FoxRockX

Juli 2008
Sonderheft 1 Deutsch

Kostenloses Sonderheft!

Sehr geehrte FoxPro Entwickler,

hier ist Ihr kostenloses deutsches Probeheft unseres neuen ausschließlich an Visual FoxPro orientierten und eigentlich nur englischsprachigen Magazins FoxRockX! Dies ist keine reguläre Ausgabe sondern nur ein Beispiel, wie unsere Zeitschrift FoxRockX aussieht. Wir sind uns sicher, dass es Ihnen gefallen wird!

Wie Sie bestimmt wissen, lebt die Gemeinde der FoxPro-Entwickler weiterhin verhältnismäßig unbeschwert vor sich hin, entwickelt bestehende Anwendungen weiter und startet immer wieder auch neue Projekte. Die weltweit rund einhunderttausend aktiven Entwickler mit Ihrer unglaublich hohen Zahl von Anwendungen und noch viel mehr Anwendern sind dabei natürlich auch für andere Firmen als Kunden interessant. Bloß weil Microsoft niemals aktives Marketing für die Entwicklungsumgebung samt integrierter Datenbank und Berichtsgenerator betrieben hat und VFP „Sedna“ zwar bis 2015 supporten aber keine neue Version mehr publizieren möchte, heißt das ja noch lange nicht, dass nicht andere Firmen ihre Produkte erweitern und mit Visual FoxPro kompatibel machen könnten, um daraus interessante Angebote für Visual FoxPro-Entwickler zu stricken.

Eines dieser Angebote ist der Advantage Database Servers von Sybase iAnywhere (vormals Extended Systems). Mit ADS modernisieren Sie gewachsene Applikationen durch einfache Migration auf Client/Server – und das bei weiterhin direktem Zugriff auf die DBF Dateien. Visual FoxPro Entwickler können dabei mit geringem Aufwand die Vorteile der Client/Server Architektur nutzen. Hierbei kann man mit DBF-Tabellen auf dem Server arbeiten, ohne Endanwendern über das Dateisystem eine direkte Zugriffsmöglichkeit auf die Tabellen zu gewähren und ohne Beschädigung von Tabellen oder Indizes bei Netzwerkproblemen oder lokalen Reboots. Ein weiterer Vorteil ist, dass der DBF-Dateizugriff von Endanwendern wieder über ODBC erfolgen kann.

Doug Hennig hat sich die Vor- und Nachteile dieses Datenbanksystem aus einer VFP-Perspektive gründlich angeschaut. Diese von Sybase gesponsorte Sonderausgabe von FoxRockX enthält alle wichtigen Details. Wenn Sie den Advantage Database Server selbst testen möchten, können Sie mit dem beigefügten Voucher ihre persönliche 2-User Entwicklerversion kostenlos anfordern! Alternativ finden Sie weitere Informationen auf folgender Seite: www.Sybase.com/vfp Und wenn Ihnen diese Probeausgabe gefällt:...

Werden doch auch Sie heute noch Leser von FoxRockX zu unseren fairen Abonnementspreisen incl. Zugriff auf die weltweit wohl größte Artikelsammlung rund um Visual FoxPro. Viele bekannte Autoren publizieren regelmäßig hochwertige Fachartikel zu und rund um Microsoft Visual FoxPro in FoxRockX!

FoxRockX erscheint zweimonatlich mit 24 Seiten DIN A4 (gleicher Jahresumfang wie FoxTalk) und das Abonnement beinhaltet den Zugriff auf das komplette Online-Archiv von FoxTalk sowie natürlich von FoxRockX. Darüber hinaus publizieren wir Sonderhefte wie diese Ausgabe. Das Jahresabonnement kostet online EUR 75,- und als gedruckte Ausgabe EUR 109,-. Sofern Sie ein noch laufendes Abonnement von FoxTalk haben, führen wir dieses als Online-Abonnement fort. Für den Differenzpreis von nur EUR 34,- pro Jahr können Sie das Online-Abonnement aber in ein Abonnement der gedruckten Ausgabe aufstufen.

Weitere Informationen finden Sie auf unserer kleinen Homepage unter http://www.foxrockx.com. Zwecks Bestellung besuchen Sie bitte unsere Onlineshop unter http://shop.dfpug.com (Europa/Asien) oder http://www.hentzenwerke.com (USA/Kanada). Online Artikel, Artikelarchive und Begleitmaterial finden Sie unter dem “FoxRockX” Reiter in unserem Dokumentenportal unter http://portal.dfpug.de. Username und Passwort für den Zugriff schicken wir Ihnen mit der Abonnementbestätigung per eMail zu. Das dFPUG-Team wünscht Ihnen nunmehr viel Spaß beim Lesen!

Der Advantage Database Server für Visual FoxPro Entwickler

Doug Hennig

Der Advantage Database Server ist eine vollständige hoch performante Datenbankengine. Das Interessante dabei ist, dass der Server Daten in DBF-Dateien von Visual FoxPro lesen wie auch speichern kann und dabei einige Vorteile gegenüber dem direkten Zugriff auf diese Dateien bietet. Dieser Artikel bietet Ihnen eine Einführung in Advantage und zeigt Ihnen, wie Sie am Einfachsten aus Ihren VFP-Anwendungen auf den Server zugreifen.

Einführung

Visual FoxPro ist ein hervorragendes Entwicklungswerkzeug. Seine reichhaltige und mächtige objektorientierte Sprache, der integrierte Berichtsgenerator sowie seine offene und erweiterbare interaktive Entwicklungsumgebung (IDE) machen es zu einem der besten Werkzeuge für die Entwicklung von Desktop-Anwendungen. Allerdings stellt die Datenengine sowohl die größte Stärke als auch den größten Schwachpunkt von VFP dar. Die größte Stärke, weil sie nahtlos in VFP integriert und eine der schnellsten Datenbankengines auf unserem Planeten ist und die größte Schwäche, weil die Dateistruktur der DBFs eine Quelle für Beschädigungen und Sicherheitsprobleme darstellt und dadurch sie in der Größe eingeschränkt ist. Zum Glück sind VFP-Entwickler bei der Speicherung ihrer Daten nicht auf die Verwendung von VFP-Tabellen beschränkt; VFP ist auch hervorragend als Frontend für Client-/Server-Datenbanken wie SQL Server, Oracle und Sybase geeignet.

Dieser Artikel behandelt ein anderes Produkt im Client-/Server-Datenbankmarkt: Advantage Database Server. Zunächst werden wir uns ansehen, was der Advantage Database Server ist und welche Features er bereitstellt und untersuchen anschließend, wie Sie aus VFP-Anwendungen heraus auf Advantage zugreifen. Für den Fall, dass Sie relativ neu in der Materie der Client-/Server-Technologien sind, werde ich davon ausgehen, dass Sie über keine umfangreichen Erfahrungen im Zugriff auf Backend-Datenbanken verfügen und werde den Zugriff detailliert beschreiben.

Einführung in den Advantage Database Server

Advantage Database Server, kurz ADS, stammt von Sybase iAnywhere, einer Division von Sybase. „Advantage Database Server ist ein hoch-performantes Client/Server Datenbank Management System mit vollem Funktionsumfang, das speziell auf die Anforderungen der Entwickler von Geschäftsapplikationen abgestimmt ist..“ Je mehr Sie über ADS lesen, desto klarer wird Ihnen, dass die Features von ADS gut mit denen der Datenengine von VFP harmonisieren. Allerdings wird VFP nicht ersetzt. Wie der SQL Server ist auch ADS eine Datenbankengine, keine vollständige Programmiersprache und Sie können mittels ODBC oder ADO von VFP aus auf einfache Weise auf die Daten von ADS zugreifen. Sie werden aber feststellen, dass ADS eine bessere Unterstützung für VFP bietet als jede andere Datenbankengine und die neueste Version 9 erweitert diese Unterstützung noch weiter.

Hier eine Übersicht der Features von ADS im Vergleich zu VFP:

- ADS ist eine echte Client-/Server-Datenbankengine. Bei dateibasierten Engines wie VFP fungiert der Server, der die Datendateien enthält, lediglich als Dateiserver. Jede Verarbeitung, beispielsweise die Auswahl von Datensätzen, erfolgt auf der Workstation, so dass die gesamte Tabelle vom Server übertragen werden muss. Bei Client-/Server-Engines wird die gesamte Verarbeitung auf dem Server ausgeführt, so dass nur die Ergebnisse der Abfragen an die Workstation gesendet werden. Dadurch werden verschiedene Vorteile erreicht, beispielsweise eine geringere Netzwerklast sowie vermehrte Möglichkeiten des Datenbankmanagements. Zusätzlich ist die Engine multithreaded und unterstützt mehrere Prozessoren, was eine bessere Skalierbarkeit ermöglicht.
- ADS wird mit zwei Datenbankengines ausgeliefert: lokal und remote. Die lokale Engine ist kein echter Datenbankserver, entspricht aber

eher VFP in der Hinsicht, dass ein dateibasierter Zugriff auf die Daten erfolgt. Sie verwendet eine In-Process-DLL, die in den ODBC-Treiber auf dem Client geladen wird. Die remote Engine ist ein echter Datenbankserver, der alle Vorteile einer Client-/Server-Architektur bietet. Advantage Local Server ist für Tests auf einem einzelnen Entwicklungssystem oder als Low-Cost-Datenbankengine für kommerzielle Anwendungen geeignet (diese Engine wird kostenfrei ausgeliefert). Sie enthält aber viele signifikante Einschränkungen, die im Advantage Remote Server nicht vorhanden sind. Der Vorteil des Advantage Local Servers ist, dass Sie über einen Client-/Server-ähnlichen Mechanismus verfügen, den Sie, falls erforderlich, auf den vollständigen remoten Server skalieren können (ohne Quellcodeänderung).

- Falls erforderlich können Sie auf den remoten Server über das Internet zugreifen. Er unterstützt die verschlüsselte und komprimierte Datenübertragung und bietet damit Mechanismen für die Sicherheit und Performance.
- Eine interessante Eigenschaft von ADS ist, dass Sie für die Speicherung der Daten wahlweise ein proprietäres Dateiformat (Dateien mit der Namenserweiterung ADT) oder das DBF-Format nutzen können. Auch wenn die Verwendung von ADT-Dateien Vorteile bietet, beispielsweise zusätzliche Datentypen, die von DBFs nicht unterstützt werden, erleichtert die Verwendung von DBFs die Migration von bestehenden VFP-Anwendungen auf das Client-/Server-Modell. Interessant ist daran, dass Sie über ADS auf Ihre vorhandenen DBFs zugreifen können, während Sie sie auch weiterhin direkt als VFP-Tabellen verwenden. Dadurch haben Sie eine sehr attraktive Migrationsstrategie: Sie können Ihre Anwendungsmodule einzeln auf Client-Server-Techniken umstellen, während ältere Module weiterhin unverändert funktionieren.
- Beim Zugriff auf DBF-Dateien werden zwei Sperrmechanismen unterstützt: kompatibel und proprietär. Die kompatible Sperre, die Sperrmechanismen des Betriebssystems verwendet, um Bytes in den DBF-Dateien zu sperren, ermöglicht den simultanen Zugriff durch ADS und Nicht-ADS-Anwendungen (beispielsweise durch VFP). Die proprietäre Sperre nutzt einen internen Sperrmechanismus. Sie bietet eine bessere Stabilität und Kontrolle, erfordert aber, dass die Dateien exklusiv von ADS geöffnet werden und dass es nicht möglich ist, auf die Dateien mit anderen Anwendungen schreibend zuzugreifen, bevor sie von ADS wieder geschlossen werden.
- ADS stellt Mechanismen für die Datensicherheit bereit. Für die Verbindung mit der Datenbank ist ein gültiges Anwenderkonto erforderlich, so dass nicht-autorisierte Anwender nicht auf Ihre Daten zugreifen können. Unterschiedliche Benutzerkonten können unterschiedliche Genehmigungsstufen aufweisen. Als Beispiel ist es unwahrscheinlich, dass normale Anwender Tabellen ändern, erstellen oder löschen müssen. Aus diesem Grund ist es möglich, nur den Administratoren diese Aufgaben zu erlauben. Auch wenn Sie ADS mit DBF-Dateien verwenden, können Sie diese Dateien in einem Verzeichnis auf dem Server zusammenfassen, auf das die normalen Anwender keinen Zugriff haben, so dass sie über ADS mit den Daten arbeiten müssen. Diese Lösung ist etwas schwieriger zu implementieren als eine reine VFP-Lösung.
- Um eine zusätzliche Sicherheit zu erhalten können Sie die Tabellen mit einem Passwort verschlüsseln, das zwischen Groß- und Kleinschreibung unterscheidet. Wenn Sie dies in einer reinen VFP-Anwendung machen, ist es erforderlich, dass Sie dafür ein Produkt eines Drittherstellers verwenden, beispielsweise Cryptor von Xitech, und dass Sie den Zugriff auf die verschlüsselten Daten selbst verwalten.
- Wie bei VFP können auch ADS-Tabellen als freie Tabellen oder innerhalb eines Data Dictionarys (einer ADD-Datei) vorhanden sein. Wie beim Datenbankcontainer von VFP (DBC) enthalten die ADD-Dateien nicht die Tabellen, sondern stellen zusätzliche Informationen oder Metadaten über die Tabellen bereit. Advantages Data Dictionary bildet den DBC von VFP weitgehend nach, auch Eigenschaften wie lange Feldnamen, Primärschlüssel, Regeln für die Validierung von Feldern, Regeln für die referenzielle Integrität, Standardwerte für einzelne Felder und benutzerdefinierte Fehlermeldungen (auch wenn ADS nur Minimalwerte, Maximalwerte oder Nullwerte unterstützt, keine Ausdrücke, die beliebige Überprüfungen durchführen können), Regeln für die Validierung von Tabellen und benutzerdefinierte Fehlermeldungen, Ansichten, Trigger und gespeicherte Prozeduren. Weiter hinten in diesem Artikel werde ich beschreiben, dass ADS ein Hilfsprogramm enthält, das ein ADD aus einem DBC generiert und damit den Großteil der Arbeit der Erstellung eines Data Dictionary automatisiert.
- Obwohl die Dokumentation von ADS den Begriff „Advantage optimized filters“ verwendet, erinnert die Technologie, die die Abfragen von ADS beschleunigt stark an Rushmore, die Technologie, durch die VFP seine Geschwindigkeit erhält. ADS prüft einen Index, um festzulegen, welche Datensätze den Filterbedingungen entsprechen und greift nur dann auf die physika-

lischen Datensätze zu, wenn ein Index nicht zur Verfügung steht. Die ADS-Dokumentation verwendet genau wie die VFP-Dokumentation Begriffe wie „vollständig optimiert“ und „teilweise optimiert“. Das bedeutet, dass VFP-Entwickler ihr vorhandenes Wissen über die Optimierung von VFP-Abfragen auch mit ADS-Datenbanken nutzen können.

