Datenbankimplementierung

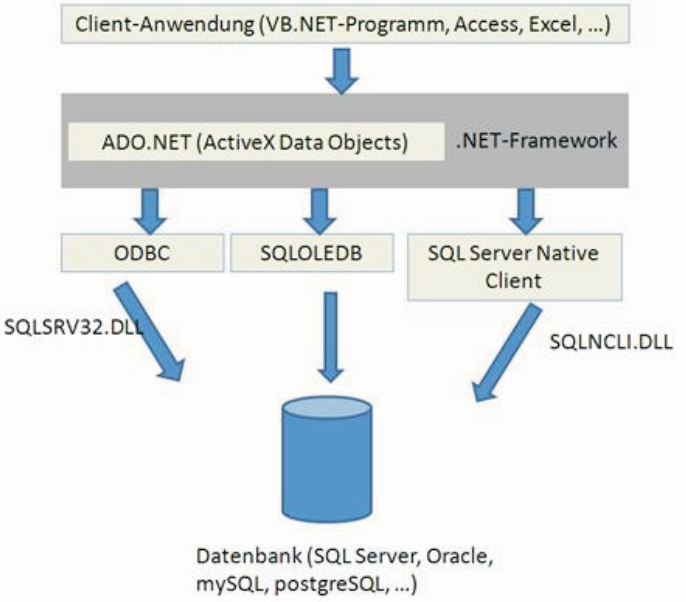
Sofern Ausführungen auf SQL Server 2005 Bezug nehmen, gelten sie gleichermaßen auch für SQL Server 2008.

Christian Zahler

6 Client-Programmierung von MS SQL Server 2005

6.1 Grundlagen

Um eine (Server-)Datenbank programmiertechnisch anzusprechen, ist es nötig, eine Schnittstelle zu definieren. Grundsätzlich gilt: Es ist nicht möglich, die Datenbank direkt anzusprechen.



6.2 MS Access 2007 als Client mit Hilfe einer ODBC-Systemschnittstelle

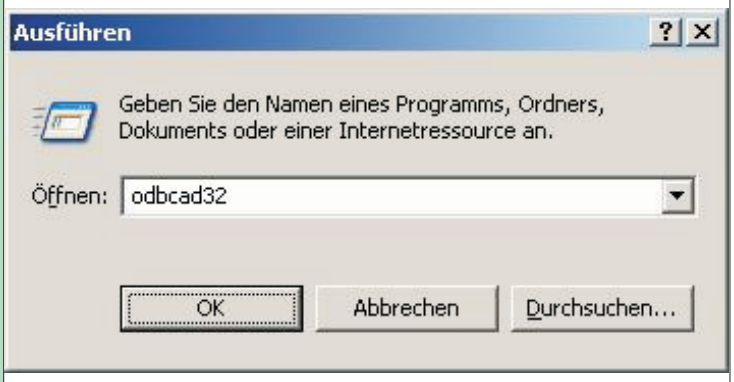
Ein relativ einfaches Verfahren zur Erstellung eines SQL Server-Clients bietet MS Access (ab Version 2003). Der eigentliche Datenbankzugriff wird von einer ODBC-Schnittstelle durchgeführt.

ODBC *(Open DataBase Connectivity)* stellt über spezielle Treiber (ODBC-Treiber) eine Programmierschnittstelle bereit, die standardmäßig (von Access oder durch VB-Programmierung) angesprochen werden kann.

Schritt 1: Einrichten einer ODBC-Schnittstelle

mit dem ODBC-Datenquellen-Administrator

Start – Ausführen – odbcad32



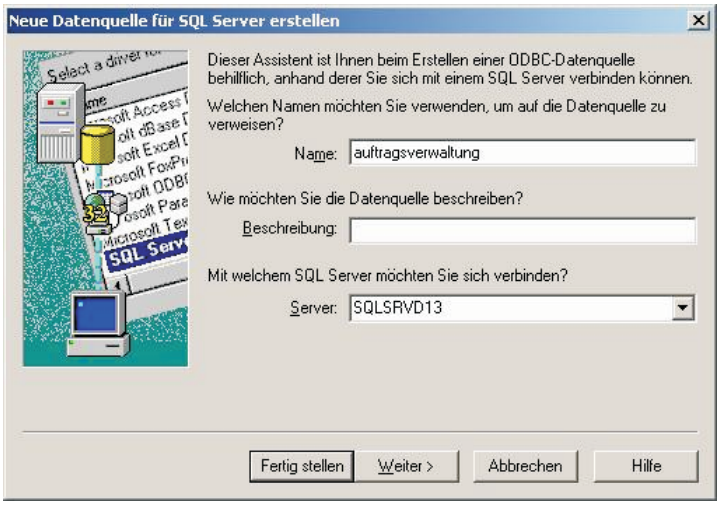
Der ODBC-Datenquellen-Administrator erlaubt die Erstellen von drei Schnittstellentypen, die auch als DSN (*data source name*, Datenquellenname) bezeichnet werden:

- **Benutzer-DSN**: Diese Schnittstelle kann nur von dem Benutzer verwendet werden, der sie erstellt hat.
- **System-DSN**: Diese Schnittstelle steht allen Benutzern und dem lokalen Systemkonto zur Verfügung.
- **Datei-DSN**: Die Schnittstellenparameter werden in einer *.dsn-Datei gespeichert und können so auf andere PCs transportiert werden.

