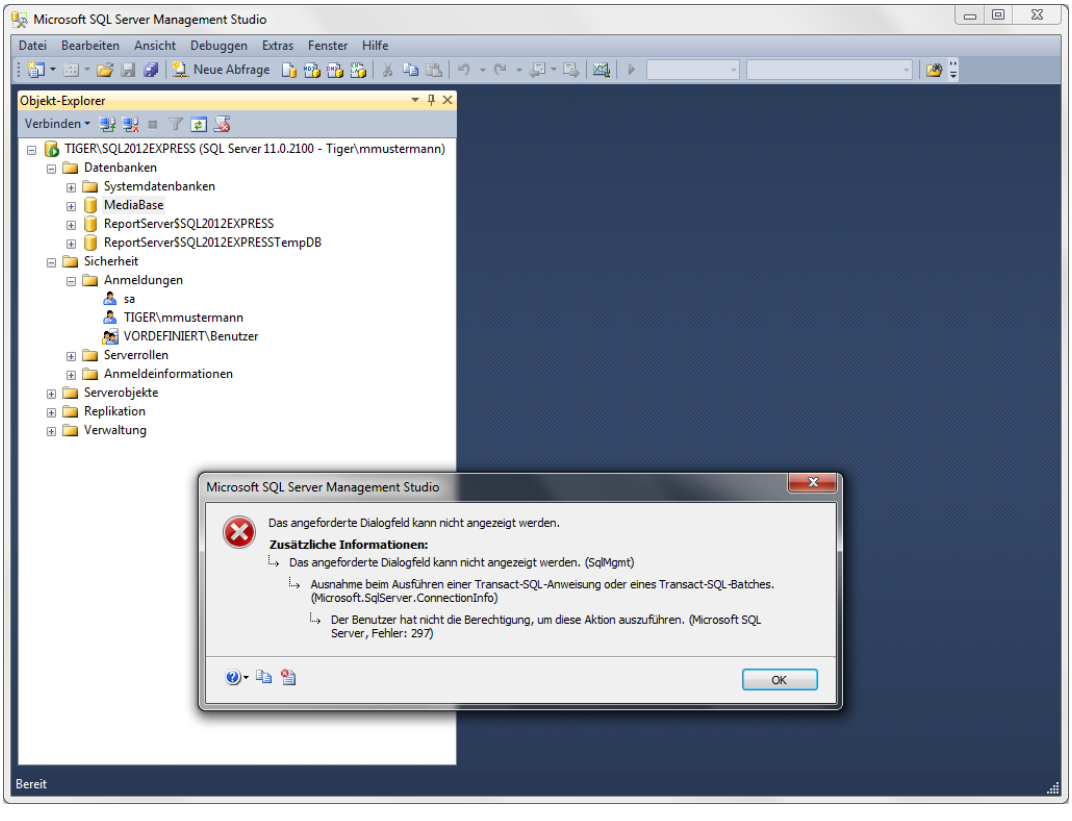

Abbildung 11.5: Fehlgeschlagener Versuch, auf die Datenbank *MediaBase* zuzugreifen

Der Grund hierfür liegt darin, dass der Windows-Benutzer *mmustermann* zwar als Anmeldung angelegt wurde (sonst wäre bereits der Verbindungsaufbau zum SQL Server mit einer entsprechenden Fehlermeldung gescheitert), diese Anmeldung aber noch keine Zugriffsrechte auf irgendwelche Datenbanken hat.

Robert Panther: Richtig einsteigen: Datenbanken entwickeln mit SQL Server 2012

5. Schließen Sie das SQL Server Management Studio, melden Sie sich vom Rechner ab und dann mit dem Windows-Benutzer an, mit dem Sie den SQL Server installiert haben.

Hinweis: Lokale Admin-Rechte erforderlich

Wie der Screenshot in Abbildung 11.5 zeigt, sind im Objekt-Explorer auch nicht alle Anmeldungen sichtbar, sondern – abgesehen vom Systemadministrator (*sa*) – nur die gerade verwendete Anmeldung (*mmustermann*) selbst sowie die Benutzer und Gruppen, auf die die verwendete Anmeldung Rechte hat. Das ist in diesem Fall lediglich die Standardgruppe *VORDEFINIERT\Benutzer*, der automatisch jeder erstellte Benutzer zugeordnet wird.

Sie können das gerade durchgeführte Experiment noch einmal über SQL Server-Authentifizierung wiederholen, indem Sie bei der Anmeldung am SQL Server die SQL Server-Authentifizierung wählen und dazu eine der frisch erstellten SQL-Anmeldungen (*MediaBaseReadWrite*, *MediaBaseReadOnly* oder *MediaBaseAudioOnly*) mit dem entsprechenden Passwort verwenden. Das Ergebnis beim versuchten Zugriff auf die Datenbank *MediaBase* wird dasselbe sein wie zuvor. Da auch für diese Anmeldungen noch kein Zugriff auf die Datenbank eingerichtet wurde, fehlen auch hier die entsprechenden Berechtigungen.

Im Objekt-Explorer sind unter *Sicherheit* dann sogar nur noch die verwendete Anmeldung selbst sowie der Systemadministrator (*sa*) zu sehen, da die Windows-Authentifizierung nun nicht aktiv ist.

Hinweis: Mehrere Verbindungen in einer Management Studio-Session

Um die Verbindung mit anderen SQL Server-Anmeldungen zu testen, müssen Sie das SQL Server Management Studio nicht einmal verlassen, da dieses mehrere Verbindungen mit verschiedenen Anmeldungen zulässt. Klicken Sie einfach im Objekt-Explorer auf *Verbinden/Datenbankmodul* und wählen Sie im folgenden Dialogfeld *SQL Server-Authentifizierung* aus. Nach Eingabe des Anmeldenamens und Kennworts erscheint im Objekt-Explorer eine zweite Verbindung, die nun die gerade eingegebene Anmeldung verwendet.

11.3 Verwalten von Datenbankbenutzern

Damit eine Anmeldung Zugriff auf eine Datenbank erhält, muss dieser ein Datenbankbenutzer zugeordnet werden, der eventuell vorher noch erstellt werden muss und entsprechende Rechte auf der Datenbank erhält. Die Datenbankbenutzer sind im Objekt-Explorer bei der jeweiligen Datenbank im Zweig *Sicherheit/Benutzer* zu finden. Hier sind bereits einige Benutzer vordefiniert:

- *dbo* – Database Owner (Vollzugriff auf die gesamte Datenbank)
- *guest* – Gast (lediglich Rechte, sich mit der Datenbank zu verbinden)
- *INFORMATION_SCHEMA* – (wird intern verwendet)
- *sys* – (wird intern verwendet)

Hier könnten Sie durch einen Klick mit der rechten Maustaste auf *Benutzer* und Auswahl von *Neuer Benutzer* einen neuen Datenbankbenutzer anlegen, um diesen später einer Anmeldung zuzuordnen. Doch es gibt auch einen komfortableren Weg, den wir nun ausprobieren wollen:

Robert Panther: Richtig einsteigen: Datenbanken entwickeln mit SQL Server 2012
Copyright 2012 O'Reilly Verlag GmbH & Co. KG, ISBN 978-3-86645-219-0

1. Stellen Sie im SQL Server Management Studio unter Verwendung der Windows-Authentifizierung eine Verbindung zur lokalen Serverinstanz *SQL2012EXPRESS* her.
2. Öffnen Sie im Objekt-Explorer den Zweig *Sicherheit/Anmeldungen*.
3. Klicken Sie mit der rechten Maustaste auf den Eintrag *MediaBaseReadWrite* und wählen Sie die Option *Eigenschaften* aus (alternativ dazu wäre auch ein Doppelklick auf den Anmeldungsnamen möglich).
4. Es erscheint das Dialogfeld *Anmeldungseigenschaften*, das weitgehend dem Dialogfeld entspricht, mit dem Sie ursprünglich die Anmeldung erstellt haben. Ändern Sie auf der Seite *Allgemein* die Standarddatenbank auf *MediaBase*. Damit wird zwar das Problem der fehlenden Berechtigung noch nicht gelöst, dies sorgt aber dafür, dass der SQL Server bei Verwendung dieser Anmeldung automatisch versucht, sich mit der Datenbank *MediaBase* zu verbinden.
5. Wechseln Sie nun zur Seite *Benutzerzuordnung*. Hier sind alle verfügbaren Datenbanken aufgelistet und Sie können durch Setzen eines Häkchens Zugriff auf die jeweilige Datenbank erteilen. In der Spalte *Benutzer* wird der entsprechende Datenbankbenutzer zugeordnet. Standardmäßig wird hier in dem Moment, in dem Sie die Datenbank zuordnen, der Name der Anmeldung eingetragen. Dadurch wird implizit ein Datenbankbenutzer mit demselben Namen wie die SQL Server-Anmeldung erstellt, sobald Sie auf *OK* klicken.