- ADS enthält eine Volltextsuche, die eine sehr schnelle Suche in Memofeldern anbietet. Viele VFP-Entwickler nutzen Produkte von Drittherstellern, beispielsweise PhDbase, um in ihren Anwendungen eine Volltextsuche durchzuführen, aber viele dieser Werkzeuge werden nicht mehr vertrieben oder wurden nicht mehr aktualisiert, um mit den neuen Versionen von VFP zu arbeiten.
- Obwohl ADS auf DBF-Dateien zugreifen kann, hat es nicht die gleichen Einschränkungen wie VFP. Als Beispiel sind in VFP DBF- und FPT-Dateien auf zwei Gigabyte beschränkt. In ADS gibt es keine direkte Einschränkung der Dateigröße; stattdessen liegt die Beschränkung bei zwei Milliarden (2.147.483.648) Datensätzen. Selbstverständlich können Sie nur mit ADS auf Ihre DBF zugreifen, wenn diese größer als zwei Gigabyte wird, VFP betrachtet die Datei dann als ungültig.
- Da der ADS ODBC-Treiber die Datentypen von VFP 9 vollständig unterstützt, können Sie ihn anstelle des ODBC-Treibers von VFP verwenden. Der ODBC-Treiber von VFP wurde seit VFP 6 nicht aktualisiert und unterstützt daher nicht die neuen Datentypen wie Varchar, Varbinary und Blob.
- ADS unterstützt Transaktionen mit Commit, Rollback und automatischem Rollback, wenn die Workstation oder der Server während der Transaktion abstürzt.
- Die Replikation ist ein Prozess, der Änderungen an den Datensätzen der Tabellen einer Datenbank in die Tabellen einer anderen Datenbank schreibt; beispielsweise können die Änderungen, die an einer Datenbank in einer Filiale vorgenommen werden, in die Datenbank in der Firmenzentrale geschrieben werden. Dieser Prozess funktioniert in beide Richtungen. Auch in VFP ist die Replikation von Daten möglich, allerdings müssen Sie den Code dafür selbst schreiben und Sie müssen alle Probleme wie die Auflösung von Konflikten selbst behandeln. Diese Funktionalität müssen Sie extensiv testen, um sicherzustellen, dass sie unter allen Bedingungen korrekt funktioniert. ADS enthält alle Features der Replikation, so dass diese Arbeit bereits erledigt ist.
- ADS enthält die Fähigkeit für ein Onlinebackup, was bedeutet, dass Sie Ihre Tabellen sichern können, während sie durch eine Anwendung geöffnet sind. Das ist bei einem normalen Backup gegen VFP-Tabellen nicht möglich. Sie können vollständige oder inkrementelle Backups durchführen.

Den Advantage Database Server installieren

Advantage setzt sich aus mehreren Komponenten zusammen: dem Datenbankserver selbst, dem Advantage Data Architect, dem ODBC-Treiber und dem OLE DB Provider. Als dieser Artikel geschrieben wurde befand sich ADS Version 9 im Betastadium und stand unter http://devzone.advantagedatabase.com als Download zur Verfügung. Die endgültige Version wird für den März 2008 erwartet und die Betaversion wird am Ende dieses Monats auslaufen.

ADS läuft unter Windows, Netware und Linux. Für Windows heißt der Installer für den Datenbankserver NT.EXE. Führen Sie dieses Programm auf dem Server aus, auf dem Sie ADS installieren wollen. Selbstverständlich können sie ihn auf dem gleichen System ausführen, auf dem Sie entwickeln, nicht auf einem separaten Server, aber in der Regel werden Sie ADS in einer Produktionsumgebung auf einem Server installieren. Standardmäßig wird die Engine in

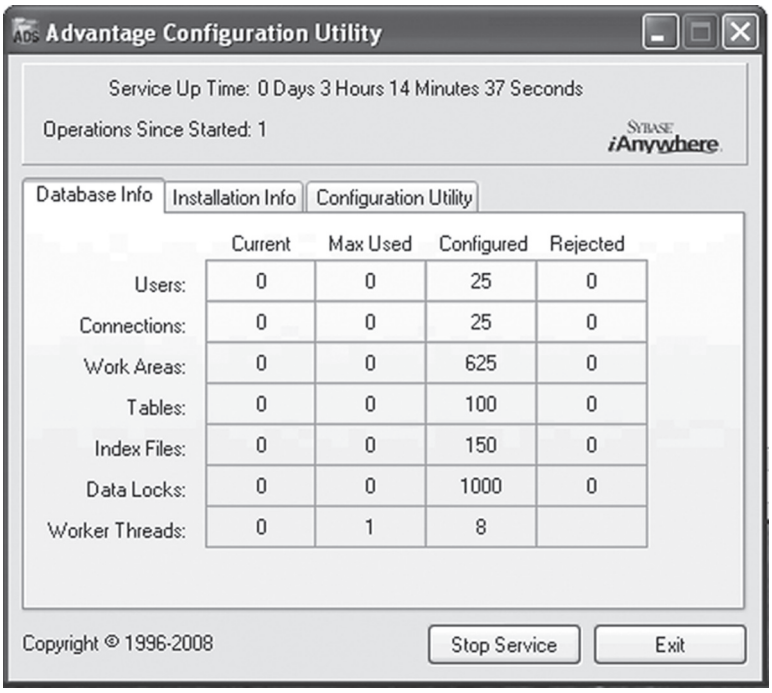

Abbildung 1. Nach der Installation von ADS wird das Configuration Utility gestartet und gibt Ihnen die Möglichkeit, die Servereigenschaften zu konfigurieren.

C:\Program Files\Advantage9.0 installiert (dies ist auch das Standardverzeichnis für die anderen Komponenten). Nach der Installation der Enginedateien fragt Sie der Installer nach dem Namen des Eigentümers, ob der Dienst der Engine automatisch oder manuell gestartet werden soll (Standardwert ist automatisch), welcher ANSI-Zeichensatz verwendet werden soll (Standardwert ist die aktuelle Einstellung der Maschine) und welcher OEM- oder lokalisierte Zeichensatz verwendet werden soll. Nachdem Sie diese Frage beantwortet haben, startet das Advantage Configuration Utility (siehe Abbildung 1), das Ihnen Statistiken über den Server anzeigt, unter anderem die Anzahl der Benutzer und Verbindungen und es gibt Ihnen die Möglichkeit, verschiedene Eigenschaften zu konfigurieren, beispielsweise den Timeout, die verwendeten Ports und den Speicherort für die Logdateien.

Anschließend müssen Sie den Advantage Data Architect installieren, ein ADS-Werkzeug, das ich im nächsten Abschnitt behandelt werde. Dessen Installer trägt den Namen Arc32.EXE. Wie bei der Serverinstallation können Sie den Namen des Installationsverzeichnisses, den ANSI-Zeichensatz und den OEM-Zeichensatz angeben. Anschließend installieren Sie den ODBC-Treiber, indem Sie ODBC.EXE ausführen, wenn Sie vorhaben, ADS über ODBC anzusprechen und installieren Sie die OLE DB Provider, indem Sie OLEDB.EXE ausführen (den OLE DB Provider sollten Sie immer installieren, unabhängig davon, ob Sie vorhaben, ADO zu verwenden oder nicht, da OLEDB.EXE ein VFP-spezifisches Hilfsprogramm installiert, das wir später in diesem Artikel noch behandeln werden). Beide Installationsprogramme fragen Sie nach dem Namen des Installationsverzeichnisses, dem ANSI-Zeichensatz und dem OEM-Zeichensatz.

Jetzt können Sie ADS verwenden.

Der Advantage Data Architect

Der Advantage Data Architect (kurz ARC) ist ein Werkzeug für die Verwaltung der Verbindungen und Datenbanken von ADS. Wenn Sie bereits den SQL Server Enterprise Manager, das Management Studio oder auch den VFP Data Explorer verwendet haben, werden Sie sich in diesem Werkzeug sofort zurechtfinden. Interessant ist, dass der vollständige Quellcode für ARC, der in Delphi geschrieben ist, mit dem Werkzeug ausgeliefert wird. Sie sehen ARC in Abbildung 2.

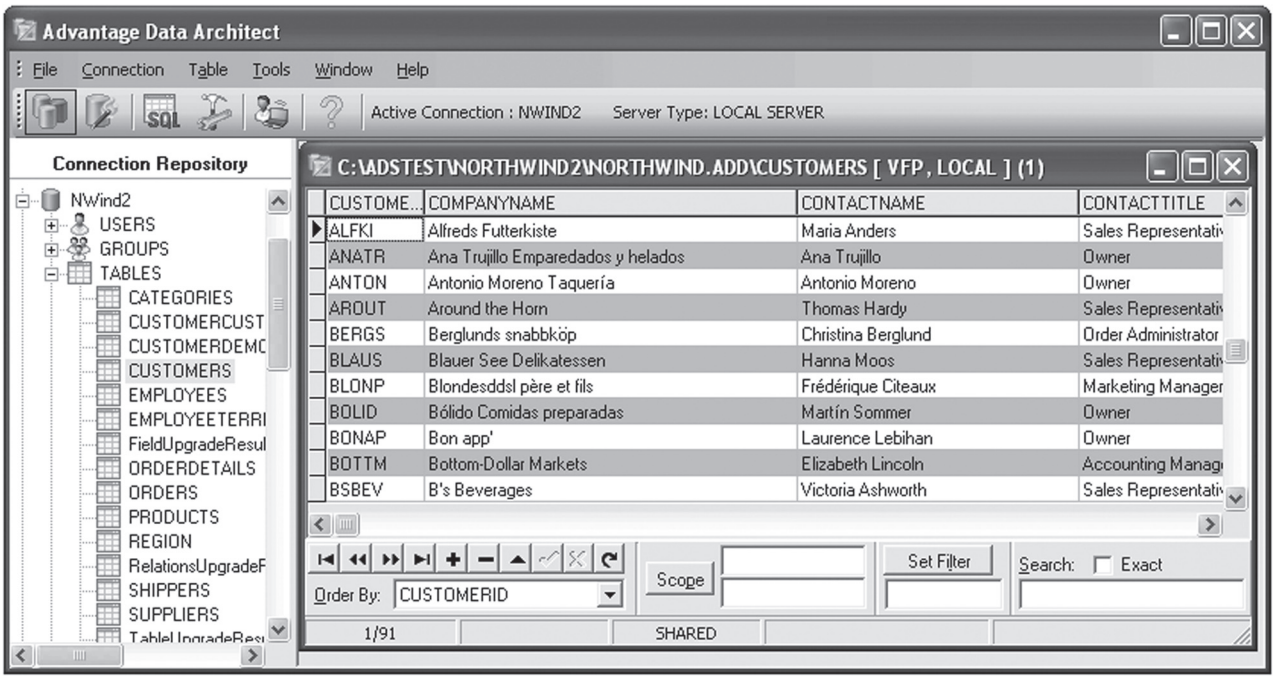

Abbildung 2. Advantage Data Architect bietet viele der Features des VFP Data Explorers oder des SQL Server Management Studios.

- Mit ARC können Sie die folgenden Aufgaben erledigen:
- Datenbanken und Tabellen erstellen, warten und löschen
- Tabellen anzeigen und sie filtern, suchen, sortieren und in ihnen navigieren
- Daten importieren und exportieren
- Tabellenstrukturen als Code exportieren
- Sicherheitseinstellungen und Benutzerkonten verwalten
- Mit den Werkzeugen SQL Utility und Query Builder Abfragen ausführen
- Data Dictionarys vergleichen

Das linke Fenster im ARC enthält das Connection Repository. Es ermöglicht Ihnen den einfachen

Zugriff auf die ADS-Datenbanken, die Sie mit ARC registriert haben (eine Datenbank muss nicht im ARC registriert sein, wenn Sie darauf aus anderen Anwendungen heraus zugreifen wollen). Um eine Datenbank zu erstellen und sie dem Repository hinzuzufügen, wählen Sie im Menü File den Eintrag Create New Data Dictionary; damit erstellen Sie eine leere ADD-Datei mit den Eigenschaften, die Sie in dem daraufhin angezeigten Dialog angeben. Um dem Repository eine vorhandene Datenbank oder ein Verzeichnis mit freien Tabellen hinzuzufügen, wählen Sie im Menü File den Eintrag New Connection Wizard und folgenden Sie den Schritten im Dialog des Assistenten.

In den Beispielen dieses Artikels werden Sie verschiedene Funktionen des ARC verwenden.

Eine VFP-Datenbank upsizen

Obwohl ADS 9 die meisten Datenfeatures von VFP unterstützt, ist die Unterstützung teilweise davon abhängig, dass eine Advantage-Datenbank verwendet wird und keine freien Tabellen (aus der Sicht von ADS sind auch Tabellen in einer DBC freie Tabellen, wenn sie sich nicht in einer ADS-Datenbank befinden). Dazu gehören die Unterstützung langer Feldnamen, Primärschlüssel, die referenzielle Integrität, die Prüfung von Feld- und Tabellenwerten, Trigger usw.; Anders ausgedrückt also die gleichen Dinge, für die der Datenbankcontainer von VFP verwendet wird. Außer in kleinen Datenbanken bedeutet es einige Arbeit, eine Advantage-Datenbank für eine vorhandene VFP-Datenbank zu erstellen. Zum Glück wird ADS mit dem in VFP geschriebenen Hilfsprogramm ADSUpsize ausgeliefert, das eine Advantage-Datenbank erstellt und sie mit den Informationen über die Tabellen in einer VFP-Datenbank füllt. Der Name ADSUpsize.PRG ist etwas unglücklich gewählt, da das Programm den VFP-DBC nicht „upsized“ oder irgendwelche Änderungen an den DBF-Dateien vornimmt. In der endgültigen Version von ADS 9 hat Sybase iAnywhere das Hilfsprogramm in DBCConvert.PRG umbenannt.

Um die Arbeitsweise von ADSUpsize.PRG zu sehen, upsizen Sie die Beispieldatenbank Northwind, die mit VFP ausgeliefert wird. Beginnen Sie damit, dass Sie ein neues Verzeichnis erstellen und alle Dateien aus dem Verzeichnis Samples\Northwind unterhalb des Hauptverzeichnisses von Visual FoxPro dorthin kopieren. Auf diese Weise ändern Sie Ihre ursprüngliche Datenbank nicht, wenn Sie die weiter unten beschriebenen Schritte ausfuhren. Starten Sie VFP und rufen Sie die Kopie von ADSUpsize.PRG auf, die Sie im Quellcode zu diesem Artikel finden. Das Programm enthält gegenüber dem mit der Betaversion ausgelieferten Programm einige Änderungen, die im Kasten „Das ADS VFP Upsizing Utility fixen“ beschrieben sind. Wenn Sie nach der Datenbank gefragt werden, wählen Sie Northwind.DBC in dem Verzeichnis, in das Sie die Dateien kopiert haben. Nach einigen Sekunden ist das Programm mit seiner Arbeit fertig. Beachten Sie aber die angezeigte Nachricht: es sind 15 Fehler vorhanden, aber es wird nicht angezeigt, worin die Fehler bestehen. Zum Glück schreibt ADSUpsize.PRG den Upsizing-Prozess mit. Das werde ich später beschreiben.

Sehen Sie sich das Verzeichnis an, das die Datenbank Northwind enthält, und Sie sehen einige neue Dateien:

- Northwind.ADD, AI und AM: die ADS-Datenbank.
- BAK-Versionen jeder DBF und FPT: Der Upsizing-Prozess (nicht ADSUpsize.PRG, sondern ADS selbst) legt diese Dateien an.
- FieldUpgradeResults.ADM und ADT, RelationUpgradeResults.ADM und ADT, TableUpgradeResults.ADM und ADT sowie ViewUpgradeResults.ADM und ADT: Diese ADS-Tabellen enthalten Loginformationen über den Upsizing-Prozess einschließlich aller aufgetretenen Fehler. Sie werden diese Tabellen extensiv nutzen, um Probleme zu finden und zu lösen, die während des Upsizing-Prozesses aufgetreten sind.
- FAIL_EMPLOYEES.DBF und FPT: Mit diesen Dateien befassen wir uns später.