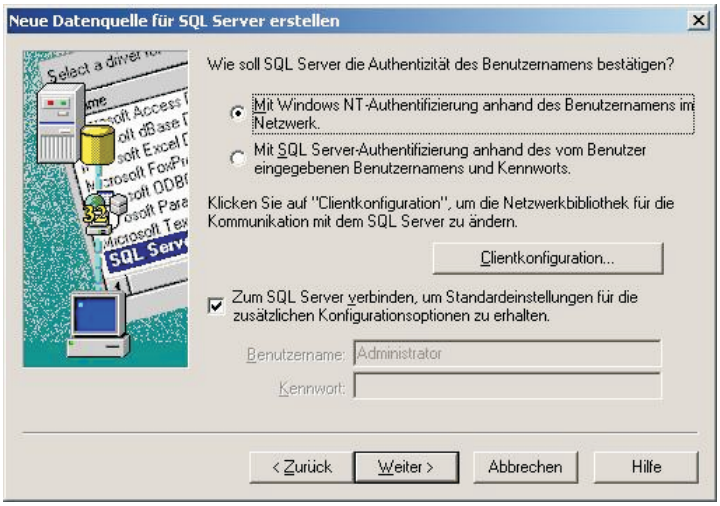
(a) Verwenden des ODBC-Treibers für SQL Server (SQLSRV32.DLL; verwendbar für Versionen ab SQL Server 7.0)

Auf *„Hinzufügen"* klicken, dann den ODBC-Treiber für SQL Server (`SQLSRV32.DLL`) auswählen:

Auf *„Fertigstellen"* klicken.

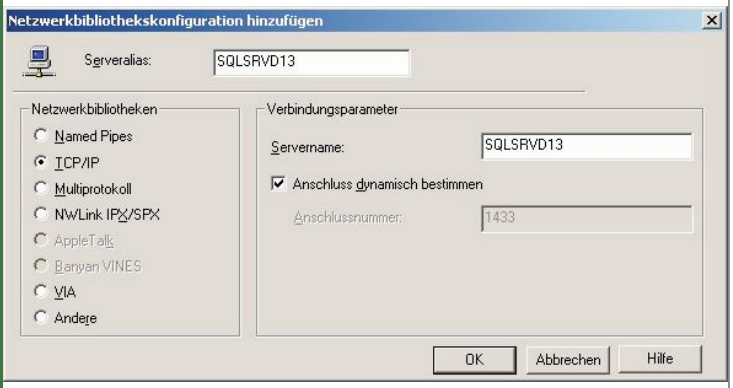


Der Name ist als DSN-Name zu verstehen, der zukünftig für das Ansprechen der Datenbank verwendet wird.

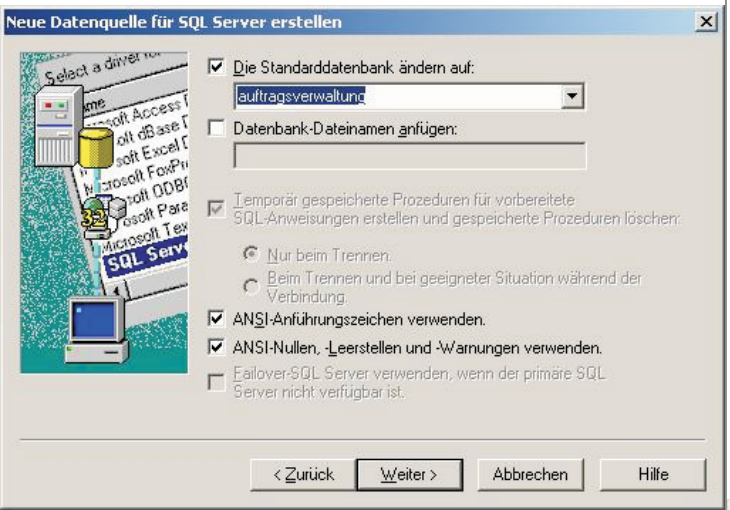


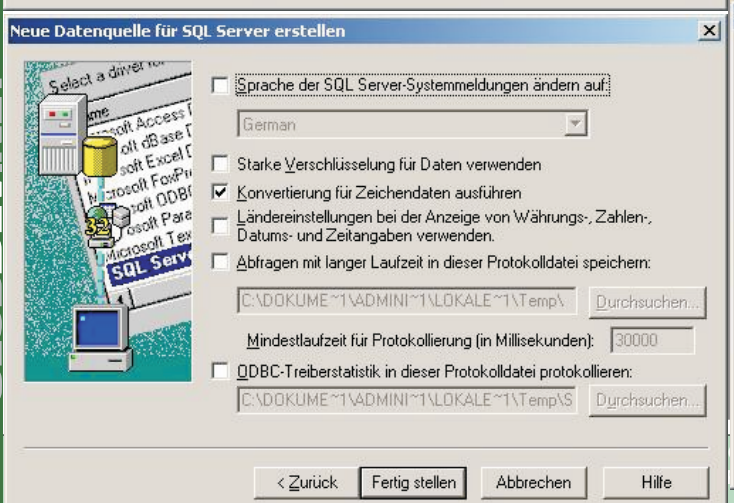
Hier wählen Sie bitte aus, ob Windows- oder SQL Server-Authentifizierung verwendet werden soll.

Unter *„Clientkonfiguration"* überprüfen Sie, ob TCP/IP als verwendete Netzwerkbibliothek eingestellt ist:

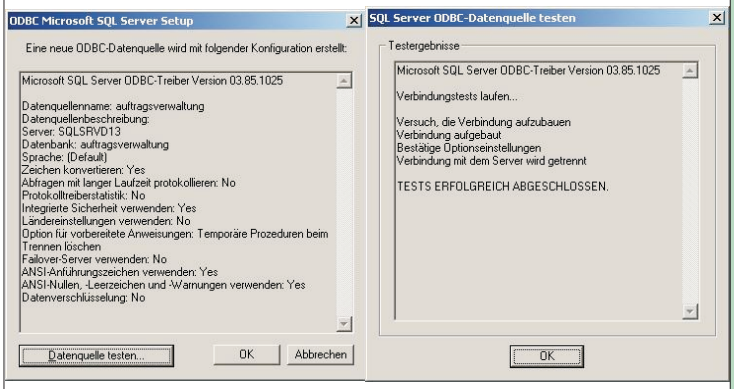


Wählen Sie anschließend die zu verwendende Datenbank:



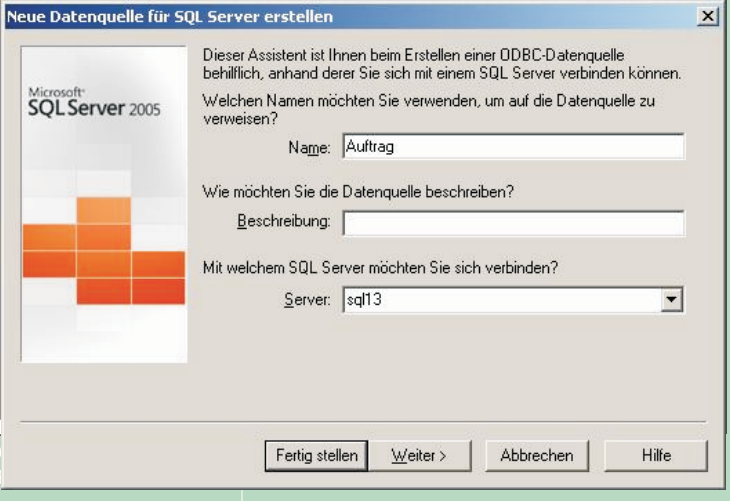


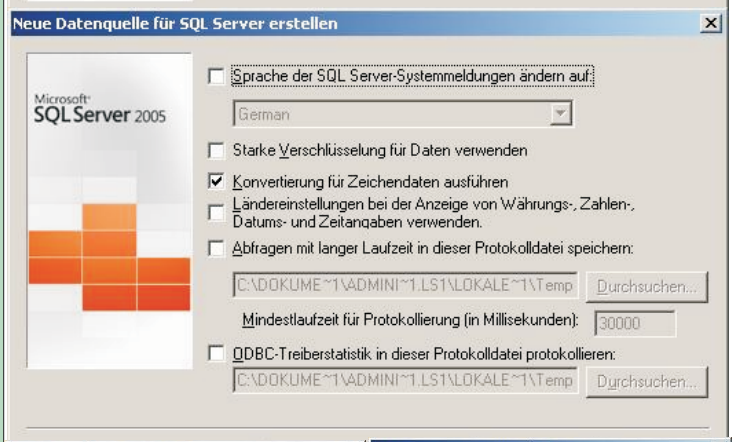
Mit *„Datenquelle testen…"* können Sie den Zugriff auf die Server-Datenquelle überprüfen:

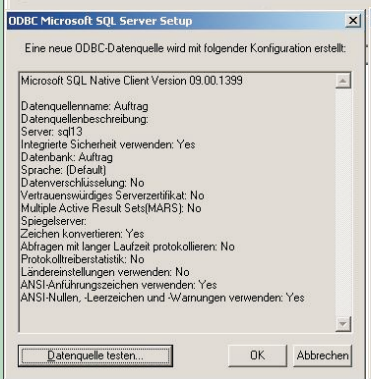


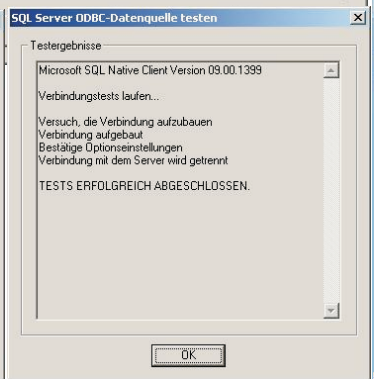
Wenn Sie die ODBC-Schnittstelle erfolgreich erstellt haben, sollte das ungefähr so aussehen:

(b) Verwenden des ODBC-Treibers für SQL Native Client (SQLNCLI.DLL; verwendbar ab SQL Server 2005)