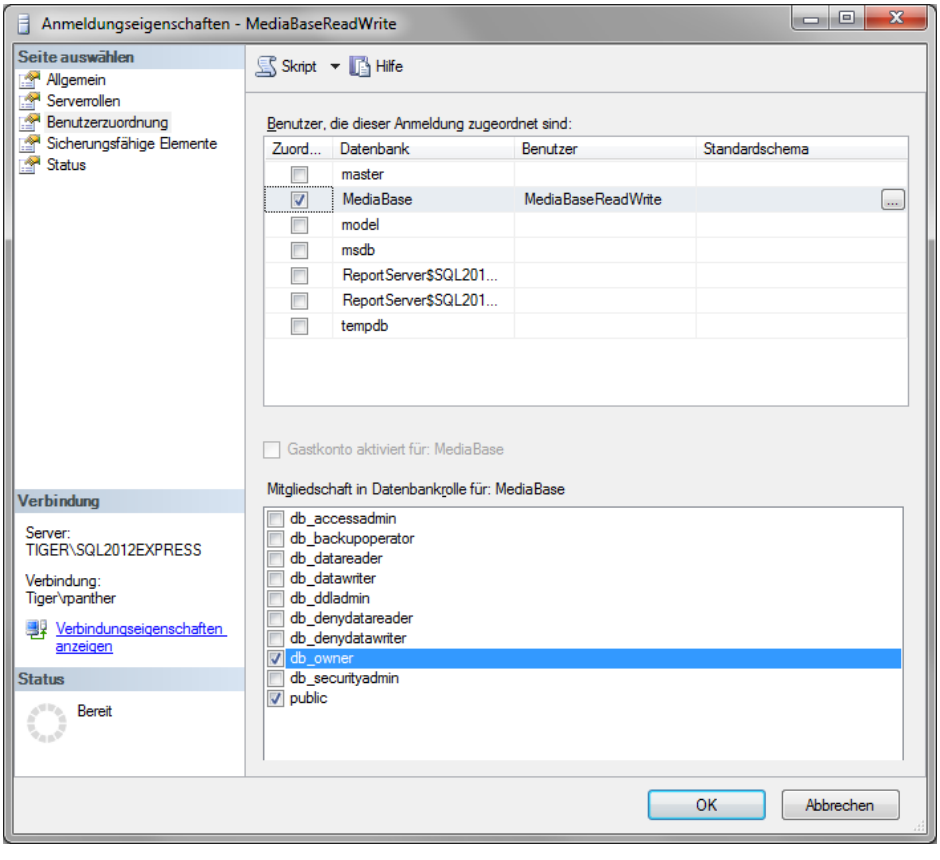

Abbildung 11.6: Die Benutzerzuordnung für die Anmeldung *MediaBaseReadWrite*

Robert Panther: Richtig einsteigen: Datenbanken entwickeln mit SQL Server 2012

6. Im unteren Bereich des Dialogfeldes erfolgt die Rollenzuordnung, über die auf einfachem Weg eine ganze Menge an Berechtigungen erteilt werden kann. Standardmäßig ist hier die Rolle *public* ausgewählt, die lediglich eine Verbindung mit der Datenbank gewährt. Klicken Sie hier zusätzlich noch die Rolle *db_owner* an, damit der Benutzer *MediaBaseReadWrite* Vollzugriff auf die Datenbank *MediaBase* erhält.
7. Klicken Sie auf *OK*, um die Benutzerzuordnung durchzuführen. Anschließend können Sie im Objekt-Explorer unter *MediaBase/Sicherheit/Benutzer* nachprüfen, dass der Datenbankbenutzer wirklich angelegt wurde.
8. Erstellen Sie auf demselben Weg Datenbankbenutzer für die anderen Anmeldungen und ordnen Sie diesen folgende Rollen zu:
   - *MediaBaseReadOnly* (Rollen: *public, db_datareader)*
   - *MediaBaseAudioOnly* (Rollen: *public*)
   - *TIGER\mmustermann* (Rollen: *public*, *db_owner*)

Den gesamten Vorgang (Ändern der Standarddatenbank, Anlegen eines Datenbankbenutzers und Zuordnung desselben zu einer Anmeldung sowie Hinzufügen der Rolle *db_datareader* für diesen Benutzer) können Sie alternativ auch mit den folgenden SQL-Anweisungen durchführen:

```
USE [master]
GO
ALTER LOGIN [MediaBaseReadOnly] WITH DEFAULT_DATABASE=[MediaBase], DEFAULT_LANGUAGE=[Deutsch],
CHECK_EXPIRATION=OFF, CHECK_POLICY=OFF
ALTER LOGIN [MediaBaseAudioOnly] WITH DEFAULT_DATABASE=[MediaBase], DEFAULT_LANGUAGE=[Deutsch],
CHECK_EXPIRATION=OFF, CHECK_POLICY=OFF
ALTER LOGIN [TIGER\mmustermann] WITH DEFAULT_DATABASE=[MediaBase], DEFAULT_LANGUAGE=[Deutsch]
GO
USE [MediaBase]
GO
CREATE USER [MediaBaseReadOnly] FOR LOGIN [MediaBaseReadOnly]
CREATE USER [MediaBaseAudioOnly] FOR LOGIN [MediaBaseAudioOnly]
CREATE USER [TIGER\mmustermann] FOR LOGIN [TIGER\mmustermann]
GO
EXEC sp_addrolemember N'db_datareader', N'MediaBaseReadOnly'
EXEC sp_addrolemember N'db_owner', N'TIGER\mmustermann'
GO
```

Für die Rollenzuordnung wird in diesem Skript die gespeicherte Systemprozedur `sp_addrolemember` verwendet. Dies geschieht allerdings nicht für die Rolle *public*, da diese automatisch zugeordnet wird, sobald ein Datenbankbenutzer für die Anmeldung erstellt wird.

Nun können Sie ausprobieren, ob der Zugriff über die gerade erstellten Datenbankbenutzer funktioniert. Der Einfachheit halber nutzen wir nun hierzu die SQL Server-Authentifizierung:

1. Stellen Sie im SQL Server Management Studio unter Verwendung der SQL Server-Authentifizierung eine Verbindung zur lokalen Serverinstanz *SQL2012EXPRESS* her. Verwenden Sie dabei die Anmeldung *MediaBaseReadOnly* mit dem entsprechenden Passwort.
2. Öffnen Sie nun im Objekt-Explorer den Zweig *Datenbanken/MediaBase/Tabellen*. Dort werden nun alle Tabellen angezeigt, weil Sie über die Rolle *db_datareader* Lesezugriff auf alle Daten der Datenbank haben.

Robert Panther: Richtig einsteigen: Datenbanken entwickeln mit SQL Server 2012

3. Klicken Sie die Tabelle *dbo.Buch* mit der rechten Maustaste an und wählen Sie die Option *Oberste 200 Zeilen bearbeiten* aus, worauf der Inhalt der Tabelle angezeigt wird. Wenn Sie nun versuchen, eines der Felder zu bearbeiten, erhalten Sie beim Versuch, diese Änderung zu speichern, eine Fehlermeldung, die Sie darauf hinweist, dass die Schreibrechte nicht erteilt wurden.