Öffnen Sie ARC und wählen Sie im Menü File den Eintrag New Connection Wizard oder klicken Sie in der Toolbar auf die Schaltfläche New Connection Wizard. Im ersten Schritt des Assistenten wählen Sie „Create a connection to an existing data dictionary“. In Schritt 2 geben Sie einen Namen für die Datenbank und den Pfad zu Northwind.ADD an, also zur Advantage-Datenbank, die durch das Upsizing Utility erstellt wurde. Die anderen Einstellungen belassen Sie auf deren Standardwerten und klicken auf Finish (siehe Abbildung 3). ARC fragt Sie nach dem Login für den Benutzer AdsSys (der Standardname für den Administrator); da in diesem Beispiel kein Passwort für diesen Anwender angegeben wurde klicken Sie auf OK.

Unterhalb des Knotens Tables finden Sie alle Tabellen von Northwind, aber unter Views werden nur fünf der 17 Ansichten angezeigt. Zusätzlich werden Ihnen die bereits erwähnten Tabellen FieldUpgradeResulst, RelationsUpgradeResults, TableUpgradeResults und ViewUpgradeResults aufgelistet.

Öffnen Sie TableUpgradeResults, indem Sie darauf doppelt klicken. Jede behandelte Tabelle wird mehrfach aufgelistet, jeweils einmal für jede Operation. Die unterschiedlichen Felder dieser Tabelle zeigen an, welcher Prozess in jedem Schritt ausgeführt wurde. Als Beispiel finden Sie Datensätze für

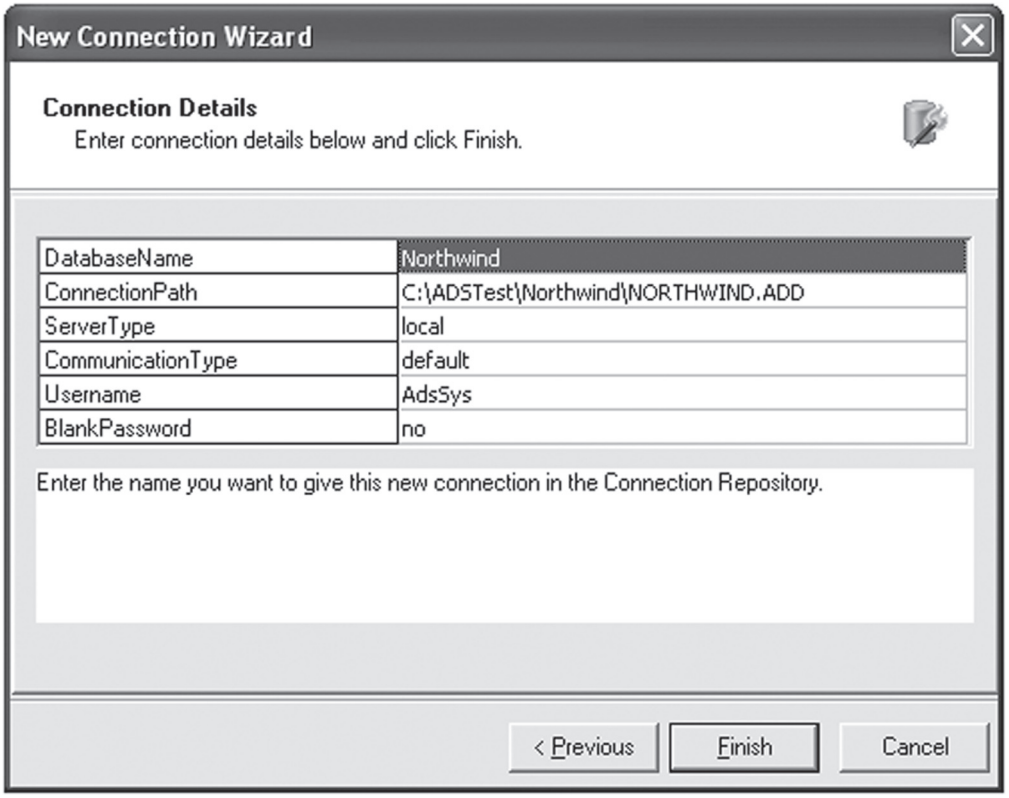

Abbildung 3. Verwenden Sie den New Connection Wizard, um im Advantage Database Architect eine Verbindung zu Ihrer upgesizeten VFP-Datenbank zu erstellen.

Customers, in denen ausgesagt wird, dass die Tabelle der ADS-Datenbank hinzugefügt wurde, die langen Feldnamen werden angegeben, die Indizes werden erstellt, der Ausdruck und die Nachricht für die Tabellenprüfung werden angegeben und der Primärschlüssel wird definiert. Beachten Sie, dass eine Tabelle, Categories, bei der Angabe der langen Feldnamen einen Fehler hervorgerufen hat. Die Fehlermeldung lautet (hier aus Platzgründen editiert):

```
The requested operation is not legal for the
given field type.  ALTER TABLE CATEGORIES
ALTER "CATEGORYID"  "CATEGORYID" autoinc
NOT NULL ALTER "CATEGORYNA" "CATEGORYNAME"
char( 15 ) NOT NULL ALTER "DESCRIPTIO"
"DESCRIPTION" memo NULL ALTER "PICTURE"
"PICTURE" blob  NULL
```

Öffnen Sie FieldUpgradeResults. Diese Tabelle zeigt die Ergebnisse des Upsizings der Feldkommentare, der Prüfmeldungen und Standardwerte. Auch hier enthält ein Feld einen Fehler, der anzeigt, dass das Upsizing Utility den Kommentar für Categories.CategoryName nicht einstellen konnte; dieses Feld ist im Data Dictionary nicht vorhanden, der Fehler, der in TableUpgradeResults festgehalten wurde, verhindert, dass die langen Feldnamen für Categories definiert wurden. Aus diesem Grund trägt das Feld den Namen CategoryNa, den Namen mit zehn Buchstaben, der im Header der DBF gespeichert ist.

RelationsUpgradeResults enthält die Loginformationen für die Relationen. Diese Tabelle enthält keine Fehler. Allerdings sind in der Tabelle ViewUpgradeResults, die die Loginformationen für die upgesizeten Ansichten enthält, viele Fehler enthalten. Bis auf fünf Ansichten konnten alle Ansichten nicht upgesizet werden. Einige der Ansichten sind fehlgeschlagen, da sie das Feld Categories.CategoryName referenzieren, das nicht vorhanden ist, andere schlugen aus anderen Gründen fehl.

Setzen Sie in VFP SET DELETED auf OFF und öffnen Sie anschließend die Tabelle Employees. Beachten Sie, dass einer der Datensätze, Andrew Fuller, als gelöscht markiert ist. Wenn Sie die Tabelle FAIL_EMPLOYEES öffnen, die während des Upsizings angelegt wurde, finden Sie Andrews Datensatz dort. Weshalb wurde dieser Datensatz während des Upsizings gelöscht? Sie werden gleich erkennen, was geschehen ist.

Während des Upsizings durchläuft eine VFP-Datenbank viele Schritte, bis Sie über eine vollständige ADS-Version der Datenbank verfügen. Dabei sind einige Dinge zu beachten.

- ADS kennt keine eingebetteten JOINs, kann also beispielsweise den Befehl SELECT * FROM Table1 JOIN Table2 JOIN Table3 ON Table2. field = Table3.Field ON Table1.Field = Table2. Field nicht verarbeiten. Sie müssen Ansichten, die eingebettete JOINs verwenden, in die normalere JOIN-Syntax umwandeln, bevor Sie sie

upsizen. Eine Möglichkeit dafür ist, dass Sie die Ansicht im Ansichten-Designer öffnen und wieder speichern. Das funktioniert, obwohl in älteren Versionen von VFP der Ansichten-Designer die eingebettete Syntax verwendete, da er seit VFP 8 standardmäßig die sequentielle Syntax nutzt.

- ADS unterstützt keine Ansichten mit der Klausel ORDER BY.
- Ansichten, die VFP-Funktionen verwenden, werden nicht korrekt umgewandelt. Als Beispiel enthält die Ansicht Sales_Totals_By_Amount in der Datenbank Northwind in einer der WHERE-Bedingungen die VFP-Funktion BETWEEN(). In diesem Fall löst die Verwendung der SQL-Klausel BETWEEN das Problem. Andere Ansichten könnten allerdings nicht so einfach zu fixen sein. Als Beispiel verwenden die Ansichten Summary_of_Sales_by_Year und Summary_of_Sales_by_Quarter in ihren WHERE-Klauseln die Funktionen EMPTY() und NVL(), so dass sie nicht umgewandelt werden können und manuell neu erstellt werden müssen.
- Tabellen mit Feldern vom Typ General werden nicht korrekt umgewandelt, da ADS diesen Datentyp nicht unterstützt (ein weiterer Grund, keine Felder vom Typ General zu verwenden). Daher hat die Tabelle Categories einen Fehler ausgelöst: Categories.Picture ist ein Feld vom Typ General. Wenn Sie versuchen, die Tabellenstruktur in ARC anzeigen zu lassen (klicken Sie dafür mit der rechten Maustaste auf die Tabelle und wählen im Kontextmenü den Eintrag Properties), erhalten Sie eine Fehlermeldung und für das Feld werden keine Eigenschaften angezeigt. Entweder müssen Sie das Feld Picture aus der Tabelle entfernen oder Sie müssen den Typen ändern, beispielsweise in Blob.
- ADS unterstützt VFPs Feldeigenschaften Standardwert, ob der Wert NULL erlaubt ist oder nicht, eine Beschreibung (die von der Feldeigenschaft Comment übernommen wird), die Nachricht für die Feldprüfung, sowie die Angabe, ob eine Codeseitenumwandlung durchgeführt werden soll, so dass diese Einstellungen alle übernommen werden. ADS unterstützt die Regeln für die Feldvalidierung nicht (mit Ausnahme der Angabe eines Höchst- und eines Mindestwerts). Aus diesem Grund könnten Sie diese Eigenschaften nach dem Upsizing manuell füllen.
- Andere Feldeigenschaften, die von ADS nicht unterstützt werden, sind Formatierungen, Feldmappings und Feldüberschriften, die alle einen Fehler auslösen. Diese Eigenschaften beziehen sich auf die Benutzeroberfläche, so dass sie in ADS nicht erforderlich sind.
- Die von ADS unterstützten Tabelleneigenschaften von VFP sind die Größe der Memoblöcke, die Tabellenbeschreibung (die von der Tabelleneigenschaft Comment übernommen wird), die Regeln für die Tabellenvalidierung, sowie die Validierungsnachricht, so dass diese Eigenschaften übernommen werden. Allerdings sind Regeln, die eine VFP-Funktion enthalten, offensichtlich ein Problem.
- Trigger werden nicht übernommen, da sie VFP-Code verwenden, der von ADS nicht verstanden wird. Allerdings werden Trigger in der Regel für die Sicherstellung der referenziellen Integrität verwendet. Diese Trigger werden übernommen. Beachten Sie, dass ADS die RI-Regel Ignore nicht unterstützt, so dass diese Regel als auf NULL gesetzt übernommen wird. Trigger, die anderen Zwecken dienen, müssen Sie neu erstellen.
- Aus dem gleichen Grund werden auch gespeicherte Prozeduren nicht übernommen. Um die gespeicherten Prozeduren, die für die RI verwendet werden, müssen Sie sich keine Gedanken machen, da diese übernommen werden. Alle anderen gespeicherten Prozeduren müssen Sie neu schreiben oder Sie verschieben den Code in eine Komponente der mittleren Schicht.
- Die referenzielle Integrität wird während des Upsizing-Prozesses erzwungen. Aus diesem Grund wurde auch der Datensatz von Andrew Fuller in Employees gelöscht. Die Tabelle Employees enthält im Feld EmployeeID den Vorgesetzten jeder Person und Andrews Wert in ReportsTo ist auf 0 gesetzt, einen Wert, der keiner EmployeeID eines Datensatzes entspricht. Die Regel Insert für die referenzielle Integrität ist auf Ignore gesetzt, so dass VFP für diesen Datensatz keinen Fehler erzeugt. Allerdings verfügt ADS über keine Insert-Regel für die referenzielle Integrität, sondern lediglich für Update und Delete, die beide auf Restrict gesetzt sind. Da der Wert ReportsTo in Andrews Datensatz ungültig ist, schlägt die RI-Regel fehl und der Datensatz wird gelöscht. Sie könnten der Meinung sein, dass das Löschen des Datensatzes etwas hart ist; eventuell könnte ADS ReportsTo stattdessen auf NULL setzen. Interessant ist, dass es in VFP möglich ist, die Löschmarkierung wieder zu entfernen.
- ADS Version 9 unterstützt die Binärindizes von VFP nicht, aber Sybase plant, sie in der Version 9.1 oder in einem Servicepack, das vor der Version 9.1 veröffentlicht wird, zu unterstützen.

Sie können einige dieser Probleme beseitigen und den Upsizing-Prozess anschließend wiederholen. Da ADSUpsize.PRG eine ADS-Datenbank erstellt, kann diese Datenbank nicht im ARC geöffnet werden. Schließen Sie daher die Verbindung zu Northwind, indem Sie mit der rechten Maustaste auf die Verbindung klicken und im Kontextmenü Disconnect wählen. Öffnen Sie die Datenbank Northwind in VFP und nehmen Sie die folgenden Änderungen vor:

- Invoices und Product_Sales_For_1997: Ändern Sie diese Ansichten, löschen Sie die Relationen und erstellen Sie sie neu. Diese Ansichten verwenden eingebettete JOINs, so dass das Neuerstellen in VFP 9 sie in die sequentielle JOIN-Syntax umwandelt (verwenden Sie die Funktion View SQL im Ansichten-Designer, um dies zu überprüfen). Obwohl Sie diesen Schritt auch mit der Ansicht Sales_By_Category vornehmen können, kann diese Ansicht aus anderen Gründen nicht übernommen werden, wie Sie später erkennen können.
- Quarterly_Orders und Sales_Totals_By_Amount: Öffnen Sie diese Ansichten zum Ändern, speichern Sie sie und schließen sie, ohne daran Änderungen vorzunehmen. Diese Ansichten verwenden in einer der WHERE-Bedingungen die VFP-Funktion BETWEEN() (Product_Sales_For_1997 und Sales_By_Category verwenden ebenfalls BETWEEN(), aber diese Ansichten werden durch den vorherigen Schritt beim Speichern der Ansicht automatisch gefixt).
- Alphabetical_List_of_Products, Current_Product_List, Invoices und Products_By_Category: Ändern Sie diese Ansichten und entfernen Sie die Klausel ORDER BY. Leider funktioniert dies nicht mit Ten_Most_Expensive_Products, da diese Ansicht die Klausel TOP enthält und daher die Klausel ORDER BY benötigt. Diese Ansicht müssen Sie manuell neu erstellen. Obwohl Sie diese Änderung auch in den Ansichten Sales_By_Category, Summary_of_Sales_by_Quarter und Summary_of_Sales_by_Year vornehmen könnten, würde das nichts helfen, da diese Ansichten aus anderen Gründen nicht übernommen werden können.
- Invoices: Öffnen Sie diese Ansicht zum Ändern, wählen Sie View SQL und ändern Sie beide Aufrufe von ALLTRIM in TRIM, da ALLTRIM() in ADS im Gegensatz zu TRIM() keine unterstützte Funktion ist.
- Categories: Öffnen Sie diese Ansicht zum Ändern und entfernen Sie das Feld Picture.