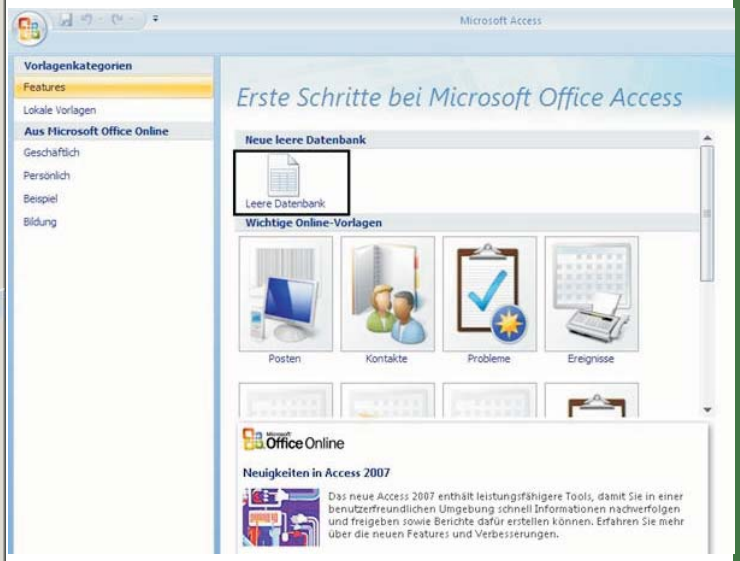


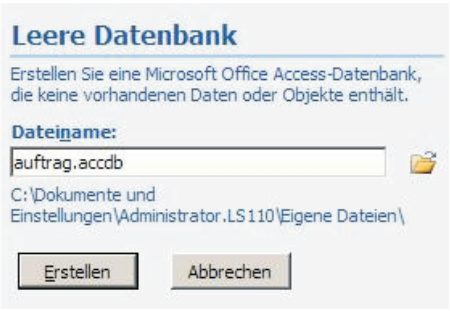




Schritt 2: Erstellen verknüpfter Tabellen in Access

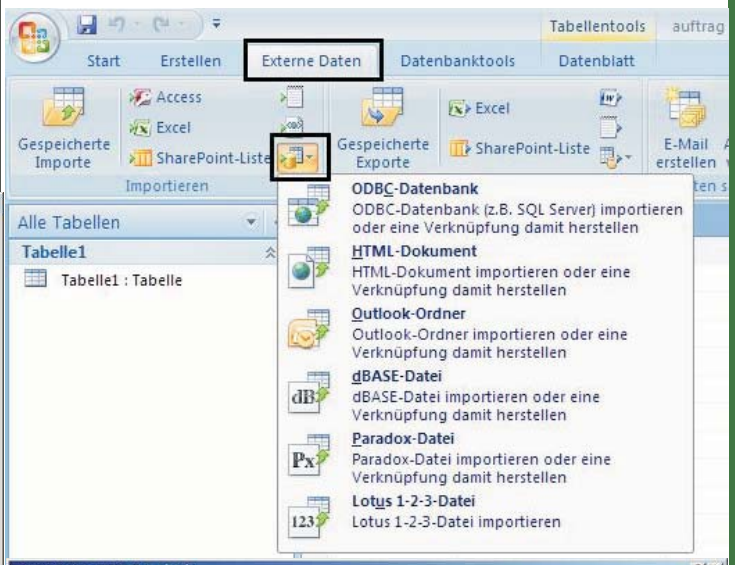
Legen Sie zunächst eine neue Access-Datenbank an.

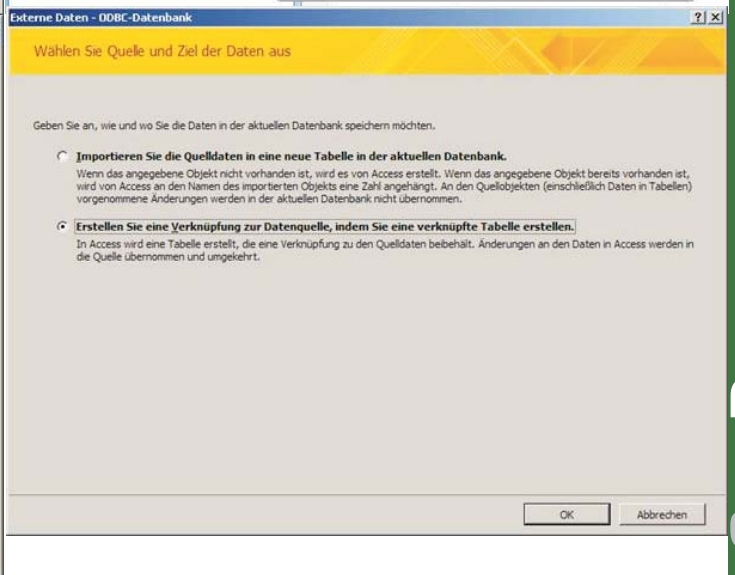




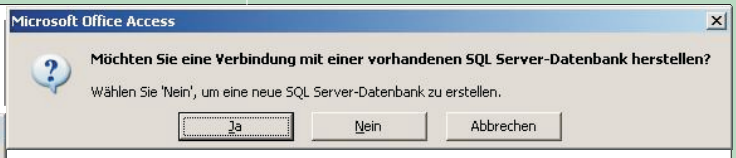
Nun wählen Sie den wählen Sie im Ribbon *„Externe Daten“* das Symbol für *„Weitere Datenbankformate importieren“* aus und wählen *„ODBC-Datenbank“*:

Wählen Sie im erscheinenden Dialog den Punkt *„Erstellen Sie eine Verknüpfung zur Datenquelle, indem Sie eine verknüpfte Tabelle erstellen“*:

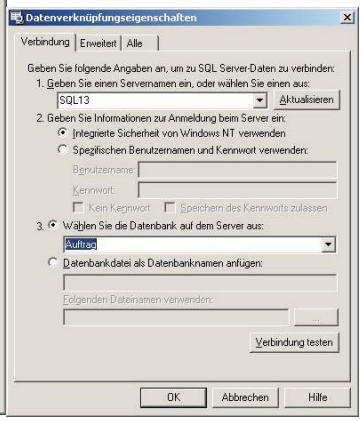




Im Menüpunkt *„Datenquelle auswählen"* aktivieren Sie die Karteikarte *„Computerdatenquelle"* und wählen die vorher konfigurierte ODBC-Schnittstelle aus:

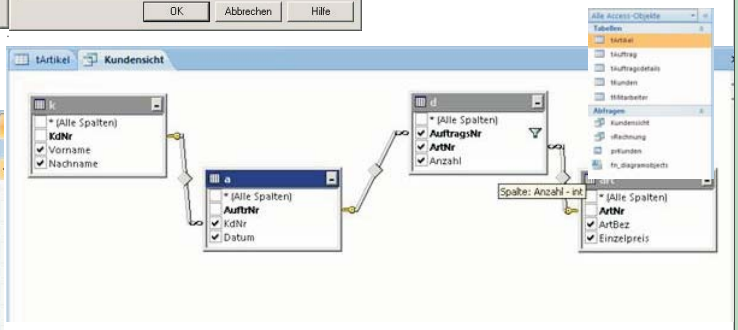


Wählen Sie in diesem Dialog den SQL-Server, die Art der Authentifizierung und die Datenbank aus.