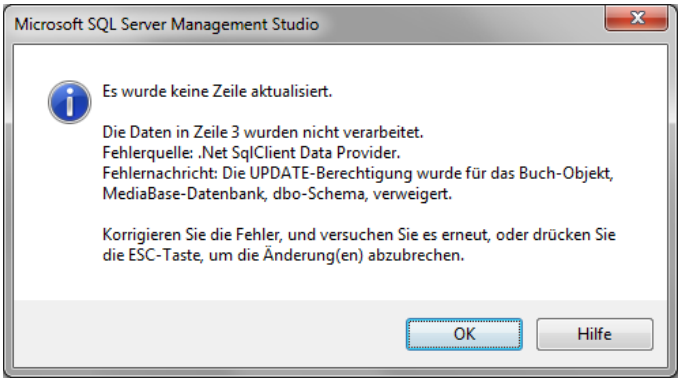

Abbildung 11.7: Fehlende Schreibrechte

4. Probieren Sie anschließend dieselbe Aktion mit der Anmeldung *MediaBaseReadWrite* aus und das Speichern der Daten sollte funktionieren.
5. Melden Sie sich danach mit der Anmeldung *MediaBaseAudioOnly* an. Jetzt wird im Objekt-Explorer zur Datenbank *MediaBase* keine Tabelle angezeigt, da keine Leseberechtigungen für diese Anmeldung existieren.

Für die meisten Anwendungsfälle – insbesondere im Umfeld von SQL Server Express Edition – reicht eine indirekte Erteilung von Rechten durch die Zuweisung von vordefinierten Rollen für die entsprechenden Datenbanken aus. Bei Bedarf können Sie die Berechtigungen aber auch detaillierter erteilen bzw. verweigern, wie die folgenden Seiten zeigen.

11.4 Rechte und Rollen

Bei der Vergabe von Rechten (und Zuordnung von Rollen) wird zwischen zwei Arten von Rechten unterschieden. Das eine sind die Berechtigungen zum Zugriff auf Datenbankobjekte wie Tabellen, Sichten etc., die für Datenbankbenutzer erteilt werden. Das andere sind allgemeine Berechtigungen, die für den gesamten Server gelten (sogenannte Serverrechte) und daher für Anmeldungen erteilt werden. Für beide Varianten existieren auch entsprechende Rollen, mit deren Hilfe sich einheitliche Berechtigungen für eine Gruppe von Anmeldungen bzw. Datenbankbenutzern vereinfachen lassen.

Serverrechte und -rollen

Serverrollen sind fest definiert und ließen sich bis einschließlich SQL Server 2008 R2 nicht erweitern oder ändern. Mit SQL Server 2012 wurden die benutzerdefinierten Serverrollen eingeführt, die sich jedoch nicht über die Management Studio-Oberfläche, sondern ausschließlich über T-SQL erstellen

Robert Panther: Richtig einsteigen: Datenbanken entwickeln mit SQL Server 2012

lassen. Da man jedoch im Allgemeinen gut mit den vorgegebenen Serverrollen auskommt, werde ich mich im Folgenden auf ebendiese beschränken.

Folgende Rollen stehen in diesem Zusammenhang zur Auswahl:

Tabelle 11.1: Übersicht der Serverrollen

Rolle	Bedeutung und enthaltene Berechtigungen
bulkadmin	Durchführen von Masseneinfügeoperationen (`bcp` bzw. `BULK INSERT`)
dbcreator	Erstellen, Ändern und Wiederherstellen von eigenen Datenbanken
diskadmin	Verwalten von Datenträgerdateien
processadmin	Beenden von SQL Server-Prozessen
public	Standardrolle ohne besondere Rechte
securityadmin	Verwalten von Anmeldungen und Berechtigungen
serveradmin	Konfiguration und Herunterfahren des Servers
setupadmin	Hinzufügen von Verbindungsservern
sysadmin	Systemadministrator (Vollzugriff auf den gesamten Server)

Die Zuordnung einer Anmeldung zu einer Rolle können Sie im Dialogfeld *Anmeldungseigenschaften* auf der Seite *Serverrollen* vornehmen. Alternativ dazu können Sie auch eine gespeicherte Systemprozedur nutzen. Die folgende SQL-Anweisung würde beispielsweise der Anmeldung *MediaBaseReadWrite* die Serverrolle *diskadmin* zuordnen:

```
EXEC master..sp_addsrvrolemember @loginame = N'MediaBaseReadWrite', @rolename = N'diskadmin'
GO
```

Neben den vordefinierten Serverrollen lassen sich auch explizite Serverrechte an einzelne Anmeldungen erteilen. Dies geschieht wiederum über das Dialogfeld *Anmeldungseigenschaften* des jeweiligen Benutzers, nun aber auf der Seite *Sicherungsfähige Elemente*. Hier können Sie über die *Suchen*-Schaltfläche bestimmte Objekte wie beispielsweise Server, Endpunkte und Anmeldungen auswählen, um dann auf diese Objekte einzelne Rechte zu *erteilen* oder um diese explizit zu *verweigern*. Die Variante *Mit Erteilung* steht dafür, dass die jeweilige Anmeldung die entsprechende Berechtigung auch an andere Anmeldungen weitergeben darf. Die folgende Abbildung zeigt beispielsweise die Erteilung (mit Weitergabe) der Berechtigung *Herunterfahren* des Servers bei gleichzeitiger Verweigerung der Berechtigung *Massenvorgänge verwalten*.

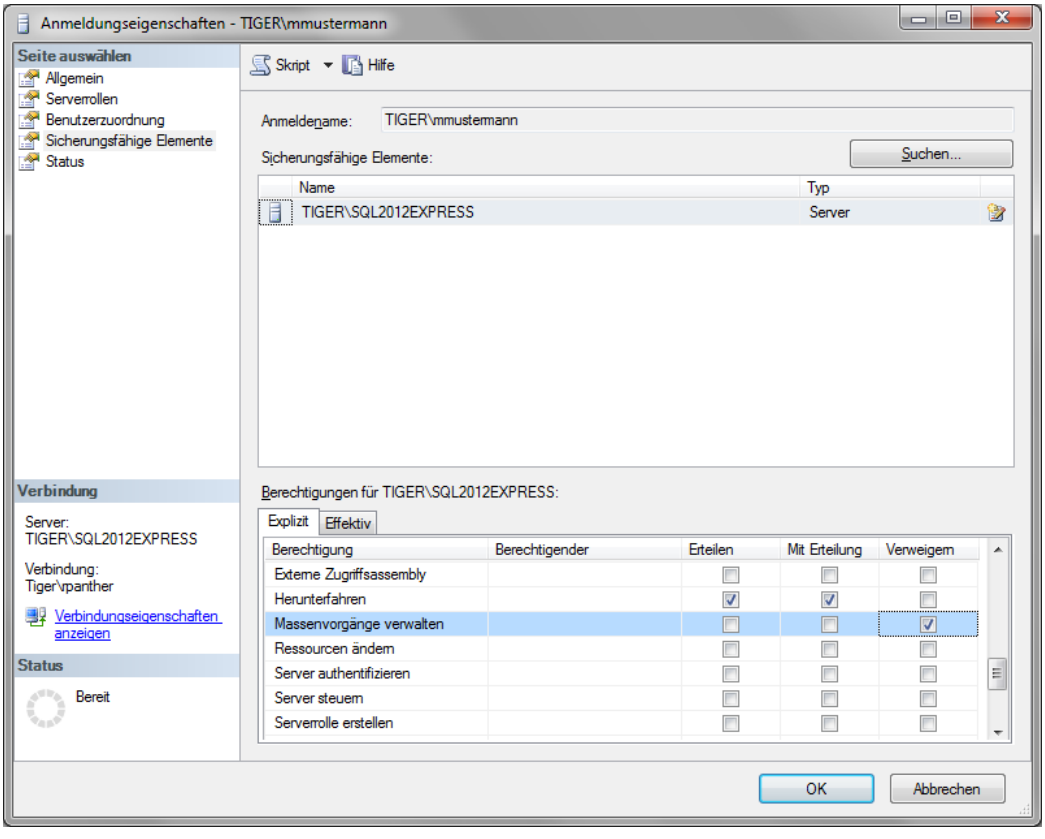

Abbildung 11.8: Erteilung von Serverrechten

Wenn Sie im unteren Bereich die Registerkarte *Effektiv* öffnen, werden die aktuellen Berechtigungen zum oben ausgewählten sicherungsfähigen Element angezeigt.

