Überblick über den Visual FoxPro-ODBC-Treiber

Visual FoxPro stellt eine leistungsfähige, objektorientierte Umgebung dar, mit der Datenbanken erstellt und Anwendungen entwickelt werden können. Der Microsoft Visual FoxPro-ODBC-Treiber versetzt Anwendungen in die Lage, über die ODBC-Schnittstelle (ODBC = Open Database Connectivity) Daten zu öffnen, abzufragen und zu aktualisieren, die im Format von Visual FoxPro oder im Format einer früheren FoxPro-Version vorliegen.

Zum Beispiel können Sie mit dem Microsoft Visual FoxPro-ODBC-Treiber folgendes tun:

- Verwenden von Microsoft Query, um Visual FoxPro-Daten von einem Microsoft Excel-Tabellenblatt aus abzufragen und zu aktualisieren.
- Erstellen von Serienbriefen mit Microsoft Word, für die Word Visual FoxPro-Daten verwendet.
- Abfragen und Aktualisieren von Visual FoxPro-Ansichten und -Tabellen aus Microsoft Access heraus.
- Verwenden von Visual FoxPro als Datenspeicher für Anwendungen, die in Visual Basic, Visual C++ oder C geschrieben sind.

Mit dem Treiber können Sie noch viele weitere Aufgaben erledigen. In der folgenden Tabelle sind einige Themen aufgelistet, die Ihnen den Einstieg erleichtern sollen.

Sie möchten	Siehe
Mehr darüber wissen, wie Visual FoxPro-Daten mit Microsoft Office-Anwendungen verwendet werden können.	Zugreifen auf Visual FoxPro-Daten aus einer Microsoft Office-Anwendung
Lernen, wie Visual FoxPro-Daten in einer Visual Basic-Anwendung verwendet werden können.	Verwenden des Visual FoxPro-ODBC-Treibers mit einer Visual Basic-Anwendung
Ein einfaches Beispiel sehen, in dem mit Visual C++ auf Visual FoxPro-Daten zugegriffen wird.	Verwenden des Visual FoxPro-ODBC-Treibers mit einer C- oder C++ Anwendung
Eine Liste anzeigen, in der die unterstützte Hardware und Software aufgeführt sind.	Systemanforderungen
Einen Überblick über den Treiber lesen.	Überblick über die Architektur des Treibers

Installieren des Visual FoxPro-ODBC-Treibers

Mit dem Setup-Programm des Visual FoxPro-ODBC-Treibers können Sie folgende Aufgaben erledigen:

- Hinzufügen neuer Komponenten.
- Löschen von installierten Komponenten.
- Erneutes Installieren, um fehlende Dateien und Einstellungen hinzuzufügen.
- Löschen aller zuvor installierten Komponenten.

So installieren Sie den Visual FoxPro-ODBC-Treiber

1. Starten Sie SETUP.EXE auf der Diskette 1 des Visual FoxPro-ODBC-Treibers.
2. Befolgen Sie die Anleitungen auf dem Bildschirm.

Nachdem Sie den Treiber auf Ihrem Computer installiert haben, erkennt das Setup-Programm die installierten Komponenten und zeigt weitere Dialogfelder an, mit denen Sie die Konfiguration des Treibers ändern können.

Zugreifen auf Visual FoxPro-Daten aus einer Microsoft Office-Anwendung

Sie können den Microsoft Visual FoxPro-ODBC-Treiber verwenden, um aus Ihren Microsoft Office für Windows 95-Anwendungen auf Visual FoxPro-Daten zuzugreifen.

Anwendung	Siehe
Microsoft Access	Abfragen und Aktualisieren von Visual FoxPro-Daten aus Microsoft Access
Microsoft Excel	Importieren von Daten in Microsoft Excel aus einer Visual FoxPro-Datenbank
Microsoft Word	Erstellen von Adreßetiketten in Microsoft Word mit Visual FoxPro-Daten

Hinzufügen einer Visual FoxPro-Datenquelle

Damit Sie aus Ihrer Anwendung heraus auf Visual FoxPro-Daten zugreifen können, müssen Sie über eine Datenquelle verfügen. Sie haben zwei Möglichkeiten, eine Datenquelle zu erstellen:

- Innerhalb einer Anwendung, wie z.B. Microsoft Word, Microsoft Excel oder Microsoft Access, die mit ODBC-Treibern arbeitet.
- Außerhalb einer Anwendung mit der Systemsteuerung von Windows 95 oder Windows NT.