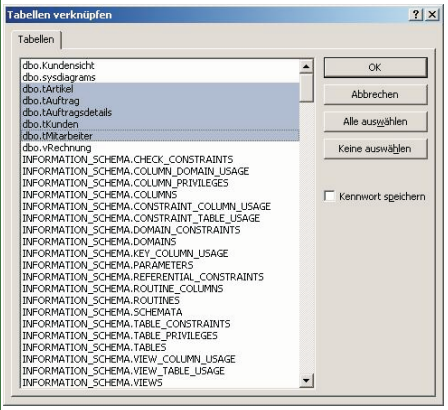


Die Verbindung kann auch getestet werden:

Man sieht, dass hier nicht nur Tabellenzugriffe übernommen wurden, sondern auch Sichten und gespeicherte Prozeduren (unter *„Abfragen"*).



Wählen Sie dann die zu verknüpfenden Tabellen aus:



Ergebnis:

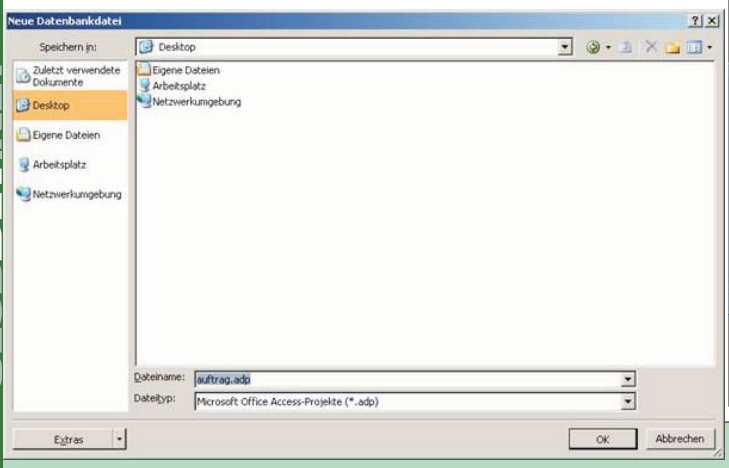
Auf Basis dieser Verknüpfungen können nun Abfragen, Formulare und Berichte erstellt werden.

6.3 MS Access-Datenbankprojekte (ohne ODBC-Schnittstelle)

Eine zweite Möglichkeit besteht in der Verwendung einer Access-internen Zugriffsmöglichkeit, die aber erst seit Access 2003 fehlerfrei und stabil arbeitet.

Datenbankprojekte werden als `*.ADP` *(Access Data Project)* gespeichert.

Speichern Sie das Projekt:

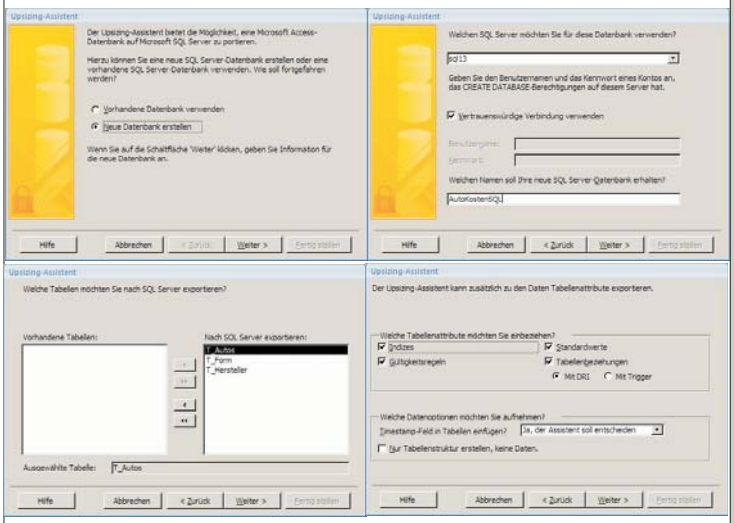


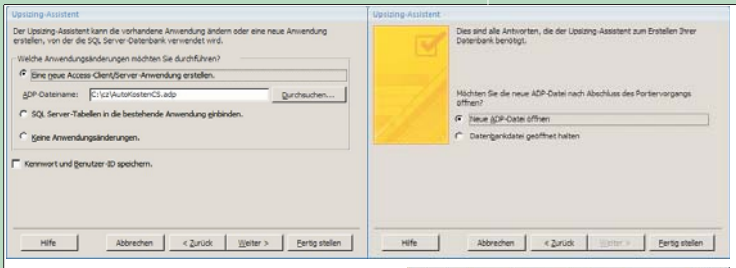
7 Upgrade Access auf SQL Server 2005

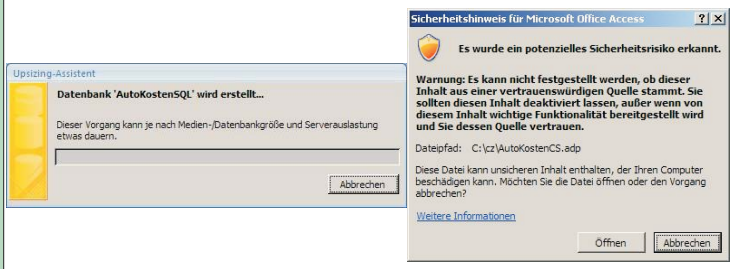
7.1 Upgrade mit dem Access 2007-Upsizing-Assistenten

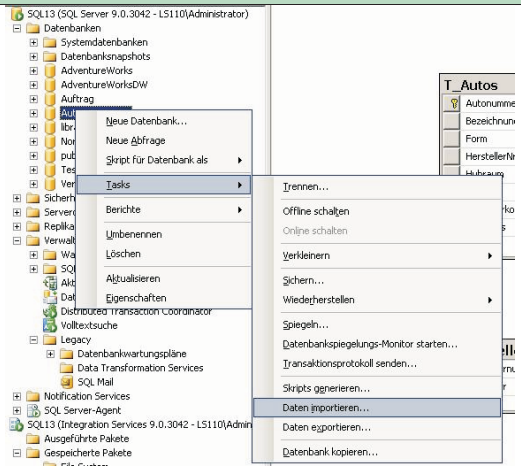
Öffnen Sie die Access-Datenbank und wählen Sie aus dem Menüband *"Datenbanktools"* das Symbol *"SQL Server"*:

Es startet der *"Upsizing-Assistent"*, mit dem Sie sowohl eine neue SQL Server-Datenbank erstellen können, als auch eine vorhandene SQL Server-Datenbank mit Daten befüllen können.