Das Erteilen und Verweigern der in Abbildung 11.8 dargestellten Rechte ist alternativ auch mit folgendem SQL-Skript ausführbar:

```
use [master]
GO
GRANT SHUTDOWN TO [TIGER\mmustermann] WITH GRANT OPTION
DENY ADMINISTER BULK OPERATIONS TO [TIGER\mmustermann]
GO
```

Best Practices: Verwendung von Serverrollen und -rechten

Die Bedeutung von Serverrollen und -rechten ist im Umfeld von SQL Server Express eher gering. Im Normalfall sind alle Anmeldungen lediglich der Rolle *public* zugeordnet und nur ein paar einzelne Anmeldungen bekommen zusätzlich noch die Rolle *sysadmin*, um auch administrative Aufgaben durchführen zu können. Einzelne Serverrechte werden selten vergeben.

Die anderen Serverrollen sowie die explizite Vergabe von einzelnen Serverrechten spielen eher in größeren Unternehmen eine Rolle, wo sich verschiedene Mitarbeiter dediziert um die Verwaltung von Berechtigungen, das Erstellen von Datenbanksicherungen etc. kümmern, ohne dabei auf andere Bereiche des SQL Server zugreifen zu müssen.

Robert Panther: Richtig einsteigen: Datenbanken entwickeln mit SQL Server 2012

Datenbankrechte und -rollen

Die Erteilung von Datenbankrechten erfolgt nach demselben Prinzip wie bei den Serverrechten, nur mit dem Unterschied, dass hier das *Eigenschaften*-Dialogfeld des entsprechenden Datenbankbenutzers verwendet wird. Auch hier gibt es eine Seite *Sicherungsfähige Elemente*, auf der Sie nach Auswahl der entsprechenden Datenbankobjekte (wie Tabellen, Sichten, Funktionen, gespeicherte Prozeduren etc.) explizit Berechtigungen auf diese erteilen (mit und ohne Weitergabe) oder verweigern können. Für Tabellen und Sichten lassen sich die Berechtigungen sogar für einzelne Spalten verwalten.

Dazu können Sie auf der Seite *Mitgliedschaft* dem Datenbankbenutzer bestehende Datenbankrollen zuordnen, die bereits über einen vordefinierten Satz von Berechtigungen verfügen.

Tabelle 11.2: Übersicht der vordefinierten Datenbankrollen

Rolle	Bedeutung und enthaltene Berechtigungen
db_accessadmin	Verwalten des Zugriffs für Windows-Anmeldungen und -Gruppen sowie SQL Server-Anmeldungen
db_backupoperator	Durchführen von Datenbanksicherungen
db_datareader	Lesezugriff auf alle Tabellen und Sichten der Datenbank
db_datawriter	Schreibzugriff auf alle Tabellen und Sichten der Datenbank
db_ddladmin	Ausführung von DDL-Anweisungen
db_denydatareader	Verweigert Lesezugriff auf alle Benutzertabellen der Datenbank
db_denydatawriter	Verweigert Schreibzugriff auf alle Benutzertabellen der Datenbank
db_owner	Datenbankbesitzer (Vollzugriff auf die gesamte Datenbank)
db_securityadmin	Verwalten von Rollenmitgliedschaften und Berechtigungen

Im Gegensatz zu den Serverrollen lassen sich die Datenbankrollen aber auch bereits bei älteren SQL Server-Versionen um selbstdefinierte Rollen erweitern, die dann – genau wie die Benutzer – detaillierte Berechtigungen erhalten können.

Probieren wir dies nun aus, indem wir eine Datenbankrolle *MediaBaseCDRole* erstellen, die den Vollzugriff auf die Tabellen *dbo.CD* und *dbo.CD_Track* ermöglicht:

1. Stellen Sie im SQL Server Management Studio unter Verwendung der Windows-Authentifizierung eine Verbindung zur lokalen Serverinstanz *SQL2012EXPRESS* her.
2. Öffnen Sie im Objekt-Explorer den Zweig *Datenbanken/MediaBase/Sicherheit/Rollen/Datenbankrollen*. Sie sehen hier die in Tabelle 11.2 dargestellten Datenbankrollen.
3. Klicken Sie mit der rechten Maustaste auf *Datenbankrollen* und wählen Sie die Option *Neue Datenbankrolle*. Es erscheint das Dialogfeld *Neue Datenbankrolle*.
4. Geben Sie als Namen für die Datenbankrolle **MediaBaseCDRole** ein und klicken Sie auf die Schaltfläche *Hinzufügen*, um der Rolle einen neuen Datenbankbenutzer zuzuordnen.
5. Geben Sie den Namen *MediaBaseAudioOnly* ein und klicken Sie auf *OK*.
6. Wechseln Sie nun zur Seite *Sicherungsfähige Elemente* und klicken Sie auf die Schaltfläche *Suchen*. Wählen Sie im Dialogfeld *Objekte hinzufügen* die Option *Alle Objekte des Typs* und klicken Sie wiederum auf *OK*. Im darauf folgenden Dialogfeld *(Objekttypen auswählen)* markieren Sie den Eintrag *Tabellen* und klicken nochmals auf *OK*.

7. Auf der Seite *Sicherungsfähige Elemente* werden nun alle Tabellen der Datenbank *MediaBase* aufgelistet. Klicken Sie hier die Tabelle *dbo.CD* an und markieren Sie darauf in der Berechtigungsliste unten die gesamte Spalte *Erteilen*. Verfahren Sie dann für die Tabelle *dbo.CD_Track* genauso.
8. Klicken Sie nun auf *OK* und die Rechte werden gemäß den gerade vorgenommenen Einstellungen erteilt.

Das SQL-Skript, mit dem Sie die Datenbankrolle alternativ erstellen können, ist diesmal etwas länger, da eine ganze Menge an Berechtigungen explizit vergeben wird:

```
USE [MediaBase]
GO
CREATE ROLE [MediaBaseCDRole]
GO
ALTER ROLE [MediaBaseCDRole] ADD MEMBER [MediaBaseAudioOnly]
GO

-- Berechtigungen für Tabelle dbo.CD erteilen
GRANT UPDATE ON [dbo].[CD] TO [MediaBaseCDRole]
GRANT ALTER ON [dbo].[CD] TO [MediaBaseCDRole]
GRANT VIEW CHANGE TRACKING ON [dbo].[CD] TO [MediaBaseCDRole]
GRANT SELECT ON [dbo].[CD] TO [MediaBaseCDRole]
GRANT TAKE OWNERSHIP ON [dbo].[CD] TO [MediaBaseCDRole]
GRANT VIEW DEFINITION ON [dbo].[CD] TO [MediaBaseCDRole]
GRANT INSERT ON [dbo].[CD] TO [MediaBaseCDRole]
GRANT DELETE ON [dbo].[CD] TO [MediaBaseCDRole]
GRANT CONTROL ON [dbo].[CD] TO [MediaBaseCDRole]
GRANT REFERENCES ON [dbo].[CD] TO [MediaBaseCDRole]

-- Berechtigungen für Tabelle dbo.CD_Track erteilen
GRANT UPDATE ON [dbo].[CD_Track] TO [MediaBaseCDRole]
GRANT ALTER ON [dbo].[CD_Track] TO [MediaBaseCDRole]
GRANT VIEW CHANGE TRACKING ON [dbo].[CD_Track] TO [MediaBaseCDRole]
GRANT SELECT ON [dbo].[CD_Track] TO [MediaBaseCDRole]
GRANT TAKE OWNERSHIP ON [dbo].[CD_Track] TO [MediaBaseCDRole]
GRANT VIEW DEFINITION ON [dbo].[CD_Track] TO [MediaBaseCDRole]
GRANT INSERT ON [dbo].[CD_Track] TO [MediaBaseCDRole]
GRANT DELETE ON [dbo].[CD_Track] TO [MediaBaseCDRole]
GRANT CONTROL ON [dbo].[CD_Track] TO [MediaBaseCDRole]
GRANT REFERENCES ON [dbo].[CD_Track] TO [MediaBaseCDRole]
GO
```

Nun können Sie ausprobieren, welche Rechte der Datenbankbenutzer *MediaBaseAudioOnly* durch seine Zugehörigkeit zur Rolle *MediaBaseCDRole* erhalten hat:

1. Stellen Sie im SQL Server Management Studio unter Verwendung der SQL Server-Authentifizierung eine Verbindung zur lokalen Serverinstanz *SQL2012EXPRESS* her. Verwenden Sie dabei die Anmeldung *MediaBaseAudioOnly* mit dem entsprechenden Passwort.
2. Öffnen Sie im Objekt-Explorer den Zweig *Datenbanken/MediaBase/Tabellen*. Dort sollten nun nur die beiden Tabellen *dbo.CD* und *dbo.CD_Track* sichtbar sein, auf die Sie mit dem verwendeten Benutzer volle Zugriffsrechte haben.