Nachdem Sie eine Datenquelle auf Ihrem Computer erstellt haben, können Sie diese Datenquelle jedesmal dann verwenden, wenn Sie auf Visual FoxPro-Daten zugreifen möchten. Für den Fall, daß Sie auf mehrere Datenbanken oder Tabellen zugreifen möchten, können Sie für jede Datenbank bzw. jedes Verzeichnis eine eigene Datenquelle erstellen.

In den folgenden Schritten wird die Systemsteuerung verwendet, um eine Datenquelle zu erstellen. Informationen, wie eine Datenquelle aus einer Anwendung heraus erstellt wird, finden Sie unter Zugreifen auf Visual FoxPro-Daten aus einer Microsoft Office-Anwendung.

So fügen Sie eine Visual FoxPro-Datenquelle hinzu

1. Wählen Sie in der Systemsteuerung von Windows 95 oder Windows NT das Symbol **32-Bit-ODBC**. Diese Option steht zur Verfügung, nachdem Sie den Visual FoxPro-ODBC-Treiber oder einen anderen ODBC-Treiber installiert haben.
2. Klicken Sie im Dialogfeld **Datenquellen** auf **Hinzufügen**.
3. Wählen Sie im Dialogfeld **Datenquelle hinzufügen** in der Liste **Installierte ODBC-Treiber** den Eintrag des Microsoft Visual FoxPro-Treibers aus, und klicken Sie auf **OK**.
4. Es wird das Dialogfeld **ODBC Visual FoxPro-Setup** angezeigt. Geben Sie den Namen der Datenquelle ein; wählen Sie den Typ der Datenbank aus; wählen Sie die Datenbank oder das Verzeichnis aus; und klicken Sie auf **OK**.

   Der Name der neuen Datenquelle wird im Dialogfeld **Datenquellen** in der Liste **Benutzerdatenquellen (Treiber)** angezeigt.
5. Klicken Sie im Dialogfeld **Datenquellen** auf **Schließen**.

Ändern der Einstellungen einer Visual FoxPro-Datenquelle

Sie können die Einstellungen einer Visual FoxPro-Datenquelle ändern.

So ändern Sie die Einstellungen einer Visual FoxPro-Datenquelle

1. Wählen Sie in der Systemsteuerung von Windows 95 oder Windows NT das Symbol **32-Bit-ODBC**. Diese Option steht zur Verfügung, nachdem Sie den Visual FoxPro-ODBC-Treiber oder einen anderen ODBC-Treiber installiert haben.
2. Wählen Sie im Dialogfeld **Datenquellen** in der Liste **Benutzerdatenquellen (Treiber)** den Namen der Datenquelle aus, deren Einstellungen Sie ändern möchten, und klicken Sie auf **Installieren**.
3. Es wird das Dialogfeld **ODBC Visual FoxPro-Setup** angezeigt. Nehmen Sie die gewünschten Änderungen der Einstellungen vor, und klicken Sie auf **OK**.
4. Klicken Sie im Dialogfeld **Datenquellen** auf **Schließen**.

Die von Ihnen vorgenommenen Änderungen werden gespeichert. Diese Änderungen werden wirksam, sobald Sie das nächste Mal aus Ihrer Anwendung auf die Datenquelle zugreifen.

Löschen einer Visual FoxPro-Datenquelle

Sie können eine Visual FoxPro-Datenquelle löschen.

So löschen Sie eine Visual FoxPro-Datenquelle

1. Wählen Sie in der Systemsteuerung von Windows 95 oder Windows NT das Symbol **32-Bit-ODBC**. Diese Option steht zur Verfügung, nachdem Sie den Visual FoxPro-ODBC-Treiber oder einen anderen ODBC-Treiber installiert haben.
2. Wählen Sie im Dialogfeld **Datenquellen** in der Liste **Benutzerdatenquellen (Treiber)** den Namen der Datenquelle aus, die Sie löschen möchten.
3. Klicken Sie auf **Löschen**.

   Danach wird der Name der Datenquelle nicht mehr im Dialogfeld **Datenquellen** angezeigt.
4. Klicken Sie auf **Schließen**.

Herstellen einer Verbindung zu einer Visual FoxPro-Datenquelle

Eine Verbindung zu einer Visual FoxPro-Datenquelle können Sie mit Ihrer Microsoft Office-Anwendung oder über das SQL-API herstellen.

Verbindung herstellen aus	Siehe
Microsoft Access, Microsoft Excel oder Word.	Zugreifen auf Visual FoxPro-Daten aus einer Microsoft Office-Anwendung
Einer in C oder C++ geschriebenen Anwendung.	SQLConnect SQLDriverConnect
Einer Visual Basic-Anwendung.	Verwenden des Visual FoxPro-ODBC-Treibers mit einer Visual Basic-Anwendung

Verwenden von Verbindungszeichenfolgen

Sie können eine Verbindungszeichenfolge verwenden, um eine Verbindung zu einer Visual FoxPro-Datenquelle herzustellen.