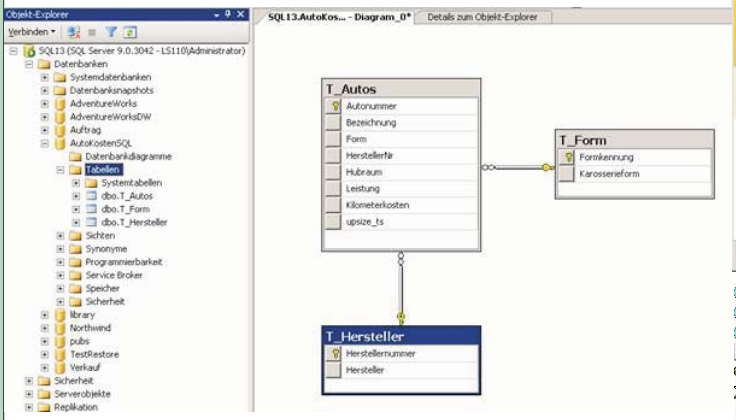


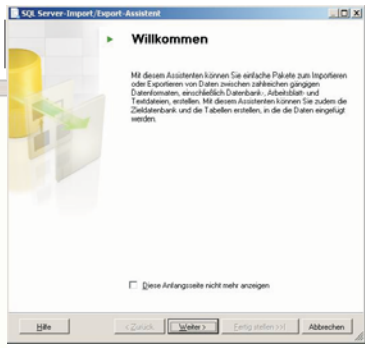


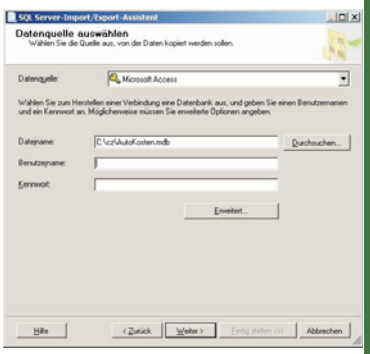


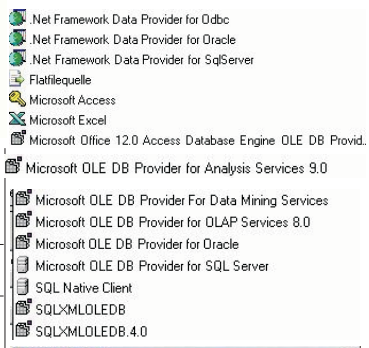
Ablauf des Assistenten:

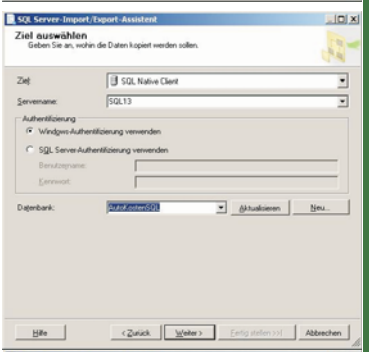
Ergebnis







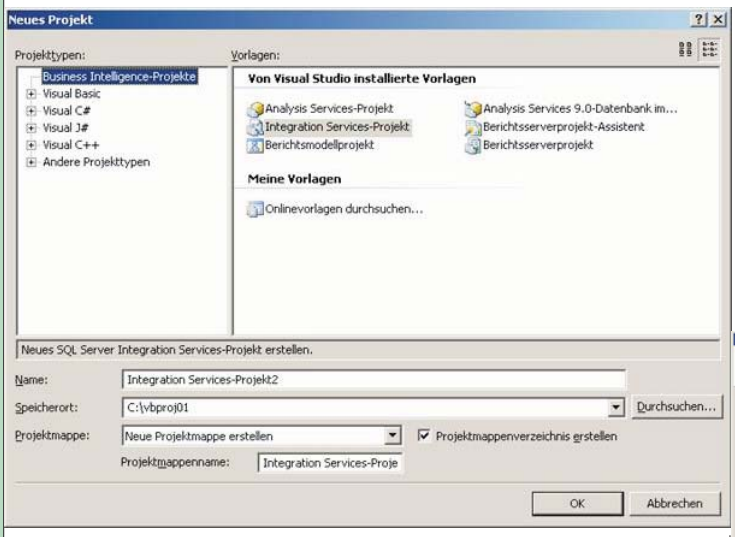


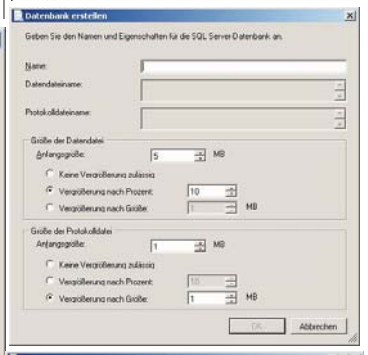


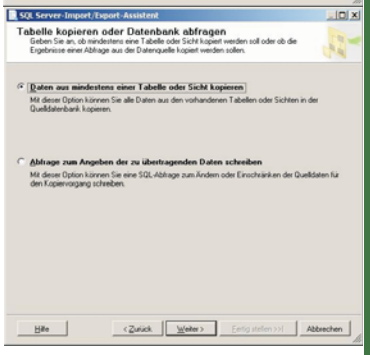
Hinweis: Abfragen werden nicht übernommen; weder werden Sie in Views oder Procedures am SQL Server konvertiert, noch im ADP-Projekt gespeichert.