Best Practices: Verwendung von Datenbankrechten und -rollen

Wenn ein hohes Maß an Sicherheit notwendig ist, hat sich folgendes Vorgehen bereits in vielen Projekten bewährt:

Für die von der Anwendung verwendeten Datenbankbenutzer existiert generell kein direkter Zugriff auf die Tabellen. Lesende Zugriffe erfolgen ausschließlich über Sichten, die auch nur die Spalten zur Verfügung stellen, die in der Anwendung benötigt werden. Schreibende Zugriffe erfolgen ausschließlich über gespeicherte Prozeduren. Auf diese Art werden die Möglichkeiten der Datenmanipulation (für den Fall, dass die Anmeldedaten doch einmal in die falschen Hände geraten) sehr gering gehalten, da einerseits nur die von der Anwendung benötigten Daten zugreifbar sind und durch die Verwendung von gespeicherten Prozeduren nur fest definierte Datenänderungen möglich sind.

11.5 Contained Databases

Die Aufteilung zwischen Anmeldungen, die auf dem Server (genauer: in der *master*-Datenbank) und Datenbankbenutzern, die in der jeweiligen Datenbank gespeichert werden, bringt oft Probleme mit sich: Wenn eine Datenbank auf einem Server gesichert und anschließend auf einem anderen Server wiederhergestellt wird, kennt der neue Server eventuell nicht dieselben Anmeldungen. Die Datenbankbenutzer existieren zwar noch in der Datenbank, sind aber nicht nutzbar, da sie nicht mehr mit einer existierenden Server-Anmeldung verbunden sind. Man spricht hier von verwaisten Benutzern. Nun kann man natürlich diese Zuordnung manuell wiederherstellen, was aber einen gewissen Administrationsaufwand erfordert.

Mit SQL Server 2012 wurde jedoch eine neue Technologie eingeführt, mit der dies nicht mehr nötig ist. Die sogenannten Contained Databases (eine griffige deutsche Übersetzung gibt es dazu bisher wohl noch nicht) enthalten alle benötigten Informationen, damit diese auch ohne Server-Anmeldung nutzbar sind. Damit werden Contained Databases unabhängig vom jeweiligen Server, was auch eine Portierung in die Cloud erleichtert.

Beim Dialogfeld zum Anlegen einer Datenbank ist Ihnen vielleicht eine Einstellung namens *Einschlusstyp* aufgefallen, die standardmäßig auf *Keine* steht. Durch Ändern dieser Konfiguration auf *Teilweise* wird die Datenbank zur Contained Database (eine Option *Vollständig* gibt es hier nicht, was wohl darauf hinweist, dass einige weniger wichtige Einstellungen doch noch vom Server – nämlich aus der *master*-Datenbank – gelesen werden).

Wenn Sie nun versuchen, eine Contained Database anzulegen, werden Sie wahrscheinlich die folgende Fehlermeldung erhalten:

Robert Panther: Richtig einsteigen: Datenbanken entwickeln mit SQL Server 2012

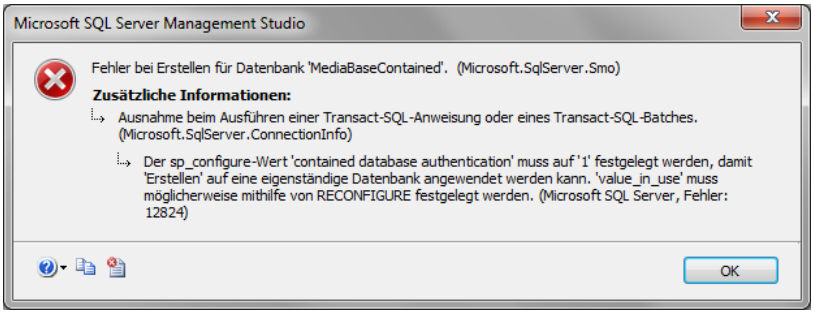

Abbildung 11.9: Fehlermeldung bei nicht aktivierter Contained Database Authentication

Der neue Authentifizierungstyp muss also zuerst noch aktiviert werden, wozu die folgenden SQL-Anweisungen notwendig sind:

```
EXEC sp_configure 'show advanced options', 1
RECONFIGURE
GO

EXEC sp_configure 'contained database authentication', 1
RECONFIGURE
GO
```

Hintergrundinfo: Safe by Default

Der Grund, warum die Technik der Contained Databases nicht standardmäßig aktiviert und die Aktivierung auch nur per SQL-Code möglich ist, liegt darin, dass diese Variante nicht dieselbe Zugriffssicherheit bietet, wie das zweistufige System mit Anmeldungen und Datenbankbenutzern. Microsoft verfolgt beim SQL Server seit geraumer Zeit die Strategie, dass dieser bereits in den Standardeinstellungen möglichst sicher sein soll, wodurch alle potenziellen Gefahrenquellen erst einmal deaktiviert sind.

Legen wir nun eine Contained Database an:

1. Stellen Sie im SQL Server Management Studio unter Verwendung der Windows-Authentifizierung eine Verbindung zur lokalen Serverinstanz *SQL2012EXPRESS* her.
2. Klicken Sie mit der rechten Maustaste im Objekt-Explorer auf den Eintrag *Datenbanken* und wählen Sie die Option *Neue Datenbank*.
3. Geben Sie auf der Seite *Allgemein* den Datenbanknamen *MediaBaseContained* ein. Die restlichen Einstellungen auf dieser Seite können beibehalten werden.
4. Ändern Sie auf der Seite *Optionen* den Einschlusstyp in *Teilweise* und klicken Sie dann auf *OK*, um die Datenbank anzulegen. Im Objekt-Explorer ist nun die neue Datenbank zu sehen.
5. Öffnen Sie im Objekt-Explorer den Zweig *Datenbanken/MediaBaseContained/Sicherheit/Benutzer* und klicken Sie mit der rechten Maustaste darauf, um die Option *Neuer Benutzer* auszuwählen.
6. Bei der Option *Benutzertyp* gibt es nun mit *SQL-Benutzer mit Kennwort* eine neue Auswahlmöglichkeit, die auch bereits voreingestellt ist.
7. Geben Sie als Benutzernamen *MediaBaseContainedUser* und als Passwort *mbcu* an.

8. Wählen Sie auf der Seite *Mitgliedschaft* die Rolle *db_owner* aus und klicken Sie anschließend auf *OK*, um den Benutzer anzulegen.
9. Trennen Sie nun die Verbindung zur Datenbank und klicken Sie anschließend im Objekt-Explorer auf *Verbinden/Datenbankmodul*, um eine neue Verbindung aufzubauen.
10. Wenn Sie nun den Benutzernamen *MediaBaseContainedUser* und das Passwort *mbcu* eingeben und versuchen, sich mit der Datenbank zu verbinden, werden Sie eine Fehlermeldung erhalten, da der Benutzer für den SQL Server nicht bekannt ist.
11. Klicken Sie daher auf *Optionen* und geben Sie auf der Seite *Verbindungseigenschaften* unter *Verbindung mit Datenbank herstellen* den Namen der Datenbank (*MediaBaseContained*) ein.
12. Wenn Sie nun erneut auf *Verbinden* klicken, verbindet sich das Management Studio direkt mit der Datenbank *MediaBaseContained*, in der auch der Benutzer *MediaBaseContainedUser* bekannt ist.

Im Objekt-Explorer sind nun nur noch die Objekte verfügbar, die auch in der Datenbank enthalten sind.