Wenn Sie z.B. eine Verbindung zu der Datenquelle **TasTrade** herstellen und die aktuelle Einstellung der Option **Exklusiv** außer Kraft setzen möchten, müßten Sie folgende Zeichenfolge verwenden:

```
DSN=TasTrade;Exclusive=Yes
```

Eine Liste der Attributschlüsselwörter und -werte, die Sie in einer Verbindungszeichenfolge angeben können, finden Sie unter **SQLDriverConnect**.

Eine umfassende Erläuterung der Syntax einer Verbindungszeichenfolge finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

Zugreifen auf eine Visual FoxPro-Datenquelle aus Microsoft Excel

Wenn Sie Microsoft Query installiert haben, können Sie in Microsoft Excel eine Datenquelle erstellen, die mit Visual FoxPro-Daten verbunden ist.

So greifen Sie aus Microsoft Excel auf Visual FoxPro-Daten zu

1. Öffnen Sie ein Microsoft Excel-Tabellenblatt.
2. Klicken Sie im Menü **Daten** auf **Daten importieren**. Daraufhin wird Microsoft Query gestartet.
3. Klicken Sie im Dialogfeld **Datenquelle auswählen** auf **Andere**.
4. Klicken Sie im Dialogfeld **ODBC-Datenquellen** auf **Neu**.
5. Es wird das Dialogfeld **Datenquelle hinzufügen** angezeigt. Wählen Sie in der Liste **Installierte ODBC-Treiber** den Eintrag **Microsoft Visual FoxPro-Treiber** aus, und klicken Sie auf **OK**.
6. Es wird das Dialogfeld **ODBC Visual FoxPro-Setup** angezeigt. Geben Sie den Namen der Datenquelle ein; wählen Sie den Datenbanktyp aus; geben Sie den Pfad der Datenbank oder des Verzeichnisses ein; und klicken Sie auf **OK**.

   Der Name der neuen Datenquelle wird im Dialogfeld **ODBC-Datenquellen** in dem Textfeld **Datenquelle** angezeigt.
7. Klicken Sie auf **OK**.

   Der Name der neuen Datenquelle ist im Dialogfeld **Datenquelle auswählen** in dem Textfeld **Verfügbare Datenquellen** ausgewählt.
8. Klicken Sie auf **Verwenden**.

Sie können jetzt Tabellen zu der geöffneten Abfrage hinzufügen. Weitere Informationen, wie eine Abfrage erstellt wird, finden Sie unter Importieren von Daten in Microsoft Excel aus einer Visual FoxPro-Datenbank.

Importieren von Daten in Microsoft Excel aus einer Visual FoxPro-Datenbank

Sie können Visual FoxPro-Daten auf ein Microsoft Excel-Tabellenblatt importieren, wenn Sie für die Visual FoxPro-Daten eine Datenquelle definiert haben. Informationen, wie Sie eine Visual FoxPro-Datenquelle erstellen können, finden Sie unter Zugreifen auf eine Visual FoxPro-Datenquelle aus Microsoft Excel.

So importieren Sie Visual FoxPro-Daten auf ein Microsoft Excel-Tabellenblatt

1. Öffnen Sie ein Microsoft Excel-Tabellenblatt.
2. Klicken Sie im Menü **Daten** auf **Daten importieren**.

   Daraufhin wird Microsoft Query gestartet.
3. Wählen Sie im Dialogfeld **Datenquelle auswählen** eine Visual FoxPro-Datenquelle aus, und klicken Sie auf **Verwenden**.
4. Enthält die Datenbank, auf die Sie über die Datenquelle zugreifen, Tabellen, wählen Sie im Dialogfeld **Tabellen hinzufügen** den Namen einer Tabelle aus.

   Microsoft Query zeigt die hinzugefügte Tabelle in der oberen Hälfte des Abfragefensters an.

   Anmerkung Die Liste **Eigentümer** steht in diesem Dialogfeld nicht zur Verfügung, da der Treiber keine Eigentümer unterstützt. Die Liste **Datenbank** steht nicht zur Verfügung, da der Treiber Datenquellen mit mehreren Datenbanken nicht unterstützt.
5. Wählen Sie Felder für Ihre Abfrage aus, indem Sie diese Felder aus der Tabelle in die untere Hälfte des Abfragefensters ziehen.
6. Schließen Sie Microsoft Query.