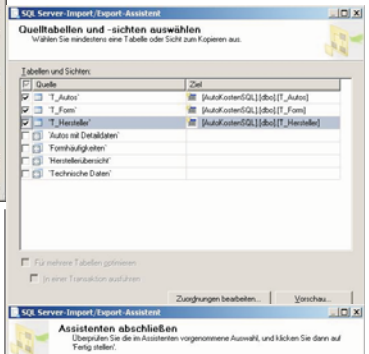
7.2 Datenimport aus einer Access-Datenbank mit dem SQL Server Integration Services (SSIS)-Import/Export-Assistent:

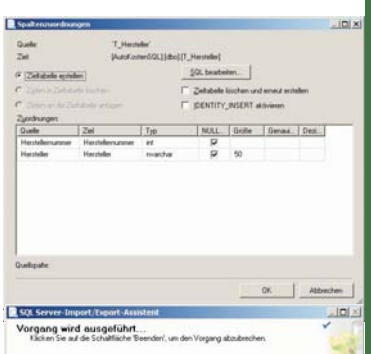
Variante 1: Starten Sie das SQL Server Business Intelligence Development-Studio und erstellen Sie ein neues Integration Services-Projekt

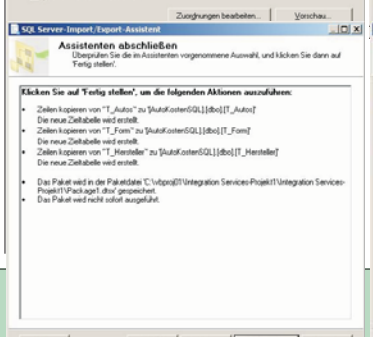


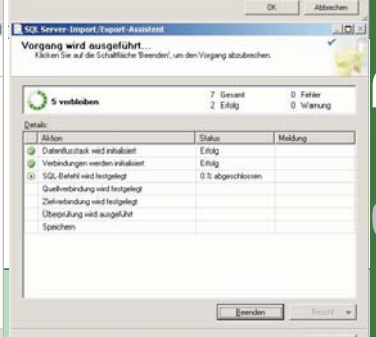


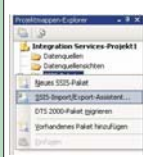










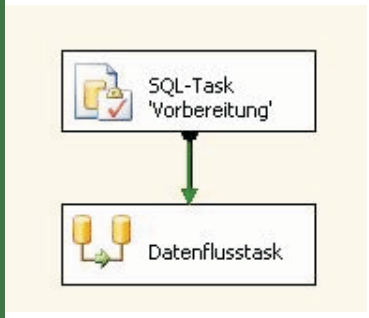


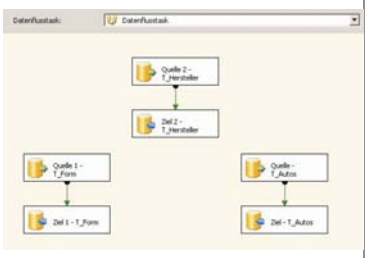
Im Projektmappen-Explorer klicken Sie mit der rechten Maustaste auf *"SSIS-Pakete"* und wählen aus dem Kontextmenü *[SSIS-Import/Export-Assistent]*.

Variante 2: Führen Sie in einem Eingabeaufforderungsfenster `DTSWizard.exe` aus. Diese Datei ist im Verzeichnis `C:\Programme\Microsoft SQL Server\90\DTS\Binn` gespeichert.

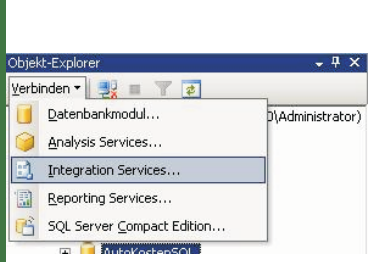
Variante 3: Im SQL Server Management Studio Kontextmenü einer Datenbank auswählen, *[Tasks] – [Daten importieren]*

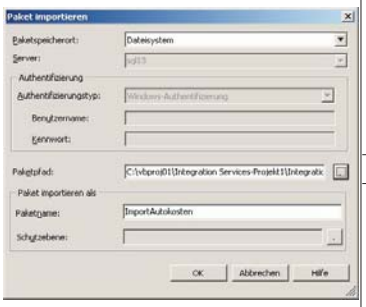
Ein neues SSIS-Paket wird erzeugt. Ablaufsteuerung:

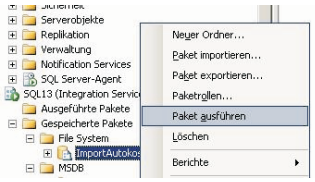


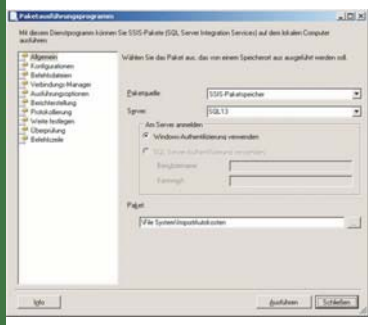


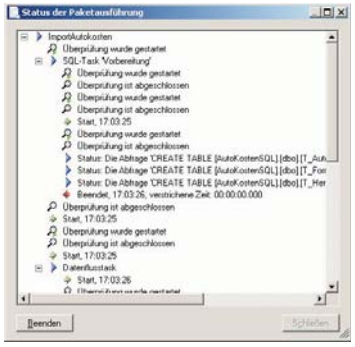
Management Studio:











Beachten Sie: Es sind keine Fremdschlüsseleinschränkungen vorhanden!

8 ActiveX Data Objekts (ADO) und ADO.NET

Die *ActiveX Data Objects* wurden vor einigen Jahren als eine Technologie eingeführt, die den Datenzugriff nicht nur über ein lokales Netzwerk, sondern auch über das Internet ermöglicht. ADO löste damit sowohl RDO *(Remote Data Objects)* als auch DAO *(Data Access Objects)* ab, das ursprünglich für die Jet-Datenbankengine entwickelt wurde. Nun taucht zusätzlich ADO.NET auf.