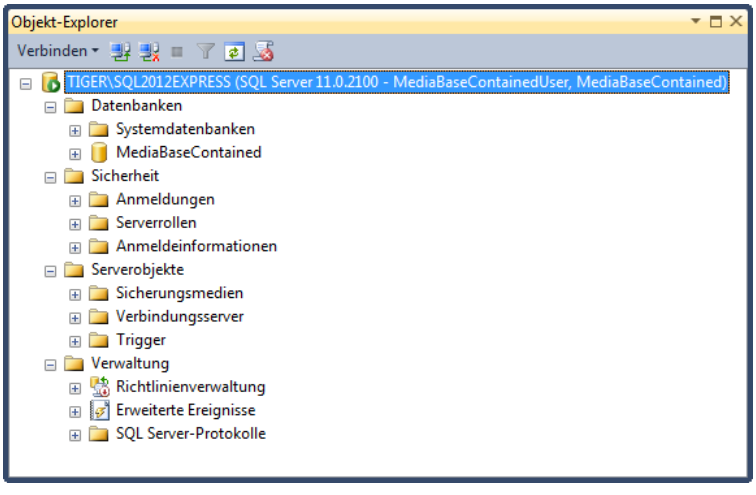

Abbildung 11.10: Objekt-Explorer bei Verbindung mit einer Contained Database

Bei vielen dieser Objekte – wie beispielsweise den Systemdatenbanken *master* und *tempdb* – handelt es sich im Prinzip um datenbankinterne Kopien der zugehörigen Server-Objekte.

Ansonsten können Sie nun mit der Datenbank arbeiten, wie mit fast jeder anderen Datenbank auch, allerdings mit einigen Einschränkungen, da einige Features, die sich aber größtenteils auf die kostenpflichtigen SQL Server-Editionen beziehen, nicht verwendbar sind. Dies sind unter anderem die folgenden:

- *Change Data Capture*
- *Change Tracking*
- *Nummerierte gespeicherte Prozeduren*
- *Replikation*

11.6 Verwendung von Schemas

Sie haben sich vielleicht schon gewundert, warum in vielen Angaben zu Datenbankobjekten wie Tabellen, Sichten etc. das Präfix *dbo* gefolgt von einem Punkt auftaucht. Dabei handelt es sich um das sogenannte Schema, zu dem das jeweilige Datenbankobjekt gehört. Dabei steht das Standardschema *dbo* für *Database Owner* und bezeichnet damit den Benutzer, der die Datenbank angelegt hat.

In früheren Versionen von SQL Server wurde zu jedem Datenbankobjekt ein Datenbankbenutzer als Besitzer definiert (nämlich der Benutzer, der das Objekt angelegt hat). Inzwischen wird ein Schema als Besitzer jedes Datenbankobjekts angegeben. Lediglich das Schema selbst hat einen Benutzer oder eine Rolle als Besitzer. Dazu wird über die Eigenschaften des Datenbankbenutzers für jeden Benutzer ein Standardschema definiert, das verwendet wird, wenn dieser Benutzer neue Datenbankobjekte erstellt. Auf diesem Weg können auch mehrere Benutzer (die dieses Schema als Standardschema verwenden) quasi als Besitzer des Objekts fungieren.

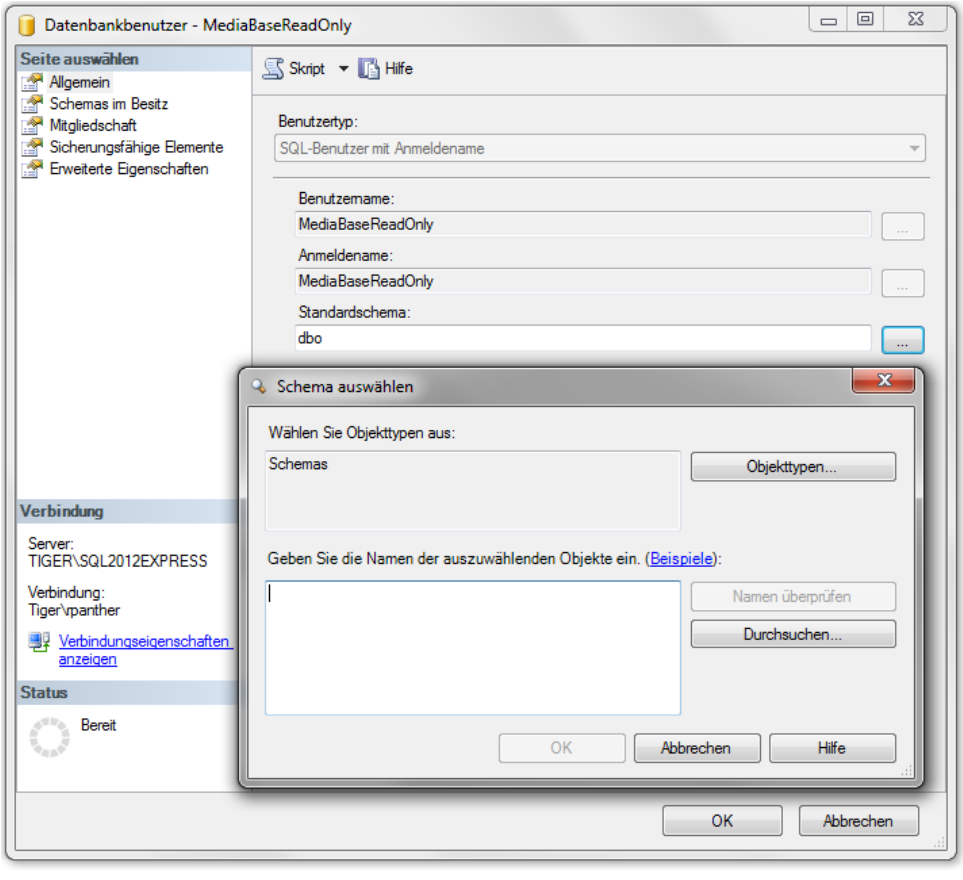

Abbildung 11.11: Auswahl eines Standardschemas für den Benutzer *MediaBaseReadOnly*

Dem ursprünglichen Ansatz, dass Benutzer und Schema identisch sind, ist es wohl zu verdanken, dass auch heute noch für jeden vordefinierten Benutzer und jede vordefinierte Rolle ein entsprechendes Schema existiert.

Robert Panther: Richtig einsteigen: Datenbanken entwickeln mit SQL Server 2012

Neben der Protokollierung des Objektbesitzers werden Schemas primär zu zwei Zwecken verwendet:

- Logische Gruppierung von zusammengehörenden Objekten innerhalb einer Datenbank zur besseren Übersicht
- Einfachere Verwaltung von Berechtigungen, indem diese nicht auf einzelne Datenbankobjekte, sondern auf ein gesamtes Schema erteilt werden können

Schemas erstellen

Die Verwaltung von Schemas ist sehr einfach, da hier so gut wie keine Einstellungen vorzunehmen sind. Probieren wir dies nun einmal aus und erstellen wir ein neues Schema, um Datenbankobjekte, die später für Auswertungen verwendet werden, von den eigentlichen Daten zu trennen.

1. Stellen Sie im SQL Server Management Studio unter Verwendung der Windows-Authentifizierung eine Verbindung zur lokalen Serverinstanz *SQL2012EXPRESS* her.
2. Klicken Sie im Objekt-Explorer den Zweig *Datenbanken/MediaBase/Sicherheit/Schemas* mit der rechten Maustaste an und wählen Sie den Befehl *Neues Schema*.
3. Hier reicht es erst einmal aus, den Schemanamen **Auswertungen** anzugeben. Für den Schemabesitzer wird automatisch der Benutzer verwendet, mit dem Sie gerade arbeiten. Schließen Sie nun das Dialogfeld mit der Schaltfläche *OK* und das neue Schema wird angelegt.

Dieselbe Aktion wäre auch mit folgendem SQL-Skript durchführbar:

```
CREATE SCHEMA Auswertungen
```

Das Löschen eines Schemas geht ebenso einfach (soll aber an dieser Stelle nicht ausgeführt werden):

```
DROP SCHEMA Auswertungen
```

Schemas verwenden

Wenn Sie nun ein Datenbankobjekt in einem SQL-Skript ansprechen wollen, das nicht Bestandteil des *dbo*-Schemas ist, müssen Sie vor dem eigentlichen Objektnamen (mit einem Punkt getrennt) den Namen des Schemas angeben.