Führen Sie den Befehl CLOSE TABLES ALL aus und führen Sie ADSUpsize.PRG erneut aus. Jetzt erhalten Sie nur vier Fehler. Öffnen Sie die Verbindung zu Northwind in ARC und prüfen Sie die Logtabellen. Diesmal wurde Categories korrekt übernommen, so dass alle Fehler in den Ansichten auftreten:

- Sales_by_Category kann nicht upgesizet werden, da sie eine andere Ansicht abfragt, was in ADS nicht unterstützt wird.
- Ten_Most_Expensive_Products enthält die Klausel TOP und erfordert daher die Klausel ORDER BY, die in ADS nicht unterstützt wird.
- Summary_of_Sales_by_Year und Summary_of_Sales_by_Quarter verwenden in ihren WHERE-Klauseln sowohl EMPTY() als auch NVL().

Zugreifen auf die Daten in VFP oder ADS

Eine VFP-Datenbank upzusizen bedeutet nicht, dass Sie auf diese Datenbank anschließend nicht mehr aus VFP heraus zugreifen können. Es bedeutet einfach, dass Sie jetzt über zwei Möglichkeiten verfügen, auf die Datenbank zu verwenden: als native VFP-Tabellen oder über ADS. Sie können die Daten in den Tabellen mit VFP oder ADS ändern und die andere Datenbank sieht die Änderungen. Dazu gehört auch die Unterstützung automatisch hochgezählter Felder.

Öffnen Sie als Beispiel nach dem Upsizen die Datenbank Northwind in ARC und klicken Sie doppelt auf die Tabelle Employees, um sie zu öffnen. Klicken Sie in der Toolbar am Fuß des Tabellenfensters auf die Schaltfläche „+“ (siehe Abbildung 2), um einen neuen Datensatz hinzuzufügen. Lassen Sie das Feld EmployeeID leer, füllen Sie aber die anderen Felder aus. Achten Sie darauf, im Feld ReportsTo einen gültigen Wert einzugeben (beispielsweise „2“), da die RI-Regel für die Tabelle keinen leeren oder ungültigen Wert erlaubt. Nachdem Sie den Datensatz verlassen haben, sollten Sie feststellen, dass das Feld EmployeeID automatisch mit dem nächsten Wert gefüllt wurde.

Schließen Sie jetzt das Tabellenfenster und lösen Sie im ARC die Verbindung mit der Datenbank. Öffnen Sie die Datenbank in VFP und anschließend die Tabelle Employees. Beachten Sie, dass der neue Datensatz eingefügt wurde. Fügen Sie einen weiteren Datensatz hinzu und beachten Sie, dass EmployeeID automatisch mit dem nächsten verfügbaren Wert gefüllt wird. VFP erfordert keinen gültigen Wert in ReportsTo; Sie können das Feld auch leer lassen oder einen ungültigen Wert eingeben, da die RI-Regel auf Ignore eingestellt ist.

Wie bereits ausgeführt können Sie auf Ihre Tabellen sowohl direkt aus VFP heraus als auch über

ADS zugreifen, was es Ihnen ermöglicht, eine bestehende Anwendung Modul für Modul auf ein Client-/Servermodell zu migrieren. Als Beispiel könnten Sie in einer Buchhaltung das Modul Offene Rechnungen ändern, so dass es auf die Daten über ADS zugreift und dieses Modul ausliefern, nachdem die Änderungen beendet und vollständig getestet wurden. Die anderen Module würden weiterhin aus VFP heraus auf die Tabellen zugreifen. Der Vorteil dieser Lösung besteht darin, dass Sie nicht die gesamte Anwendung auf einmal migrieren müssen, was eine erheblich längere Entwicklungszeit sowie einen deutlich höheren Testaufwand erfordern würde.

Wenn Sie mit der Entwicklung einer neuen Anwendung beginnen und nicht auf die Abwärtskompatibilität der Daten achten müssen, könnten Sie überlegen, statt der DBF-Dateien native ADS-Tabellen (ADT-Dateien) zu verwenden. ADT-Dateien haben gegenüber DBFs viele Vorteile. Dazu gehören zusätzliche Datentypen (beispielsweise Zeichenfelder, die zwischen Groß- und Kleinschreibung nicht unterscheiden), längere Feldnamen, größere Dateien, mehr als 255 Felder in einer Tabelle sowie die automatische Wiederverwendung gelöschter Datensätze.

Die Volltextsuche des ADS

ADS verfügt über eine schnelle und mächtige Volltextsuche (FTS). FTS setzt einen Index auf jedes Wort in einem Memofeld ein, um die gesuchten Worte schnell aufzufinden. Um FTS auf einer Tabelle zu ermöglichen, müssen Sie einen FTS-Index auf ein oder mehrere Felder in der Tabelle erstellen.

Wenn Sie die Performance von FTS testen wollen, aber keine großen Tabellen mit viel Inhalt in Memofeldern zur Verfügung haben, kann Ihnen das Programm MakeDemoMemo.PRG helfen, das Sie im Quellcode zu diesem Artikel finden. Es prüft Ihre gesamte Festplatte und sucht nach Textdateien (einschließlich PRF, TXT und HTML) sowie nach VCX- und SCX-Dateien von VFP und schreibt diese in das Memofeld Content einer Tabelle namens DemoMemo.DBF. Sie können sich vorstellen, dass dieses Programm einige Zeit beschäftigt ist. Ein Test benötigte etwa zehn Minuten, um eine Tabelle mit 81.000 Datensätzen und einer FPT-Datei mit 545 MB zu erstellen.

Bevor wir die Tabelle ADS hinzufügen sucht ein Testprogramm alle Vorkommen des Worts „tableupdate“ im Feld Content:

```
select * from DemoMemo ;
  where atc('tableupdate', Content) > 0 ;
  into cursor Temp
select * from DemoMemo ;
  where 'tableupdate' $ Content ;
  into cursor Temp
```

Die erste Anweisung, die bei der Suche nicht zwischen Groß- und Kleinschreibung unterscheidet, benötigte für die Ausführung 305 Sekunden. Die zweite Anweisung, die zwischen Groß- und Kleinschreibung unterscheidet, benötigte dafür 65 Sekunden.

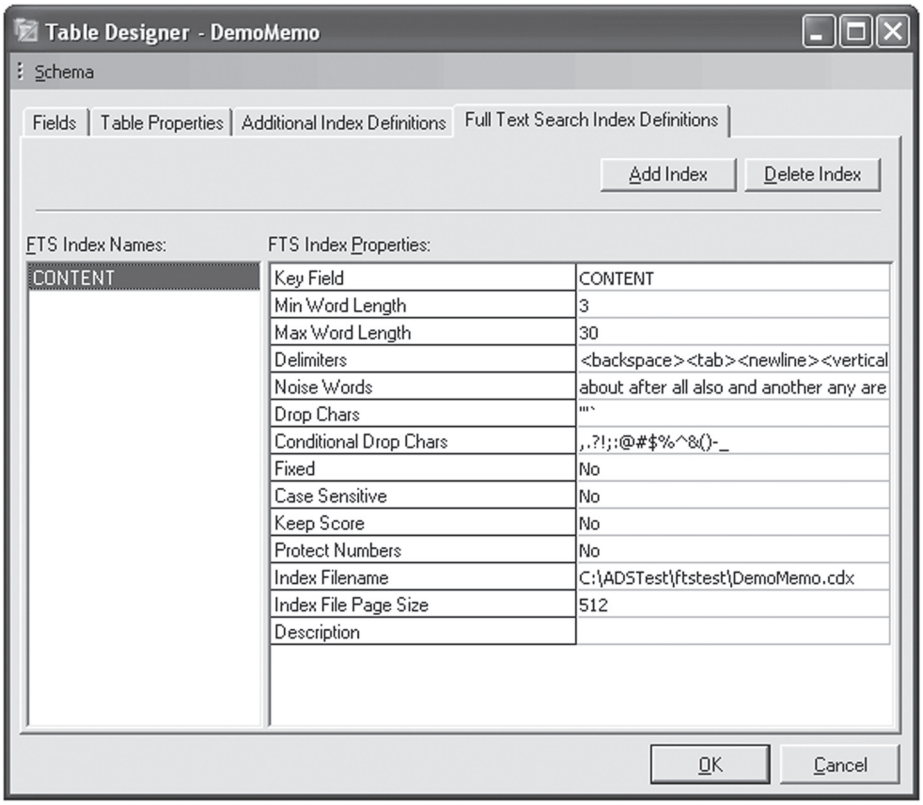

Abbildung 4. In der Seite Full Text Search Index Definitions des Tabellen-Designers können Sie Indizes erstellen, die eine schnelle und mächtige Volltextsuche ermöglichen.

Hier die Schritte, mit denen Sie diese Tabelle für die FTS vorbereiten:

- Öffnen Sie ARC und wählen Sie im Menü den Eintrag New Connection Wizard. Wählen Sie „Create a connection to a directory of existing tables“ und klicken Sie auf Next. Geben Sie den Namen der Datenbank an, wählen Sie das Verzeichnis, das DemoMemo.DBF enthält, wählen

Sie als TableType den Wert „vfp“ und klicken Sie auf OK. Beachten Sie, dass dadurch auf DemoMemo als freie Tabelle zugegriffen wird, nicht über ein Data Dictionary von ADS, da es in der Betaversion ein Problem mit dem Zugriff auf TFS-Indizes von DBF-Dateien durch ein Data Dictionary gibt.

- Klicken sie mit der rechten Maustaste im Knoten Tables auf die Tabelle DemoMemo und wählen Sie im Kontextmenü den Eintrag Properties. Wählen Sie die Seite Full Text Index Definitions (siehe Abbildung 4), klicken Sie auf Add Index und geben Sie einen Namen für den Index an. Als Key Field wählen Sie CONTENT (das Memofeld, das den zu indizierenden Text enthält). Sie können noch einige Feineinstellungen für den Index vornehmen, beispielsweise die minimale und maximale Wortlänge, Sie können Worte wie „and“ und „the“ aus dem Index ausschließen und Sie können angeben, ob der Index zwischen Groß- und Kleinschreibung unterscheiden soll oder nicht. Klicken Sie auf OK, um den Index zu erstellen.

Die Erstellung des FTS-Index kann einige Zeit in Anspruch nehmen, da ADS für jedes Wort in jedem Datensatz einen Indexeintrag erstellt. Nachdem der Index erstellt wurde, wird er standardmäßig wie jeder andere Index aktualisiert, wenn Datensätze hinzugefügt, geändert oder gelöscht werden. Sie können die automatische Aktualisierung auch abschalten und den Index bei Bedarf neu erstellen, um eine bessere Performance bei der Änderung der Datensätze zu erreichen.

Nachdem der FTS-Index erstellt wurde, benötigt die folgende Anweisung, die durch die ADS-Engine ausgeführt wird, 0,070 Sekunden für eine Suche, die zwischen Groß- und Kleinschreibung unterscheidet:

```
select * from DemoMemo ;
    where contains(Content, 'tableupdate')
```

Die Volltextsuche kann aber noch mehr als nur nach der Existenz eines Worts zu suchen. Beispielsweise können Sie die folgenden Suchvorgänge ausführen:

- Sie können alle Felder nach einem bestimmten Wort durchsuchen, indem Sie als Feldnamen „*“ angeben.
- Sie können mit der Funktion SCORE nach mehrfachem Auftreten des gleichen Worts in einem gegebenen Datensatz suchen. Auch wenn es für diese Funktion nicht zwingend erforderlich ist, bringt ihnen die Aktivierung der Eigenschaft Keep Score für den Index eine bessere Performance, da der Scorewert im Index gespeichert wird.

```
select * from ;
DemoMemo where ; contains(Content,;
'tableupdate') and ;
score(Content, ;
'tableupdate') > 1
```

Sie können nach einem Wort in der Nähe eines anderen Wortes suchen:

```
select * from ;
DemoMemo where ; contains(Content,;
'tableupdate near' + ;
'aerror')
```

Zu bemerken ist noch, dass Sie nicht mehr aus VFP direkt auf eine Tabelle zugreifen können, nachdem Sie einen FTS-Index für die Tabelle erstellt haben.

Mit ADS verbinden

Es gibt zwei Möglichkeiten, auf Daten zuzugreifen, die durch ADS verwaltet werden: über den ODBC-Treiber oder über den OLE DB Provider (es gibt noch die dritte Möglichkeit, über die API-Funktionen von ADS auf die Daten zuzugreifen, aber diese Low Level-Funktionen erfordern viel Code). OLE DB erfordert einen Verbindungsstring, während ODBC wahlweise einen ODBC-Datenquellennamen (DSN) oder einen Verbindungsstring verwendet und daher manchmal auch DSNlose Verbindung genannt wird.

Hier ein Beispiel für einen OLE DB-Verbindungsstring:

```
provider=Advantage.OLEDB.1;data source=C:\
ADSTest\Northwind\Northwind.add; User
ID=adssys;Password=""
```

Ein ODBC-Verbindungsstring sieht ähnlich aus:

```
driver=Advantage StreamlineSQL
ODBC;DataDirectory=C:\ADSTest\Northwind\North-
wind.add; uid=adssys;pwd=""
```

Wenn Sie statt eines Verbindungsstrings einen ODBC DSN verwenden, können Sie ihn mit dem ODBC Data Source Administrator definieren. Öffnen Sie die Systemsteuerung von Windows, rufen Sie dort die Verwaltung auf und klicken Sie doppelt auf Datenquellen (ODBC). Wählen Sie dort den Tab Benutzer-DSN, um einen DSN zu erstellen, den nur Sie nutzen können oder Sie wählen den Tab System-DSN, um einen DSN zu erstellen, den jeder Benutzer verwenden kann, der sich auf Ihrem Rechner anmeldet. Klicken Sie auf Hinzufügen, um einen neuen DSN zu erstellen, wählen Sie Advantage StreamlineSQL ODBCC als Treiber und klicken Sie auf Fertig stellen. In Abbildung 5 sehen Sie den Dialog Advantage Streamline ODBC Driver Setup.