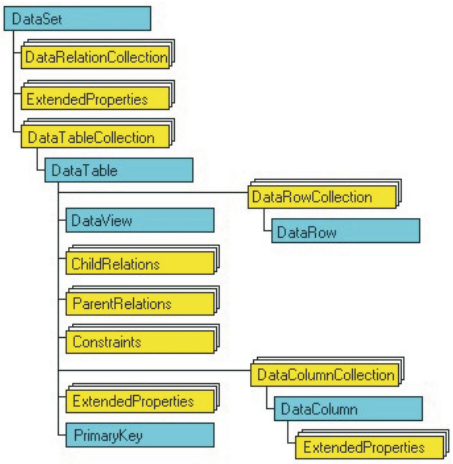
8.1 Die wesentlichen Unterschiede zwischen ADO und ADO.NET

Es gibt eine Reihe von Unterschieden zwischen ADO und ADO.NET, die hier zunächst nur stichpunktartig aufgeführt werden sollen:

- ADO arbeitet mit verbundenen Daten. Das heißt, dass wenn Sie Daten anzeigen oder aktualisieren, Sie eine Echtzeitverbindung zu ihnen haben.
- ADO.NET benutzt die Daten ohne Verbindung. Wenn Sie auf Daten zugreifen, legt ADO.NET eine Kopie der Daten mit Hilfe von XML an und hält nur während der Zeit die Verbindung zur Datenquelle aufrecht, in der die Daten abgefragt oder aktualisiert werden.
- ADO hat ein Hauptobjekt, das Recordset-Objekt, das dazu verwendet wird, auf Daten zuzugreifen. Es erlaubt eine einzige Ansicht Ihrer Daten, wobei diese natürlich auch relational sein kann. Mit ADO.NET stehen Ihnen viele Objekte zur Verfügung, die es Ihnen erlauben, auf Ihre Daten in unterschiedlichster Form zuzugreifen, darunter auch das DataSet-Objekt, das das relationale Modell Ihrer Datenbank repräsentiert.
- Mit ADO sind nur clientseitige Cursor möglich, ADO.NET hingegen lässt Ihnen die Wahl, entweder clientseitige oder serverseitige Cursor zu benutzen. In ADO.NET stehen für die Handhabung der Cursor spezielle Klassen zur Verfügung, so dass Sie sich um viele Details nicht kümmern müssen.
- Während ADO Ihnen nur erlaubt, Daten im XML-Format darzustellen, können Sie mit ADO.NET Ihre Daten auch mit Hilfe von XML manipulieren. Das ist nützlich, wenn Sie mit anderen Geschäftsanwendungen arbeiten oder Firewalls überwinden müssen, die Daten im HTML- und im XML-Format passieren lassen.

Diese Unterschiede sorgen dafür, dass ADO.NET im Bereich der Webanwendungen klare Vorteile beim Datenzugriff bietet. Aber auch bei den Desktopanwendungen kann der Einsatz von ADO.NET sinnvoll sein. Das werden Sie an den folgenden Beispielen sehen.

8.2 Objekte in ADO .NET



Wie bereits erwähnt, ist das meistgenutzte Objekt in ADO .NET das DataSet-Objekt. Sie sehen es mit seinen Eigenschaften, Methoden und Unterobjekten im Bild links. Weitere Objekte, die Ihnen bei der Programmierung mit ADO.NET immer wieder begegnen werden, sind in Tabelle 1 beschrieben.

Tabelle 1: ADO .NET-Objekte für die Manipulation von Daten

`DataSet`	Das Objekt wird in Verbindung mit anderen Datensteuerelementen benutzt, um Ergebnisse von Commands und DataAdapters zu speichern. Im Gegensatz zum `Recordset` von ADO und DAO ist das `DataSet` in der Lage, Daten hierarchisch darzustellen. Mit Hilfe der Eigenschaften und Auflistungen des `DataSet`-Objekts können Sie alles von der Beziehung bis hin zur einzelnen Zeile oder Spalte erreichen.
`DataTable`	`DataTable` ist eines der Objekte des `DataSets`, das es Ihnen erlaubt, einzelne Datentabellen zu bearbeiten. Es ähnelt dem `Recordset` von ADO.
`DataView`	Mit diesem Objekt können Sie Ihre Daten filtern und sortieren, um verschiedene Ansichten der Daten zu haben. Jedes `DataTable`-Objekt hat einen `DefaultView`, der den Ausgangs-`DataView` darstellt. Dieser kann modifiziert und gespeichert werden.
`DataRow`	Dieses Objekt ermöglicht es Ihnen, einzelne Zeilen Ihrer `DataTable` zu modifizieren. Sie können sich das Objekt wie einen Datencache vorstellen, den Sie bearbeiten können, das heißt, Sie können Daten ändern, hinzufügen und löschen. Die Änderungen schreiben Sie dann zurück in das Recordset, indem Sie SQL-Befehle auf dem Server ausführen.
`DataColumn`	Das Objekt repräsentiert Spalten. Das Interessante daran ist, dass Sie sowohl Schemainformationen als auch Daten erhalten können. Möchten Sie zum Beispiel ein Listenfeld mit Feldnamen füllen, können Sie die `DataColumn`-Collection einer `DataRow` durchlaufen und die Feldnamen auslesen.
`PrimaryKey`	Dieses Objekt erlaubt es Ihnen, einen Primärschlüssel für eine `DataTable` anzugeben, der zum Beispiel bei der Verwendung der `Find`-Methode wichtig ist.

Den Abschluss dieser Einführung finden Sie im Anhang zu diesem Heft, der als PDF-Datei bei der Webversion gespeichert ist.