Erstellen wir nun eine neue Tabelle im Schema *Auswertungen*:

1. Stellen Sie im SQL Server Management Studio unter Verwendung der Windows-Authentifizierung eine Verbindung zur lokalen Serverinstanz *SQL2012EXPRESS* her.
2. Öffnen Sie ein Abfragefenster auf der Datenbank *MediaBase* und geben Sie folgende SQL-Anweisung ein:

```
SELECT *
INTO Auswertungen.AktuelleBuecher
FROM dbo.Buch
WHERE Erscheinungsjahr > 2007
```

3. Wenn Sie nun im Objekt-Explorer die Liste der Tabellen in der Datenbank *MediaBase* aufklappen, werden Sie sehen, dass die neue Tabelle *Auswertungen.AktuelleBuecher* ganz oben in der Liste erscheint (eventuell muss die Ansicht vorher aktualisiert werden). Das liegt daran, dass diese alphabetisch sortiert ist. Wenn Sie also eine Datenbank mit sehr vielen Tabellen haben, erreichen Sie durch

die Verwendung von Schemas, dass Tabellen, die zu einem Schema gehören, direkt untereinander stehen (dasselbe gilt natürlich auch für Sichten etc.).

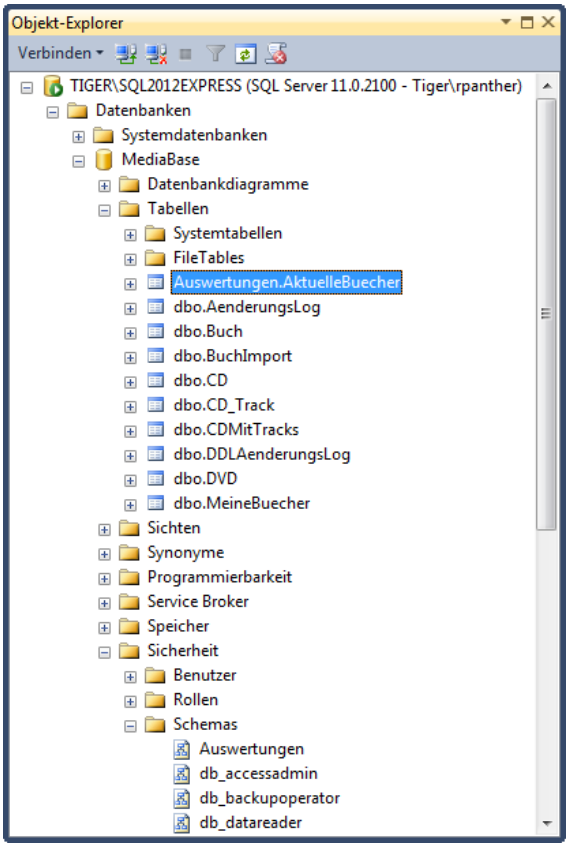

Abbildung 11.12: Das Schema *Auswertungen* und die Tabelle *AktuelleBuecher* im Objekt-Explorer

4. Um die Daten der neuen Tabelle abzufragen, muss natürlich ebenfalls der Schemaname mit angegeben werden:

```
SELECT *
FROM Auswertungen.AktuelleBuecher
```

Sie haben gerade ein Datenbankschema verwendet, um eine bessere Übersicht durch eine Gruppierung der Datenbankobjekte in Schemas zu erzielen.

Best Practices: Verwendung von Datenbankrechten und -rollen

Solange Sie ausschließlich mit dem *dbo*-Schema arbeiten, können Sie die Angabe des Schemas auch weglassen. Es wird jedoch generell empfohlen, Datenbankobjekte immer möglichst mit Angabe des Schemas zu spezifizieren, da dies einerseits Missverständnisse beim Lesen vermeidet und teilweise sogar für eine bessere Performance sorgt, da der SQL Server gleich weiß, in welchem Schema das entsprechende Objekt zu suchen ist (ansonsten wird das Objekt zuerst im Standardschema des angemeldeten Benutzers gesucht und anschließend im *dbo*-Schema, sofern die Berechtigungen dies zulassen).

Robert Panther: Richtig einsteigen: Datenbanken entwickeln mit SQL Server 2012

Berechtigungen für Schemas verwalten

Wie bereits weiter oben erwähnt, liegt ein weiterer Vorteil von Schemas darin, dass diese eine einfachere Vergabe von Berechtigungen ermöglichen. Um einer Anmeldung Zugriff auf alle Objekte eines Schemas zu erteilen, gibt es – neben der müßigen Erteilung der einzelnen Rechte für jedes Objekt, das Bestandteil des Schemas ist – zwei einfache Möglichkeiten:

- Übernahme des Besitzes des entsprechenden Schemas
- Erteilen von expliziten Berechtigungen für das gesamte Schema

Die erstgenannte Variante ist einfacher, bietet aber dafür geringere Flexibilität, weil mit der Besitzübernahme natürlich gleich alle Rechte auf das Schema erteilt sind. Dazu rufen Sie das *Eigenschaften*-Dialogfeld des entsprechenden Benutzers auf und aktivieren auf der Seite *Schemas im Besitz* die gewünschten Schemas in der Liste *Schemas im Besitz dieses Benutzers*.

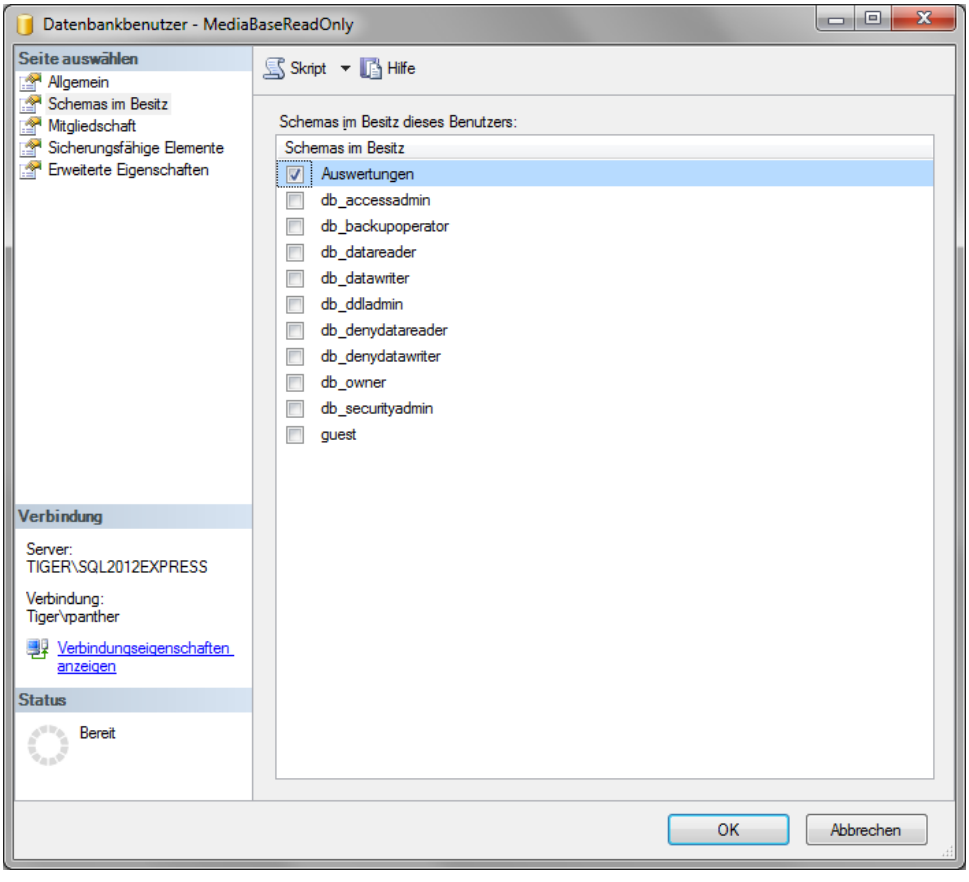

Abbildung 11.13: Besitzübernahme des Schemas *Auswertungen* durch den Benutzer *MediaBaseReadOnly*

Alternativ können Sie auch die folgende SQL-Anweisung verwenden, um beispielsweise den Benutzer *MediaBaseReadOnly* den Besitz des Schemas *Auswertungen* übernehmen zu lassen:

```
ALTER AUTHORIZATION ON SCHEMA::[Auswertungen] TO [MediaBaseReadOnly]
```


Robert Panther: Richtig einsteigen: Datenbanken entwickeln mit SQL Server 2012
Copyright 2012 O'Reilly Verlag GmbH & Co. KG, ISBN 978-3-86645-219-0