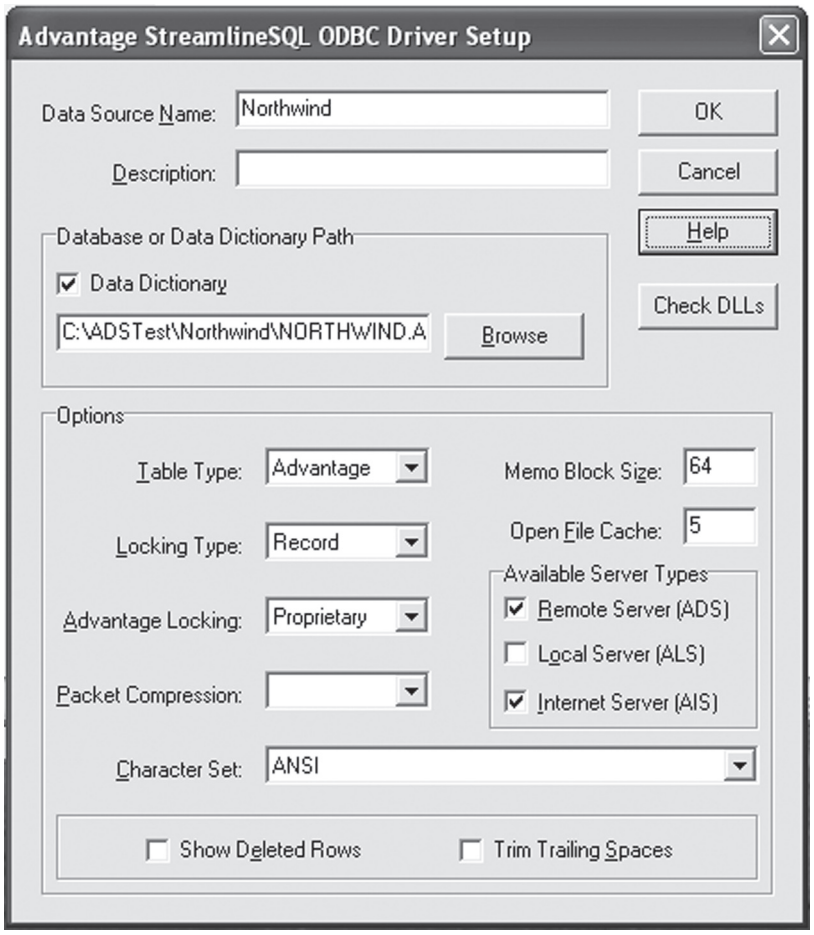

Abbildung 5. Sie können mit dem ADS ODBC-Treiber einen DSN definieren.

Geben Sie einen Namen für die Datenquelle und den Pfad zum ADS-Data Dictionary an. Stellen Sie die Optionen Ihren Anforderungen entsprechend ein (für jetzt lassen Sie die Standardwerte) und klicken Sie auf OK.

Mit remoten Ansichten auf ADS zugreifen

Wie eine lokale Ansicht ist eine remote Ansicht einfach eine vordefinierte SQL SELECT-Anweisung, die in einem Datenbankcontainer gespeichert ist. Der Unterschied besteht darin, dass eine remote Ansicht auf die Daten via ODBC zugreift, nicht nativ.

Sie können eine remote Ansicht wahlweise programmgesteuert mit dem Befehl CREATE SQL VIEW oder visuell mit dem Ansichten-Designer erstellen. In beiden Fällen müssen Sie eine ODBC-Verbindung angeben, die verwendet werden soll. Die Verbindung kann entweder ein ODBC DSN sein, der auf Ihrem System eingerichtet ist, oder ein Verbindungsobjekt, und erstellt eine remote Ansicht auf die Tabelle Customers der upgesizeden Datenbank Northwind. Der folgende Code ist ein Auszug des Codes, der von GENDBC generiert wurde; es wurde noch erheblich mehr Code generiert, der die unterschiedlichen Eigenschaften der Verbindung, der Ansicht und der Felder einstellt.

```
CREATE CONNECTION
NORTHWINDCONNECTION ;
   CONNSTRING "DSN=Northwind"
CREATE SQL VIEW "CUSTOMERSVIEW"
;
   REMOTE CONNECT
"NorthwindConnection" ;
   AS SELECT * FROM CUSTOMERS
Customers
DBSetProp('CUSTOMERSVIEW',
'View', ;
    'UpdateType', 1)
DBSetProp('CUSTOMERSVIEW',
'View', ;
    'WhereType', 3)
DBSetProp('CUSTOMERSVIEW',
'View', ;
    'FetchMemo', .T.)
DBSetProp('CUSTOMERSVIEW',
'View', ;
    'SendUpdates', .T.)
DBSetProp('CUSTOMERSVIEW',
'View', ;
    'Tables', 'CUSTOMERS')
DBSetProp('CUSTOMERSVIEW.custo-
merid', ;
    'Field', 'KeyField', .T.)
DBSetProp('CUSTOMERSVIEW.custo-
merid', ;
    'Field', 'Updatable', .F.)
DBSetProp('CUSTOMERSVIEW.custo-
merid', ;
    'Field', 'UpdateName', ;
    'CUSTOMERS.CUSTOMERID')
DBSetProp('CUSTOMERSVIEW.companyname', ;
    'Field', 'Updatable', .T.)
DBSetProp('CUSTOMERSVIEW.companyname', ;
    'Field', 'UpdateName', ;
    'CUSTOMERS.COMPANYNAME')
```

Eine der einfachsten Möglichkeiten, eine vorhandene Anwendung upzusizen, ist die Verwendung des ADS Upsizing Utility, um aus Ihrer VFP-Datenbank eine ADS-Datenbank zu erstellen. Anschließend erstellen Sie eine neue VFP-Datenbank (beispielsweise REMOTE.DBC) und legen Sie in dieser Datenbank remote Ansichten mit den gleichen Namen wie die Tabellen an, auf denen diese Ansichten basieren. Auf diese Weise ist der Code für das Öffnen einer remoten Ansicht exakt der gleiche wie für das Öffnen einer lokalen Tabelle, mit der Ausnahme, dass Sie zunächst eine andere Datenbank öffnen. Wenn Sie als Beispiel über ein Anwendungsobjekt namens oApp verfügen, das die Eigenschaft lUseLocalData enthält, um anzuzeigen, ob lokale oder remote Daten verwendet werden, öffnet dieser Code die entsprechende Datenbank und öffnet anschließend entweder die Tabelle Customers oder die remote Ansicht Customers:

```
if oApp.lUseLocalData
  open database Local
else
  open database Remote
endif
use Customers
```

Wenn Sie in der DataEnvironment von Formularen oder Berichten Cursorobjekte verwenden, müssen Sie noch einige zusätzliche Arbeit erledigen, da diese Objekte über eine Referenz auf die Datenbank verfügen, die Sie gewählt haben, als Sie die Ansichten in die DataEnvironment gezogen haben. Um dieses Problem zu lösen, fügen Sie in die Methode BeforeOpenTables der DataEnvironment einen Code wie den folgenden ein:

```
local loObject
for each loObject in This.Objects
  if upper(loObject.BaseClass)='CURSOR';
    and not empty(loObject.Database)
    loObject.Database = ;
      iif(oApp.lUseLocalData, ;
      'local.dbc', 'remote.dbc')
  endif
next
```

Vorteile

Hier die Vorteile der remoten Ansichten:

- Sie können den Ansichten-Designer verwenden, um eine remote Ansicht visuell zu erstellen. Es ist vorteilhaft, alle Felder der unter der Ansicht liegenden Tabelle zu sehen, die unterschiedlichen Teile der SQL SELECT-Anweisung in einer übersichtlichen Oberfläche einzurichten und Eigenschaften mit Checkboxen oder anderen Elementen der Benutzeroberfläche einzustellen.
- Aus der Sicht der Sprache funktionieren remote Ansichten genau wie Tabellen. Im Ergebnis können sie überall verwendet werden: Sie können sie mit USE öffnen, sie der DataEnvironment eines Formulars oder Berichts hinzufügen, an einen Grid binden, in einer SCAN-Schleife abarbeiten usw.
- Es ist einfacher, eine vorhandene Anwendung für die Verwendung remoter Ansichten umzustellen, besonders, wenn sie bereits lokale Ansichten verwendet, als wenn Sie andere Techniken einsetzen, die wir weiter hinten in diesem Artikel behandeln.
- Da Sie der DataEnvironment eines Formulars oder Berichts eine remote Ansicht hinzufügen können, können Sie die Vorteile der visuellen Oberfläche nutzen, die die DE anbietet: das Hineinziehen von Feldern oder des gesamten Cursors, um die Steuerelemente automatisch erstellen zu lassen, das einfache Binden eines Steuerelements an ein Feld, indem es in einer Combobox im Eigenschaften-Fenster gewählt wird usw. Außerdem kann VFP in Abhängigkeit von den Einstellungen in den Eigenschaften AutoOpenTables und OpenViews die remoten Ansichten für Sie automatisch öffnen.
- Es ist einfach, Änderungen an das Backend zu senden: sind die Eigenschaften der Ansicht korrekt eingestellt, rufen Sie einfach TABLEUPDATE() auf. Transaktionsverarbeitung und die Feststellung von Updatekonflikten sind bereits eingebaut.
- Remote Ansichten sind in einer Entwicklungsumgebung einfach zu verwenden: rufen Sie einfach USE und anschließend BROWSE auf.

Nachteile

Hier die Nachteile der remoten Ansichten:

- Remote Ansichten befinden sich in einem DBC, so dass Sie zusätzliche Dateien warten und auf dem System des Kunden installieren müssen.
- Da die SQL SELECT-Anweisung einer remoten Ansicht vordefiniert ist, können Sie sie nicht im laufenden Betrieb ändern. Obwohl dieses Verhalten für ein typisches Dateneingabeformular gut geeignet ist, kann es in Abfragen und Berichten ein Problem darstellen. Es ist möglich, dass Sie mehrere Ansichten mit den gleichen Daten erstellen, die sich lediglich in den gewählten Feldern, der Struktur der Klausel WHERE usw. unterscheiden.
- Sie können von einer remoten Ansicht aus keine gespeicherte Prozedur aufrufen, so dass eine remote Ansicht direkt auf die unter ihr liegenden Tabellen zugreifen muss.
- Wenn Sie TABLEUPDATE() verwenden, um die Änderungen in der Ansicht auf die Backend-Datenbank zu übertragen, haben Sie wenige Möglichkeiten (außer dem Einstellen einiger Eigenschaften), zu kontrollieren, wie VFP das Update erledigt.
- Wie auch bei den lokalen Ansichten wird eine remote Ansicht ungültig, wenn Sie mit SELECT * alle Felder einer angegebenen Tabelle abfragen und sich die Struktur der Tabelle im Backend ändert. In einem solchen Fall muss die Ansicht neu erstellt werden.
- Wenn Sie eine Ansicht öffnen, versucht VFP, die Datensätze der Ansicht im DBC zu sperren, wenn auch nur kurz. Das kann in viel verwendeten Anwendungen Probleme hervorrufen, wenn mehrere Anwender gleichzeig versuchen, ein Formular zu öffnen. Es gibt Workarounds für diese Situation (kopieren Sie den DBC auf die lokale Workstation und verwenden Sie diese Kopie oder, in VFP 7 und später, rufen Sie SET REPROCESS SYSTEM auf, um

das Timeout für die Sperre zu erhöhen), aber Sie müssen diesen Fall einplanen.

- Die Verbindungsinformation, die für eine remote Ansicht verwendet wird, ist im DBC als reiner Text hartcodiert. Das bedeutet, dass ein Hacker auf einfache Weise in Ihr Backend kommen kann und dafür nichts weiter benötigt als Notepad, um den DBC zu öffnen. Seit VFP 7 ist das kein Thema mehr, da Sie jetzt die Möglichkeit haben, einen Verbindungsstring anzugeben, wenn Sie eine remote Ansicht mit dem Befehl USE öffnen, was bedeutet, dass Sie unmittelbar vor dem Öffnen der Ansicht den Pfad zur Datenbank, den Benutzernamen und das Passwort dynamisch zusammensetzen können, in der Regel aus verschlüsselten Informationen.

Im Grunde handelt es sich um ein Kontrollproblem: remote Ansichten erleichtern die Arbeit mit den Backend-Daten, der Preis dafür ist die Einschränkung der Kontrolle, die Sie über den Vorgang haben.

Auf ADS mit SQL Passthrough zugreifen

VFP enthält verschiedene Funktionen, manchmal als SQL Passthrough (oder SPT)-Funktionen bezeichnet, die Ihnen den Zugriff auf eine Backend-Datenbank ermöglichen. SQLCONNECT() und SQLSTRINGCONNECT() erstellen eine Verbindung zur Backend-Datenbankengine. Der Unterschied zwischen diesen beiden Funktionen besteht darin, dass SQLCONNECT() einen vorhandenen ODBC DSN erfordert, während SQLSTRINGCONNECT() einen Verbindungsstring verwendet. SQLDISCONNECT() löst die Verbindung mit dem Backend. SQLEXEC() schickt einen Befehl, beispielsweise eine SQL SELECT-Anweisung, an die Datenbankengine und schreibt in der Regel (nicht zwingend, sondern in Abhängigkeit zum Befehl) die zurückgegebenen Ergebnisse in einen VFP-Cursor.

Hier ein Beispiel, das sich mit der upgesizeten Datenbank Northwind verbindet, alle Kunden empfängt und die Verbindung wieder löst (das Beispiel geht davon aus, dass ein DSN namens Northwind vorhanden ist, der definiert, wie die Verbindung zu dieser Datenbank aufgebaut wird).

```
lnHandle = sqlconnect('Northwind')
sqlexec(lnHandle, ;
    'select * from Customers')
browse
sqldisconnect(lnHandle)
```

Um stattdessen eine DSNlose Verbindung zu verwenden ersetzen Sie die Anweisung SQLCONNECT() durch den folgenden Code:

```
lcConnString = 'driver=Advantage StreamlineSQL
ODBC;' + ;
  'DataDirectory=C:\ADSTest\Northwind\
Northwind.add;'
lnHandle = sqlstringconnect(lcConnString)
```

Unabhängig davon, ob Sie für die Verbindung zu ADS einen DSN oder einen Verbindungsstring einsetzen, verwenden Sie die gleichen SQL-Anweisungen, die Sie auch verwenden würden, wenn Sie auf Datensätze in VFP-Tabellen zugreifen wollten, um diese zu aktualisieren oder zu löschen. Allerdings verwenden Sie die Funktion SQLEXEC(), um diese Anweisungen auszuführen. Zusätzlich zu den Funktionen der DML (Data Manipulation Language) wie SELECT, INSERT, UPDATE und DELETE unterstützt der ODBC-Treiber außerdem die Funktionen der DDL (Data Definition Language), beispielsweise CREATE DATABASE | TABLE | INDEX | VIEW | PROCEDURE, DROP INDEX | TABLE | VIEW | PROCEDURE und ALTER TABLE.

Auch wenn ADS den Großteil der Datentypen von VFP unterstützt, beachten Sie, dass sich die Angabe von Werten für Felder vom Typ Logical, Date und DateTime etwas von der VFP-Syntax unterscheidet. Werte vom Typ Logical werden von ADS als logische Felder an VFP zurückgegeben, aber Sie müssen Sie mit True oder 1 für True und False oder 0 für False angeben. Als Beispiel werden in folgen-

Tabelle 1 listet die Schlüsselworte auf, die Sie in einem Verbindungsstring angeben können. Die meisten dieser Schlüsselworte haben Äquivalente im ODBC-Dialog, den Sie in Abbildung 1 sehen. Alle Angaben mit Ausnahme von DataDirectory sind optional.

Schlüsselwort	Beschreibung
DataDirectory	Gibt den Pfad und Namen der ADD-Datei an. Für freie Tabellen geben Sie das Verzeichnis an, das die Dateien enthält.
DefaultType	Für freie Tabellen geben Sie „FoxPro“ für VFP-Tabellen an oder „Advantage“ für ADS-Tabellen (ADT-Dateien). Diese Einstellung wird für Datenbanken ignoriert.
ServerTypes	Gibt einen numerischen Wert an, der die Summe der Typen der ADS-Server darstellt, mit denen Sie sich verbinden wollen: Remote (2), Local (1) oder Internet (4). Als Beispiel verwenden Sie 3 (2+1) für Remote und Lokal.
AdvantageLocking	„ON“ (Standardwert) für die proprietäre Sperre von ADS oder „OFF“ für VFP-kompatible Sperre.
Locking	„Record“ (Standardwert) für eine Sperre auf Datensatzebene oder „File“ für die Sperre der gesamten Datei während der Updates.

Schlüsselwort	Beschreibung
Rows	Entspricht SET DELETED. „True“ und zeigt auch die gelöschten Datensätze an, während „False“ (Standardwert) sie nicht anzeigt. Diese Einstellung wird bei der Verwendung von ADS-Tabellen ignoriert, da gelöschte Datensätze in diesem Fall niemals sichtbar sind.
TrimTrailingSpaces	Entspricht den Feldern vom Typ Varchar. „True“ entfernt die abschließenden Leerzeichen aus Feldern vom Typ Character, die an die Anwendung zurückgegeben werden, „False“ (Standardwert) entfernt die Leerzeichen nicht.
MemoBlockSize	Entspricht dem Befehl SET BLOCKSIZE. Geben Sie die Blockgröße von Memofeldern für neue Tabellen an. Standardwert ist 64 für VFP-Tabellen und 8 für ADS-Tabellen.
CharSet	Die Zeicheneinstellung, die verwendet werden soll: „ANSI“ (Standardwert) oder „OEM“. Wenn Sie OEM verwenden, müssen Sie auch die Spracheinstellung angeben.
Language	Die Sprache, die verwendet wird, wenn CharSet=OEM.
MaxTableCloseCache	Die Anzahl der Cursor im Cache von ADS, Standardwert ist 5.
Compression	Die zu verwendende Kompressionsmethode. Lesen Sie die Hilfedatei von ADS, in der die verfügbaren Kompressionsmethoden beschrieben sind.
CommType	Das zu verwendende Kommunikationsprotokoll. Details entnehmen Sie bitte der Hilfedatei von ADS.

dem Beispiel nur die letzten beiden Anweisungen erfolgreich abgeschlossen:

```
sqlexec(lnHandle, "select * from " + ;
       "Products where Discontinued")
sqlexec(lnHandle, "select * from " + ;
       "Products where Discontinued=.T.")
sqlexec(lnHandle, "select * from " + ;
       "Products where Discontinued=True")
sqlexec(lnHandle, "select * from " + ;
       "Products where Discontinued=1")
```

Sybase iAnywhere hat signalisiert, dass in der endgültigen Version auch die Syntax .T. und .F. unterstützt wird.

Werte vom Typ Date und DateTime müssen in der Standard-ODBC-Syntax angegeben werden: {d'YYYY-MM-D'} für Date und {ts'YYYY-MM-DD HH:MM:SS'} für DateTime. Hier ein Beispiel:

```
sqlexec(lnHandle, "select * from " + ;
    "orders where OrderDate between " + ;
    "{d '1997-07-01'} and " + ;
    "{d '1997-07-31'}")
```

Hier die Funktion VFP2ODBCDate, die VFP-Werte vom Typ Date und DateTime in die ODBC-Syntax umwandelt:

```
lparameters tuDate
local lcDate, ;
  lcReturn
lcDate = transform(year(tuDate)) + ;
  '-' + padl(month(tuDate), 2, '0') + ;
  '-' + padl(day(tuDate), 2, '0')
if vartype(tuDate) = 'D'
  lcReturn = "{d '" + lcDate + "'}"
else
  lcReturn = "{t '" + lcDate + ;
    ' ' + padl(hour(tuDate), 2, '0') +;
    ':' + padl(minute(tuDate), 2, '0')+;
    ':' + padl(sec(tuDate), 2, '0') + ;
    "'}"
endif vartype(tuDate) = 'D'
return lcReturn
```

Das folgende Beispiel verwendet diese Funktion:

```
ldFrom = {^1997-07-01}
ldTo   = {^1997-07-31}
sqlexec(lnHandle, ;
   "select * from orders " + ;
   "where OrderDate between " + ;
   VFP2ODBCDate(ldFrom) + " and " + ;
   VFP2ODBCDate(ldTo))
```

Statt VFP-Werte in die ODBC-Syntax umzuwandeln, können Sie eine parametrisierte Abfrage verwenden und den hartcodierten Wert durch eine Variable mit dem Präfix „?“ ersetzen (die Variable muss sich selbstverständlich im Wirkungsbereich befinden). In diesem Fall erledigt VFP die Datentypumwandlung für Sie. Dieses Vorgehen hat noch den Vorteil, dass es SQL Injection-Angriffe (deren Beschreibung den Rahmen dieses Artikels sprengen würde) verhindert. Ein Beispiel:

```
ldFrom = {^1997-07-01}
ldTo   = {^1997-07-31}
sqlexec(lnHandle, 'select * from ' + ;
  'orders where OrderDate between ' + ;
  '?ldFrom and ?ldTo')
```

Obwohl VFP als Trennzeichen für Strings einfache Anführungszeichen, doppelte Anführungszeichen und eckige Klammern verwenden kann, übergeben Sie Stringwerte an SPT-Funktionen immer mit einfachen Anführungszeichen.

Auch wenn dieses Beispiel dies nicht tut, achten Sie darauf, dass Sie immer den Rückgabewert von SQLEXEC() prüfen. Wird ein anderer Wert als 1 zurückgegeben, ist der Befehl fehlgeschlagen und Sie sollten AERROR() verwenden, um festzustellen, was schief gelaufen ist. Außerdem können Sie als dritten Parameter von SQLEXEC() den Namen des Cursors angeben, der erstellt werden soll; wenn Sie diesen Parameter nicht angeben, wird der Cursor mit SQLResult benannt.

Vorteile

Die Vorteile der Verwendung von SPT sind:

- Sie haben eine erheblich höhere Flexibilität beim Datenzugriff als mit remoten Ansichten, beispielsweise können Sie mit der Funktion gespeicherte Prozeduren aufrufen.
- Falls erforderlich können Sie die Verbindungsinformation im laufenden Betrieb ändern. Als Beispiel können Sie den Benutzernamen und das Passwort als verschlüsselte Werte speichern und sie erst entschlüsseln, wenn Sie sie in den Funktionen SQLCONNECT() oder SQLSTRINGCONNECT() benötigen. Wie ich bereits ausgeführt habe, ist dies nicht der Vorteil gegenüber der Verwendung remoter Ansichten, da Sie seit VFP 7 im Befehl USE den Verbindungsstring angeben können.
- Falls erforderlich können Sie die SQL SELECT-Anweisung ändern. Als Beispiel können Sie auf einfache Weise die Feldliste, die Klausel WHERE, die Tabellen usw. ändern.
- Für die Verwendung von SPT benötigen Sie keinen DBC, so dass Sie ihn nicht warten oder installieren müssen, die Sperrmechanismen stellen kein Problem dar und Sie brauchen sich keine Sorgen machen, dass eine Anweisung SELECT * ungültig wird, wenn sich die Struktur der Backend-Tabellen ändert.
- Wie bei den remoten Ansichten ist das Ergebnis eines SPT-Aufrufs ein VFP-Cursor, den Sie an beliebiger Stelle in VFP verwenden können.
- Da Sie alles selbst codieren müssen (das führe ich detaillierter unter den Nachteilen aus), haben Sie eine bessere Kontrolle über die Arbeitsweise der Updates. Als Beispiel können Sie die Anweisung SQL SELECT verwenden, um den Cursor zu erstellen, aber eine gespeicherte Prozedur aufrufen, um die ADS-Tabellen zu aktualisieren.
- Sie können Ihre eigenen Verbindungen verwalten. Als Beispiel könnten Sie ein Verbindungsmanagerobjekt verwenden, um an einer Stelle alle Verbindungen zu verwalten, die Ihre Anwendung verwendet.

Nachteile

Hier die Nachteile der Verwendung von SPT:

- SPT erfordert zusätzliche Arbeit, da Sie alles selbst codieren müssen: das Erstellen und Schließen der Verbindung, die SQL SELECT-Anweisungen, die ausgeführt werden sollen usw. Sie können auf kein visuelles Werkzeug wie den Ansichten-Designer zurückgreifen, das Ihnen zeigt, welche Felder in welchen Tabellen vorhanden sind.
- Sie können einer DataEnvironment eines Formulars oder eines Berichts visuell keinen Cursor hinzufügen, den Sie für SPT erstellt haben. Stattdessen müssen Sie das Öffnen des Cursors (beispielsweise in der Methode BeforeOpenTables) codieren, Sie müssen die Steuerelemente manuell erstellen und Sie müssen die Eigenschaften für die Bindung (beispielsweise ControlSource) selbst ausfüllen, indem Sie die Werte eingeben. Machen Sie keinen Tippfehler, wenn Sie die Alias- und Feldnamen eingeben, da Ihr Formular andernfalls nicht funktioniert.
- Sie sind in einer Entwicklungsumgebung schwieriger zu verwenden als remote Ansichten: statt einfach den Befehl USE auszuführen müssen Sie eine Verbindung erstellen und anschließend SQLEXEC() aufrufen, um die Daten zu erhalten, die Sie ansehen wollen. Sie können sich die Arbeit erleichtern, indem Sie verschiedene PRGs erstellen, die die Arbeit für Sie erledigen oder Sie können den Data Explorer, der mit VFP ausgeliefert wird, nutzen, um die Strukturen und Inhalte der Tabellen zu erkunden. Sie können auch einen DBC und verschiedene remote Ansichten erstellen, die nur in der Entwicklungsumgebung verwendet werden, um einen schnellen Blick auf die Daten werfen zu können.
- Cursor, die mit SPT erstellt wurden, können änderbar sein, aber auch das müssen Sie selbst mit einer Reihe Aufrufen von CURSORSETPROP() für die Eigenschaften SendUpdates, Tables, KeyFieldList, UpdatableFieldList und UpdateNameList einstellen. Außerdem müssen Sie die Transaktionsverarbeitung und Konfliktfeststellung selbst erledigen.
- Obwohl SPT-Cursor nicht wie remote Ansichten definiert werden, können Sie mit SPT auf einfache Weise zwischen lokalen und remoten Daten umschalten, indem Sie einfach die Ansicht ändern, die in einem Formular oder Bericht geöffnet wird.

Auf ADS mit ADO zugreifen

OLE DB Provider sind ODBC-Treibern ähnlich: sie stellen eine standardisierte und konsistente Möglichkeit für den Zugriff auf Datenquellen dar. Da OLE DB aus verschiedenen Low Level COM-Schnittstellen besteht, ist es nicht einfach, damit in Sprachen wie VFP zu arbeiten. Um dieses Problem zu bewältigen hat Microsoft ActiveX-Datenobjekte (ADO) erstellt, verschiedene COM-Objekte, die eine objektorientierte Oberfläche für OLE DB bilden.

ADO besteht aus verschiedenen Objekten. Dazu gehören:

- Connection: Dieses Objekt ist für die Kommunikation mit der Datenquelle verantwortlich.

- Recordset: Dies ist das Äquivalent zu einem VFP-Cursor: das Recordset verfügt über eine definierte Struktur, enthält die Daten sowie Eigenschaften und Methoden für das Hinzufügen, Löschen und Aktualisieren von Datensätzen, die Navigation in den Datensätzen, das Filtern und Sortieren der Daten und für das Aktualisieren der Datenquelle.
- Command: Dieses Objekt dient der Ausführung umfangreicherer Abfragen als einer einfachen SELECT-Anweisung, beispielsweise von parametrisierten Abfragen und dem Aufruf gespeicherter Prozeduren.

Hier ein Beispiel (ADOExample.PRG, das alle brasilianischen Kunden aus der upgesizeden Datenbank Northwind empfängt und die Kunden-ID sowie den Firmennamen anzeigt. Beachten Sie, dass das Verbindungsobjekt die Verbindung behandelt, während das Recordset mit den Daten umgeht. Dieser Code referenziert ADOVFP.H, eine Include-Datei mit Konstanten, die bei der Arbeit mit ADO hilfreich sind.

```
#include ADOVFP.H
local loConn as ADODB.Connection, ;
  loRS as ADODB.Recordset, ;
  lcCustomers

* Connect to the ADS database.

loConn = createobject('ADODB.Connection')
loConn.ConnectionString = ,
    'provider=Advantage.OLEDB.1;' + ;
    'data source=c:\adstest\northwind\'+;
    'northwind.add'
loConn.Open()

* Create a Recordset and
* set its properties.

loRS = createobject('ADODB.Recordset')
loRS.ActiveConnection = loConn
loRS.LockType = 3  && adLockOptimistic
loRS.CursorLocation = 3  && adUseClient
loRS.CursorType = 3  && adOpenStatic

* Execute a query and display
* the results.

loRS.Open("select * from customers " + ;
    "where country='Brazil'")
lcCustomers = ''
do while not loRS.EOF
  lcCustomers = lcCustomers + ;
    loRS.Fields('customerid').Value + ;
    chr(9) + ;
    loRS.Fields('companyname').Value + ;
    chr(13)
  loRS.MoveNext()
enddo while not loRS.EOF
messagebox(lcCustomers)
loRS.Close()
loConn.Close()
```

Beachten Sie, wie dieses Programm objektorientierten Code für den Zugriff auf das Recordset verwendet. Die Eigenschaft EOF ist das Äquivalent zu VFPs Funktion EOF() und die Methode MoveNext ähnelt SKIP. Um auf den Wert eines Feldes im aktuellen Datensatz zuzugreifen, verwenden Sie Recordset.Fields('FieldName').Value.

Die Verwendung parametrisierter Abfragen mit ADO erfordert etwas mehr Arbeit als der Einsatz von ODBC. Zusätzlich zur Angabe eines Parameters als „?" (ohne den Variablennamen) müssen Sie ADO-Command- und Parameterobjekte verwenden, um den Parameter und dessen Wert anzugeben.

```
* Connect to the ADS database.

loConn = createobject('ADODB.Connection')
loConn.ConnectionString = + ;
    'provider=Advantage.OLEDB.1;' + ;
    'data source=c:\adstest\' + ;
    'northwind\northwind.add'
loConn.Open()

* Create a Command object and
* define the command type and connection.

loCommand = createobject('ADODB.Command')
loCommand.CommandType      = adCmdText

loCommand.ActiveConnection = loConn
* Create a Parameter object,
* set its properties, and
* add it to the Command object.

loParameter = loCommand.
CreateParameter('Country', adChar, ;
   adParamInput, 15)
loParameter.Value = 'UK'
loCommand.Parameters.Append(loParameter)

* Execute a parameterized query and
* display the results.

loCommand.CommandText = 'select * ' + ;
    'from customers where country = ?'
loRS = loCommand.Execute()
* same code as above to
* display the results
```

Vorteile

Die Vorteile der Verwendung von ADO sind:

- Viele der Vorteile sind die gleichen wie mit SPT: Sie können flexibler als mit remoten Ansichten auf die Daten zugreifen, Sie können falls erforderlich die Verbindungsinformation im laufenden Betrieb ändern, Sie können falls erforderlich die SQL SELECT-Anweisung ändern, Sie können Ihre eigenen Verbindungen verwalten und dabei ist kein DBC beteiligt.