Ein weiterer Nachteil dieser Variante liegt darin, dass nur ein Benutzer oder eine Rolle den Besitz eines Schemas halten kann. Daher nutzen wir nun die zweite Variante (Erteilen von expliziten Berechtigungen), um dem Benutzer *MediaBaseAudioOnly* (der ja bisher nur Zugriff auf die Tabellen *dbo.CD* und *dbo.CD_Track* hat), Lesezugriff für das gesamte Schema *Auswertungen* zu geben:

1. Stellen Sie im SQL Server Management Studio unter Verwendung der Windows-Authentifizierung eine Verbindung zur lokalen Serverinstanz *SQL2012EXPRESS* her.
2. Öffnen Sie im Objekt-Explorer den Zweig *Datenbanken/MediaBase/Sicherheit/Schemas*.
3. Klicken Sie mit der rechten Maustaste auf das Schema *Auswertungen* und wählen Sie anschließend *Eigenschaften* aus. Auf der Seite *Allgemein* können Sie ebenfalls den Besitz des Schemas an einen Benutzer (oder eine Rolle) übertragen.
4. Öffnen Sie die Seite *Berechtigungen*.
5. Über die Schaltfläche *Suchen* können Sie *Benutzer* oder *Rollen* auswählen, für die Sie anschließend Rechte erteilen. Suchen Sie hier nach dem Benutzer *MediaBaseAudioOnly*.
6. Geben Sie dem gewählten Benutzer nun Leserechte auf das gesamte Schema, indem Sie das *Erteilen*-Kästchen für die Berechtigung *Auswählen* anklicken.

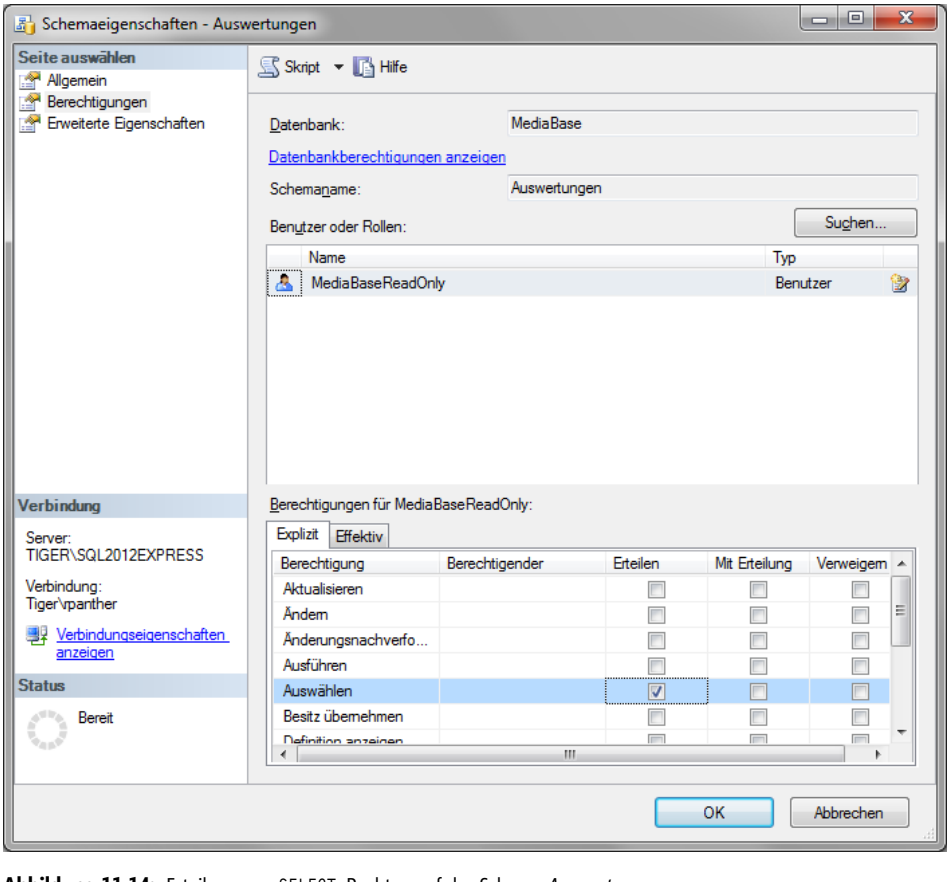

Abbildung 11.14: Erteilung von SELECT-Rechten auf das Schema *Auswertungen*

7. Schließen Sie nun das *Schemaeigenschaften*-Dialogfeld durch einen Klick auf *OK*.
8. Wenn Sie sich anschließend über SQL Server-Authentifizierung mit dem Benutzer *MediaBase-AudioOnly* anmelden, werden Sie sehen, dass dieser nun auch Zugriff auf die Tabelle *Auswertungen.AktuelleBuecher* hat, da diese Bestandteil des *Auswertungen*-Schemas ist.

Alternativ hätten Sie diese Aktion auch mit der folgenden SQL-Anweisung durchführen können:

```
GRANT SELECT ON SCHEMA::[Auswertungen] TO [MediaBaseAudioOnly]
```

Wenn Sie zusätzlich die Option *Mit Erteilung* aktivieren möchten, wäre die obige Anweisung um den Zusatz `WITH GRANT OPTION` zu erweitern:

```
GRANT SELECT ON SCHEMA::[Auswertungen] TO [MediaBaseAudioOnly] WITH GRANT OPTION
```

Die entsprechenden Anweisungen zum Entziehen oder Verweigern einer Berechtigung sind `REVOKE` (Zurücknehmen einer vorhandenen Berechtigung) und `DENY` (Verweigern einer Berechtigung, die evtl. durch eine Rollen- oder Gruppenzugehörigkeit gegeben war).

11.7 Übungen zu diesem Kapitel

In diesem Abschnitt finden Sie einige Übungen zu diesem Kapitel. Die richtigen Antworten, Lösungen und den Programmcode finden Sie wie immer auf der Website *www.richtig-einsteigen.de.*

Übung 11.1

Erstellen Sie eine neue SQL-Anmeldung *DBBackupGuy* mit dem Kennwort *dbbg* und ordnen Sie diesen der Serverrolle *diskadmin* zu.

Übung 11.2

Erstellen Sie für die SQL-Anmeldung *DBBackupGuy* einen Datenbankbenutzer für die Datenbanken *MediaBase* und *MediaBaseContained* und erteilen Sie diesem für beide Datenbanken eine Mitgliedschaft in der Datenbankrolle *db_backupoperator*.

Übung 11.3

Erstellen Sie eine SQL Server-Anmeldung *MediaBaseBooksOnly* und richten Sie dafür auch einen entsprechenden Datenbankbenutzer ein. Erteilen Sie diesem Datenbankbenutzer anschließend Vollzugriff auf die Tabelle *dbo.Buch*.

Robert Panther: Richtig einsteigen: Datenbanken entwickeln mit SQL Server 2012

11.8 Zusammenfassung

In diesem Kapitel haben Sie die wichtigsten Möglichkeiten kennengelernt, um Zugriffe auf Datenbankserver, Datenbanken und Datenbankobjekte zu regeln. Je nach Art der verwendeten Authentifizierung erstellen Sie Ihre Benutzer und Gruppen im Windows-Betriebssystem (Windows-Authentifizierung) oder Anmeldungen im SQL Server (SQL Server-Authentifizierung). Diese Anmeldungen können über vordefinierte Serverrollen serverweite Berechtigungen erhalten.

Datenbankseitig werden den Anmeldungen Datenbankbenutzer zugeordnet, die explizit Rechte erhalten können, aber ebenso Rollen zugeordnet werden können, den sogenannten Datenbankrollen. Die neuen Contained Databases erlauben einen direkten Zugriff von Datenbankbenutzern, ohne dafür Anmeldungen anlegen zu müssen.

Schemas ermöglichen das Gruppieren von Objekten wie Tabellen, Sichten etc. innerhalb einer Datenbank. Darüber wird einerseits eine übersichtlichere Anordnung der Objekte im Objekt-Explorer erreicht. Dazu erleichtern Schemas die effektive Verwaltung von Rechten, da diese nun auf das ganze Schema erteilt werden können und nicht mehr für jedes Datenbankobjekt einzeln vergeben werden müssen.

Robert Panther: Richtig einsteigen: Datenbanken entwickeln mit SQL Server 2012

Robert Panther: Richtig einsteigen: Datenbanken entwickeln mit SQL Server 2012
 ISBN 978-3-86645-219-0