- Obwohl die Geschwindigkeitsunterschiede in einfachen Szenarien nicht signifikant sind (grundsätzlich ist ODBC schneller als ADO), ist ADO in viel verwendeten Anwendungen wie Webservern besser skalierbar.
- ADO ist objektorientiert, so dass Sie mit den Daten wie mit Objekten umgehen können.
- Abhängig von ihrer Einrichtung sind ADO Recordsets automatisch aktualisierbar, ohne dass außer dem Aufruf der Methoden Update und UpdateBatch zusätzliche Arbeit erforderlich wäre. Transaktionsverarbeitung und Updatekonflikterkennung sind eingebaut.
- Sie können einen Recordset auf einfache Weise in einer lokalen Datei speichern und später erneut laden, mit der Arbeit fortfahren und am Ende die ADS-Datenquelle aktualisieren. Daher ist ADO die bessere Wahl als remote Ansichten oder SPT für Anwendungen, die häufig unterwegs eingesetzt werden.

Nachteile

Hier die Nachteile von ADO:

- ADO bedeutet zusätzliche Arbeit, da Sie alles selbst codieren müssen: das Erstellen und Schließen der Verbindung, die SQL SELECT-Anweisungen, die ausgeführt werden sollen usw. Sie können auf kein visuelles Werkzeug wie den Ansichten-Designer zurückgreifen, das Ihnen zeigt, welche Felder in welchen Tabellen vorhanden sind.
- Ein ADO Recordset ist kein VFP-Cursor, so dass Sie es nicht an Stellen verwenden können die einen Cursor erfordern, beispielsweise in Grids und Berichten. Das Hilfsprogramm VFPCOM (das Sie von der VFP-Homepage http://msdn.microsoft.com/vfoxpro herunterladen können) enthält Funktionen, die ein Recordset in einen Cursor und zurück umwandeln können, was aber die Performance einschränken kann, besonders bei großen Datenmengen. Außerdem gibt es mit einigen Datentypen Probleme. Wenn Sie ADO verwenden wollen, ist der CursorAdapter (der als nächstes behandelt wird) das Werkzeug der Wahl.
- ADO Recordsets werden nicht visuell unterstützt, weshalb Sie den gesamten Code für das Erstellen und Öffnen der Recordsets selbst erstellen müssen. Sie müssen die Steuerelemente manuell erstellen und Sie müssen die Eigenschaften für die Bindung (beispielsweise ControlSource) selbst ausfüllen, indem Sie die Werte manuell eintragen. Das bedeutet noch mehr Arbeit als für SPT, da die Syntax nicht einfach aus CURSOR.FIELD besteht – sie lautet Recordset.Fields('FieldName').Value.
- Recordsets sind die Technik, die sich in einer Entwicklungsumgebung am schwierigsten verwenden lässt, da Sie alles selbst codieren müssen: Sie müssen eine Verbindung erstellen, die Daten empfangen und sich zwischen den Datensätzen bewegen. Sie können kein BROWSE verwenden, um sich anzeigen zu lassen, wie die Ergebnismenge aussieht (es sei denn, Sie verwenden VFPCOM oder CursorAdapter, um das Recordset in einen Cursor umzuwandeln.
- Die Verwendung von ADO erfordert eine höhere Lernkurve als die Verwendung von Cursorn, die mit ODBC erzeugt wurden.

Mit CursorAdapter auf ADS zugreifen

Sie haben vermutlich bemerkt, dass sich jeder der hier beschriebenen Mechanismen grundlegend von den anderen unterscheidet. Das bedeutet für Sie eine neue Lernkurve für jede Technik und die Umstellung einer vorhandenen Anwendung von einem Mechanismus auf einen anderen ist keine einfache Aufgabe.

Zum Glück enthält VFP eine Technologie, die für ODBC und OLE DB eine gemeinsame Oberfläche bereitstellt: die Klasse CursorAdapter. Aus folgenden Gründen ist der CursorAdapter, der mit VFP 8 eingeführt wurde, eine großartige Lösung:

- Er vereinfacht die Verwendung von ODBC, ADO und XML, auch wenn Sie mit diesen Technologien nicht wirklich vertraut sind.
- Er stellt unabhängig vom von Ihnen gewählten Mechanismus eine konsistente Oberfläche für den Zugriff auf die remoten Daten dar.
- Er macht es einfach, vorhandene Anwendungen von einem Mechanismus auf einen anderen umzustellen.

Hierfür möchte ich Ihnen ein Beispiel geben. Stellen Sie sich vor, Sie hätten eine Anwendung, die ODBC mit CursorAdapter verwenden, um auf ADS-Daten zuzugreifen und aus irgendeinem Grund wollen Sie stattdessen ADO einsetzen. Sie müssen dafür lediglich die Eigenschaft DataSourceType der CursorAdapters und die Verbindung zur ADS-Datenbank ändern und damit sind Sie fertig. Die restlichen Komponenten der Anwendung wissen davon nichts und sehen weiterhin den gleichen Cursor, unabhängig davon, welcher Mechanismus verwendet wird, um auf die Daten zuzugreifen.

Hier ein Beispiel (CursorAdapterExample.PRG), das verschiedene Felder brasilianischer Kunden aus der Tabelle Customers in der Datenbank Northwind empfängt. Der Cursor kann geändert werden, so dass Sie, wenn Sie im Browsefenster Änderungen vornehmen, es schließen und das Programm erneut ausführen, feststellen werden, dass Ihre Änderungen gespeichert wurden.

```
local lcConnString, ;
  lnHandle, ;
  loCursor as CursorAdapter, ;
  laErrors[1]
close tables all

* Connect to ADS.

lcConnString = 'driver=Advantage ' + ;
   'StreamlineSQL ODBC;' + ;
   'DataDirectory=C:\ADSTest\' + ;
   'Northwind\Northwind.add;'
lnHandle     = ;
    sqlstringconnect(lcConnString)

* Create a CursorAdapter and
* set its properties.

loCursor = createobject('CursorAdapter')
with loCursor
   .Alias = 'Customers'
   .DataSourceType = 'ODBC'
   .DataSource = lnHandle
   .SelectCmd  = "select CUSTOMERID, " + ;
     " COMPANYNAME, CONTACTNAME " + ;
     "from CUSTOMERS where COUNTRY " + ;
     " = 'Brazil'"
   .KeyFieldList       = 'CUSTOMERID'
   .Tables             = 'CUSTOMERS'
   .UpdatableFieldList = ;
     'CUSTOMERID,COMPANYNAME, CONTACTNAME'
   .UpdateNameList     =
     'CUSTOMERID CUSTOMERS.CUSTOMERID,'+;
    'COMPANYNAME CUSTOMERS.COMPANYNAME,'+;
     'CONTACTNAME CUSTOMERS.CONTACTNAME'
   if .CursorFill()
     browse
   else
     aerror(laErrors)
     messagebox(laErrors[2])
   endif .CursorFill()
endwith
```

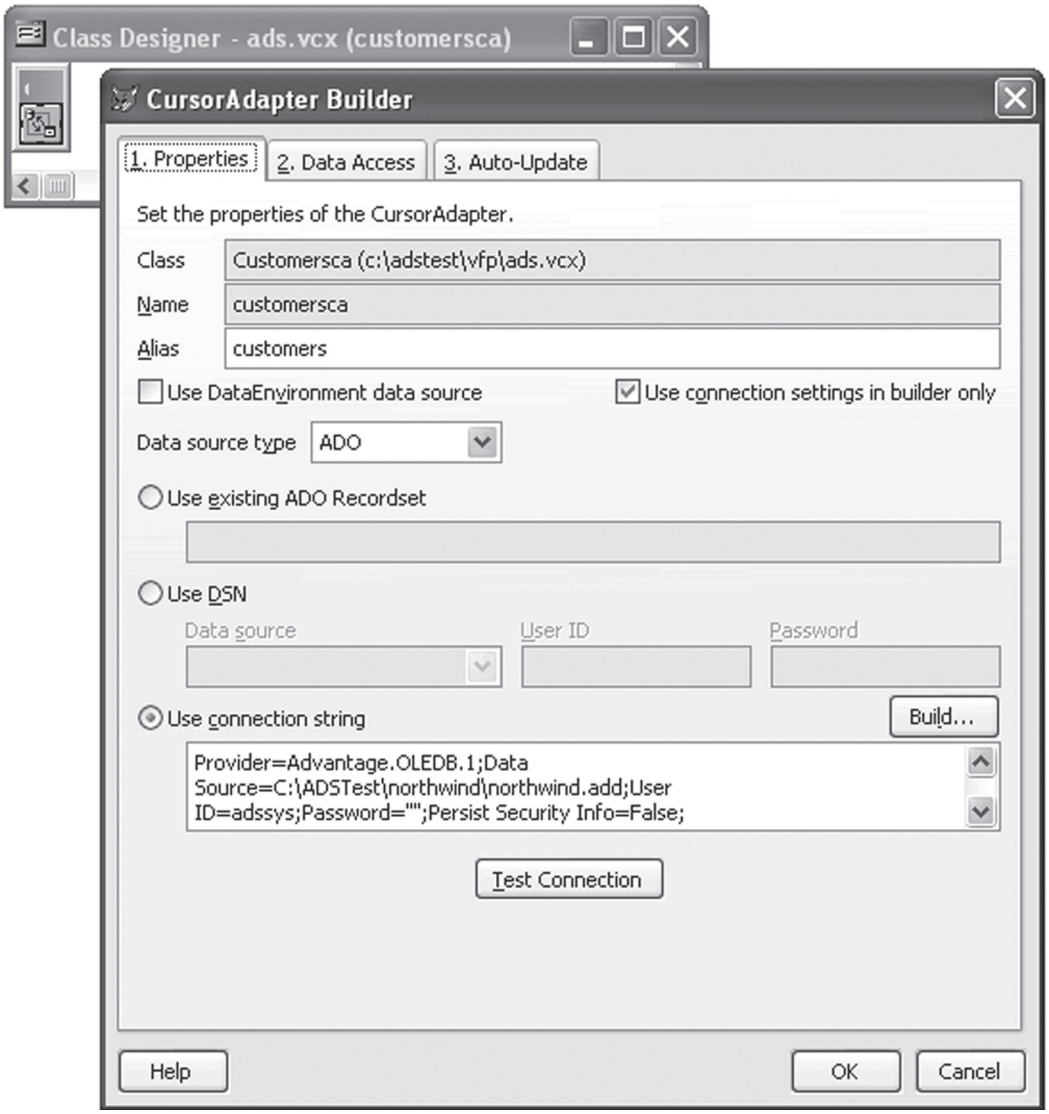

Abbildung 6. Der CursorAdapter-Generator ist ein visuelles Werkzeug, um Klassen von CursorAdapter abzuleiten.

Sie müssen einen CursorAdapter nicht im Code erzeugen. Für diese Aufgabe können Sie den Klassen-Designer verwenden, um von CursorAdapter eine Klasse abzuleiten und entweder im Eigenschaften-Fenster die Eigenschaften zu füllen oder Sie können den CursorAdapter-Generator verwenden, ein nettes Werkzeug für die Erstellung von CursorAdaptern (siehe Abbildung 6). Beachten Sie, dass der CursorAdapter-Generator nicht korrekt mit ADS zusammenarbeitet, wenn Sie ODBC verwenden, um sich mit der Datenbank zu verbinden. Details finden Sie im Abschnitt „Das ADS VFP Upsizing Utility fixen". Dort finden Sie auch Informationen, wie Sie das Problem beseitigen.

Hier die wichtigsten Eigenschaften und Methoden von CursorAdapter:

- DataSourceType: legt fest, wie auf die Daten zugegriffen wird. Sie haben die Wahl zwischen „Native", „XML", „ODBC" und „ADO", auch wenn nur die letzten beiden mit ADS verwendet werden.
- DataSource: der Wert dieser Eigenschaft ist davon abhängig, welchen Wert DataSourceType enthält. Im Fall von ODBC muss es ein geöffnetes ODBC-Handle sein. Für ADO handelt es sich um ein ADO Recordset, dessen ActiveConnection auf ein geöffnetes Verbindungsobjekt eingestellt ist. Beachten Sie in jedem Fall, dass Sie für das Öffnen und die Verwaltung der Verbindung selbst verantwortlich sind.
- SelectCmd: die SQL-Anweisung, die ausgeführt wird, um die Daten zu empfangen.
- Alias: der Alias des erzeugten Cursors.
- KeyFieldList, Tables, UpdatableFieldList und UpdateNameList: diese Felder sind der Schlüssel, um den Cursor änderbar zu gestalten. Stellen Sie die passenden Werte ein und CursorAdapter schreibt die Änderungen automatisch in die Datenquelle zurück. Details zu diesen Eigenschaften finden Sie in der Dokumentation von VFP.
- CursorFill: rufen Sie diese Methode auf, um den Cursor zu erstellen und die Anweisung in SelectCmd auszuführen, die den Cursor mit Daten füllt.
- CursorDetach: standardmäßig wird der von CursorAdapter erstellte Cursor dem CursorAdapter-Objekt „zugeordnet". Wenn der CursorAdapter freigegeben wird (dies geschieht, wenn er den Wirkungsbereich verlässt), wird der Cursor automatisch geschlossen. Soll der Cursor geöffnet bleiben, rufen Sie die Methode CursorDetach auf, um den Cursor vom CursorAdapter zu befreien.

Ein CursorAdapter kann wahlweise ODBC oder ADO verwenden, um sich mit ADS zu verbinden. Für ODBC öffnen Sie eine Verbindung zur Datenbank mit SQLCONNECT() oder SQLSTRINGCONNECT(), stellen Sie die Eigenschaft DataSource des CursorAdapters auf das Verbindungs-Handle ein und setzen Sie die Eigenschaft DataSourceType auf „ODBC". ADO erfordert etwas mehr Arbeit: Instanziieren und öffnen Sie ein ADO-Verbindungsobjekt, instanziieren Sie ein Recordset-Objekt, setzen Sie die Eigenschaft ActiveConnection des Recordsets auf das Verbindungsobjekt, setzen Sie die Eigenschaft DataSource des CursorAdapters auf das Recordset-Objekt und den DataSourceType auf „ADO". Für parametrisierte Abfragen nutzen Sie in Ihrer SQL-Anweisung die Syntax „?VariableName", auch für ADO. Für ADO müssen Sie noch ein ADO-Befehlsobjekt instanziieren und es als vierten Parameter an CursorFill übergeben (sorgen Sie sich nicht um das Parameterobjekt; darum kümmert sich VFP intern).

Statt diese Arbeit für ADO vollständig manuell zu erledigen können Sie die Klasse SFCursorAdapterADO in SFCursorAdapter.VCX verwenden, die Sie im Quellcode zu diesem Artikel finden und die diese Arbeit für Sie erledigt. Die Eigenschaft DataSourceType dieser Klasse ist auf „ADO" eingestellt und die Methode Init richtet die DataSource korrekt ein.

```
local loRS as ADODB.Recordset
loRS = createobject('ADODB.RecordSet')
loRS.CursorLocation = 3  && adUseClient
loRS.LockType = 3  && adLockOptimistic
This.DataSource = loRS
```

Nachdem Sie ein Verbindungsobjekt erstellt und geöffnet haben, übergeben Sie es an SetConnection.

```
lparameters toConnection
This.DataSource.ActiveConnection = ;
    toConnection
```

Sie müssen CursorFill kein Befehlsobjekt übergeben; SFCursorAdapter ADO verwendet automatisch ein Befehlsobjekt, wenn die SQL-Anweisung ein „?" enthält.

```
lparameters tlUseCursorSchema, ;
  tlNoData, ;
  tnOptions, ;
  toSource
local loSource as ADODB.Command, ;
  lnOptions, ;
  llUseCursorSchema, ;
  llNoData, ;
  llReturn, ;
  laError[1]

* If we have a parameterized query, we
* need an ADO Command object. Create one
* if it wasn't passed.

if '?' $ This.SelectCmd and
```

```
vartype(toSource) <> 'O'
  loSource = createobject(;
             'ADODB.Command')
  loSource.ActiveConnection = ;
       This.DataSource.ActiveConnection
  lnOptions = adCmdText
else
  loSource  = toSource
  lnOptions = tnOptions
endif '?' $ This.SelectCmd ...

* If the first two parameters weren't
* specified, we don't want to explicitly
* pass .F., so use the default values.
* If CursorSchema is empty, we'll, of
* course, pass .F. for the first
* parameter.

do case
  case pcount() >= 2
    llUseCursorSchema = tlUseCursorSchema
    llNoData = tlNoData
  case pcount() = 1
    llUseCursorSchema = tlUseCursorSchema
    llNoData = This.NoData
  case pcount() = 0
    llUseCursorSchema = ;
       This.UseCursorSchema
    llNoData = This.NoData
endcase
if empty(This.CursorSchema)
  llUseCursorSchema = .F.
endif empty(This.CursorSchema)
llReturn = dodefault(llUseCursorSchema,;
   llNoData, lnOptions, loSource) and ;
   used(This.Alias)

* If something went wrong, find out why.

if not llReturn
  aerror(laError)
  This.cErrorMessage = laError[2]
endif not llReturn
return llReturn
```

Here's an example that uses SFCursorAdapterADO, taken from ADOCursorAdapterExample.PRG:

```
local loConnection as ADODB.Connection, ;
   loCA as SFCursorAdapterADO of ;
   SFCursorAdapter.vcx
private pcCountry

* Create and open an ADO Connection
* object.

loConnection = createobject(;
   'ADODB.Connection')
loConnection.ConnectionString =
   'Provider=Advantage.OLEDB.1;' + ;
   'Data Source=C:\ADSTest\' + ;
   'northwind\northwind.add;' + ;
   'User ID=adssys;Password=""'
loConnection.Open()

* Create an SFCursorAdapterADO object
* and set it up.

loCA = newobject('SFCursorAdapterADO',;
   'SFCursorAdapter.vcx')
with loCA
  .SetConnection(loConnection)
  .SelectCmd = 'select * from ' + ;
   'customers where country = ?pcCountry'
  .Alias = 'customers'

* Do the query and either show
* the result set or an error.

  pcCountry = 'Germany'
  if .CursorFill()
    browse
  else
    messagebox(loCa.cErrorMessage)
  endif .CursorFill()
endwith

* Close the connection.

loConnection.Close()
```

Vorteile

Die Vorteile von CursorAdapter bestehen kurz gesagt aus der Kombination der Vorteile aller anderen Technologien.

- Je nachdem, wie er eingerichtet ist, kann das Öffnen eines Cursors von einer von CursorAdapter abgeleiteten Klasse so einfach sein wie das Öffnen einer remoten Ansicht: Sie instanziieren einfach die abgeleitete Klasse und rufen die Methode CursorFill auf. Sie können dies auch im Init ausführen, um daraus eine Operation mit nur einem Schritt zu machen.
- Es ist einfacher, eine vorhandene Anwendung auf die Verwendung von CursorAdapter umzustellen als auf die Verwendung von Cursorn, die mit SPT erstellt wurden.
- Wie remote Ansichten können Sie einen CursorAdapter auch der DataEnvironment eines Formulars oder Berichts hinzufügen und von der visuellen Unterstützung profitieren, die die IDE anbietet: Ziehen von Feldern, um automatisch Steuerelemente zu erstellen, das einfache Binden eines Steuerelements an ein Feld, indem es im Eigenschaften-Fenster aus einer Combobox ausgewählt wird usw.
- Es ist einfach, das Backend mit Änderungen zu aktualisieren: davon ausgehend, dass die Eigenschaften der Ansicht korrekt eingestellt sind, rufen Sie einfach TABLEUPDATE() auf.
- Da sich die Ergebnismenge, die durch einen CursorAdapter erstellt wird, in einem VFP-Cursor befindet, kann sie überall in VFP verwendet werden: in einem Grid, einem Bericht, er kann in einer SCAN-Schleife verarbeitet werden usw. Das gilt auch, wenn auf die Datenquelle via ADO und XML zugegriffen wird, da der CursorAdapter automatisch die Umwandlung für Sie ausführt.

- Sie sind beim Datenzugriff sehr flexibel, da Sie auch gespeicherte Prozeduren oder Objekte in der mittleren Schicht aufrufen können.
- Falls erforderlich können Sie die Verbindungsinformation oder die SQL SELECT-Anweisung im laufenden Betrieb ändern; sie benötigen keinen DBC und Sie können Ihre eigenen Verbindungen verwalten.
- Da Sie es selbst codieren, können haben Sie eine bessere Kontrolle über die Arbeitsweise der Updates. Als Beispiel könnten sie eine SQL SELECT-Anweisung verwenden, um den Cursor zu erstellen, aber eine gespeicherte Prozedur aufrufen, um die Änderungen an die Backend-Tabellen zu übertragen.

Nachteile

Die Verwendung von CursorAdptern hat nur wenige Nachteile:

- Sie können kein nettes visuelles Werkzeug wie den Ansichten-Designer verwenden, um CursorAdapter zu erstellen, obwohl der CursorAdapter-Generator dem schon recht nahe kommt.
- Wie bei allen neuen Technologien stehen Sie vor einer Lernkurve, die Sie meistern müssen.

Lizensierung

Advantage Database Server verwendet ein konkurrierendes Lizensierungsmodell; Sie benötigen für jeden verbundenen Anwender eine Lizenz. Jede Workstation kann eine unbegrenzte Anzahl Datenbankverbindungen verwenden. Mehrere Anwendungen, die auf einer einzelnen Workstation ausgeführt werden und die auf Advantage zugreifen, werden als ein einzelner Anwender lizensiert. Sybase iAnywhere veröffentlicht keine Preisliste für Lizenzen. Obwohl sie aussagen, dass sie ein flexibles Preismodell ausgearbeitet haben und dass ihr Preis mit dem anderer Datenbankserver vergleichbar ist, müssen Sie sich an Sybase iAnywhere wenden, um einen Preis zu erfahren.

Ressourcen

Die Advantage-Website http://devzone.advantagedatabase.com enthält zahlreiche Ressourcen, über die Sie mehr über ADS erfahren können, beispielsweise Newsgroups (einschließlich einer VFP-spezifischen Newsgroup), eine Onlinedokumentation, White Papers, Tutorials und Beispielcode. Außerdem bietet das Buch Advantage Database Server: A Developer's Guide (ISVN 978-1-4259-7726-9) von Cary Jensen und Loy Anderson einen hervorragenden Einstieg in ADS. Obwohl keine Beispiele in VFP angeboten werden, werden VFP-Entwickler keine Probleme haben, den Code zu verstehen und zu übersetzen.

Andrew MacNeill hat J. D. Mullin, R & D Manager für ADS, in The FoxShow #49, einem Podcast, der unter http://akselsoft.lobsyn.com/index.php?post_id=302994 zum Download bereitsteht, interviewt. Dieses Interview liefert einige Hintergrundinformationen zu ADS und VFP und behandelt einige Designfeatures von ADS.

Zusammenfassung

Advantage Database Server ist eine überraschende Datenbankengine, die VFP-Anwendungsentwickler besser unterstützt als jede andere Client-/Server-Datenbankengine. Die Software kann die Basis für eine Migrationsstrategie bilden, auf der aufbauend Sie Ihre Anwendungen von einem dateibasierten Datenzugriff zu einer echten Client-/Server-Technologie migrieren können.

Als dieser Artikel geschrieben wurde, war ADS Version 9 im Betastadium, so dass es möglich ist, dass sich die endgültige Version des Produkts anders verhält als in diesem Artikel beschrieben. Die grundlegenden Konzepte bleiben aber die gleichen, auch die Vorteile von ADS und der Datenzugriff von VFP-Anwendungen auf die Datenbankengine.

Das ADS VFP Upsizing Utility fixen

Obwohl es noch Änderungen geben wird, bis die endgültige Version erscheint, enthält das ADS VFP Upsizing Utility, ADSUpsize.PRG, das in der Betaversion enthalten ist, einige Probleme:

Autoinkrementelle Felder werden nicht korrekt behandelt.
Bei einigen Tabellen generierte die Methode AddDatabaseToTable einen Fehler.

- Es war nicht möglich, das Utility mehrfach auszuführen, ohne einige Aufräumarbeiten auszuführen.
- Es konnte den Fall nicht korrekt behandeln, dass es keine Verbindung zum ADS OLE DB Provider aufbauen konnte.
- Relationen zwischen Tabellen wurden wenig hilfreich mit Namen wie Relation_1 und Relation_2 benannt.
- Beim Upsizen von Ansichten werden die Unterschiede in der Syntax von VFP und ADS nicht behandelt, besonders bei den Filtern auf Daten vom Typ Logical, Date oder DateTime oder mit Referenzen auf den Datenbank-Container (also DatabaseName!TableName).
- Tabellen ohne Indizes lösten einen Fehler aus.
- Das Upsizing funktionierte nicht richtig, wenn Felder vom Typ Varchar oder Varbinary waren.

Zum Glück waren alle diese Fehler einfach zu beseitigen und der Quellcode für diesen Artikel enthält den Ersatz ADSUpsize.PRG, in dem diese Probleme beseitigt sind.

Den VFP CursorAdapter-Generator fixen

Der CursorAdapter-Generator hat ein Problem mit dem ADS ODBC-Treiber (mit dem ADS OLE DB Provider funktioniert er). Der Dialog Select Command Builder wird angezeigt, wenn Sie auf die Schaltfläche Build auf Seite 2 des CursorAdapter-Generators klicken. Dieser Dialog generiert einen Fehler mit dem ADS ODBC-Treiber, da der Treiber eine andere Ergebnismenge an die Funktionen SQLTABLES() und SQLCOLUMNS() liefert als es der SQL Server und viele andere Treiber tun. Statt die Felder mit TABLE_NAME und COLUMN_NAME zu benennen, verwendet ADS TABLENAME und COLUMNNAME.

Zum Glück stellt Microsoft uns den Quellcode des CursorAdapter-Generators zur Verfügung und es war einfach, den Fehler zu beseitigen. Der Quellcode zu diesem Artikel enthält die Ersetzung DEBuilder.APP, die diese Probleme beseitigt. Kopieren Sie diese Datei in das Verzeichnis Wizards unterhalb des Hauptverzeichnisses von VFP und überschreiben Sie die vorhandene Datei. Den Quellcode für den Fix finden Sie in DECABuilder.VCX. Diese Bibliothek finden Sie im Normalfall im Verzeichnis Tools\XSource\VFXSource\Wizards\DEBuilder unterhalb des Hauptverzeichnisses von VFP, nachdem Sie die Datei XSource.ZIP in Tools\XSource entpackt haben.

Biographie

Doug Hennig ist Partner bei Stonefield Systems Group Inc. und Stonefield Software Inc. Er ist der Autor des preisgekrönten Stonefield Database Toolkit (SDT), des preisgekrönten Stonefield Query, des MemberData Editor, des Anchor Editor sowie der CursorAdapter- und DataEnvironment-Generatoren, die mit Microsoft Visual FoxPro ausgeliefert werden, des My Namespace und des aktualisierten Upsizing-Assistenten in Sedna. Doug ist Koautor der Serie „What's New in Visual FoxPro" (die letzte Ausgabe war „What's New in Nine") und des „The Hacker's Guide to Visual FoxPro 7.0". Er ist technischer Redakteur des „The Hacker's Guide to Visual FoxPro 6.0" und „The Fundamentals". Alle diese Bücher sind bei Hentzenwerke Publishing erschienen und werden in Deutschland von der dFPUG vertrieben. Innerhalb von zehn Jahren schrieb er über 100 Artikel für FoxTalk sowie zahlreiche Artikel für FoxPro Advisor und Advisor Guide. Er hat seit 1997 auf allen Microsoft FoxPro Developers Conferences sowie vor Usergroups und Regionalkonferenzen in ganz Nordamerika und weltweit gesprochen. Er ist einer der Administratoren für die VFPX-Website (www.codeplex.com/VFPX) und einer der Organisatoren der Southwest Fox-Konferenz (www.swfox.net). Er ist seit 1996 Microsoft Most Valuable Professional (MVP). Sie erreichen ihn über seine Websites www.stonefield.com und www.stonefieldquery.com oder per E-Mail unter dhennig@stonefield.com, Blog: doughennig.blogspot.com.

BEGLEITMATERIAL UND CODEBEISPIELE

Sie können die Begleitdatei FR200804_code.zip zu dieser Ausgabe im Dokumentenportal der dFPUG unter http://portal.dfpug.de/dfpug/Dokumente/FoxRockX kostenlos downloaden.
Das Archiv enthält folgende Einzeldateien:
doughennig200804_code.zip
Quellcode für das Sonderheft „Advantage Database Server für Visual FoxPro Entwickler" von Doug Hennig

FoxRockX™(ISSN 1866-4563)	FoxRockX erscheint zweimonatlich bei der ISYS GmbH
dFPUG c/o ISYS GmbH Frankfurter Strasse 21 B 61476 Kronberg, Deutschland Telefon: +49-6173-950903 Telefax: +49-6173-950904 Email: foxrockx@dfpug.de Herausgeber: Rainer Becker	Copyright © 2008 ISYS GmbH. Die Zeitschrift ist eine unabhängige Publikation der ISYS GmbH, Kronberg. Der Inhalt ist Eigentum der ISYS GmbH oder ihrer Vertragspartner / Lizenznehmer. Alle Rechte sind vorbehalten. Dies gilt auch für kostenlos veröffentlichte Beispielartikel, Begleitdateien oder Bilder in gedruckter oder elektronischer Form. Eine vollständige oder auszugsweise Weitergabe an Dritte ist nur mit schriftlicher Zustimmung der ISYS GmbH zulässig. Entsprechende Anfragen zum Vervielfältigen oder Publizieren von Material senden Sie bitte an Herrn Rainer Becker.
FoxRockX, FoxTalk, FoxTalk 2.0 und Visual Extend sind Warenzeichen der ISYS GmbH. Alle vorkommenden Produkt-/Firmennamen oder Service-Bezeichnungen sind ggf. registrierte Warenzeichen der jeweiligen Inhaber. Gedruckt in der Tschechischen Republik.	