   Die von Ihnen ausgewählten Daten werden auf das Microsoft Excel-Tabellenblatt importiert.

Erstellen von Adreßetiketten in Microsoft Word mit Visual FoxPro-Daten

Sie können Visual FoxPro-Daten in einem Microsoft Word für Windows 95-Dokument verwenden. Zum Beispiel können Sie die Kundeninformationen, die in einer Visual FoxPro-Tabelle gespeichert sind, verwenden, um Adreßetiketten zu erstellen.

So erstellen Sie Adreßetiketten

1. Erstellen Sie in Microsoft Word ein neues, leeres Dokument.
2. Klicken Sie im Menü **Extras** auf **Seriendruck**.
3. Klicken Sie im Seriendruck-Manager auf **Erstellen** und anschließend auf **Adreßetiketten**.
4. Klicken Sie unter **Hauptdokument** auf **Aktives Fenster**.
5. Klicken Sie unter **Datenquelle** auf **Daten importieren**, und klicken Sie dann auf **Datenquelle öffnen**.
6. Klicken Sie im Dialogfeld **Datenquelle öffnen** auf **MS Query**.
7. Wählen Sie im Dialogfeld **Datenquelle auswählen** eine Visual FoxPro-Datenquelle aus, und klicken Sie dann auf **Verwenden**.
8. Enthält die Datenbank, auf die Sie über die Datenquelle zugreifen, Tabellen, wählen Sie im Dialogfeld **Tabellen hinzufügen** den Namen einer Tabelle aus.

   Microsoft Query zeigt die hinzugefügte Tabelle in der oberen Hälfte des Abfragefensters an.
9. Wählen Sie Felder für Ihre Abfrage aus, indem Sie diese Felder aus der Tabelle in die untere Hälfte des Abfragefensters ziehen.
10. Klicken Sie im Menü **Datei** auf **Daten an Microsoft Word zurücksenden**.

    Microsoft Query wird geschlossen, und die Daten, die Sie ausgewählt haben, stehen in Ihrem Seriendruckdokument zur Verfügung.
11. Klicken Sie unter **Hauptdokument** auf **Einrichten**.
12. Es wird das Dialogfeld **Etiketten einrichten** angezeigt. Wählen Sie den Drucker und die Etikettenmarke aus, und klicken Sie dann auf **OK**.
13. Wählen Sie im Dialogfeld **Etiketten erstellen** die Felder aus, deren Inhalte auf die Adreßetiketten gedruckt werden sollen, und klicken Sie dann auf **OK**.
14. Klicken Sie im Seriendruck-Manager unter **Daten mit dem Dokument verbinden** auf **Ausführen**.
15. Es wird das Dialogfeld **Seriendruck** angezeigt. Aktivieren Sie die gewünschten Optionen, und klicken Sie dann auf **Verbinden**.

Importieren von Visual FoxPro-Daten in Microsoft Access

Sie können Daten, die in einer Visual FoxPro-Datenbank gespeichert sind, mit dem Befehl **Importieren** in eine Microsoft Access-Datenbank importieren.

So importieren Sie Visual FoxPro-Daten in eine Microsoft Access-Datenbank

1. Öffnen Sie eine Microsoft Access-Datenbank.
2. Zeigen Sie im Menü **Datei** auf **Externe Daten**, und klicken Sie auf **Importieren**.
3. Es wird das Dialogfeld **Importieren** angezeigt. Klicken Sie im Listenfeld **Dateityp** auf den Eintrag **ODBC-Datenbanken( )**.
4. Daraufhin wird das Dialogfeld **SQL-Datenquellen** angezeigt. Wählen Sie die Visual FoxPro-Datenquelle aus, über die eine Verbindung zu den FoxPro-Daten hergestellt wird, die Sie abfragen möchten, und klicken Sie auf **OK**.
5. Es wird das Dialogfeld **Objekte importieren** angezeigt. Wählen Sie die Tabellen aus, die Sie importieren möchten, und klicken Sie auf **OK**.

   Die Namen der Visual FoxPro-Tabellen, die Sie importiert haben, werden im Microsoft Access-Datenbankfenster auf der Registerkarte **Tabellen** angezeigt.

Ab jetzt können Sie die importierten Visual FoxPro-Tabellen mit Microsoft Access verwalten und bearbeiten. Die importierten Daten entsprechen einem Snapshot der Daten, die in Visual FoxPro gespeichert sind; Änderungen, die Sie an den importierten Daten vornehmen, werden nicht an die Visual FoxPro-Datenquelle gesendet.

Wenn Sie möchten, daß Änderungen, die Sie in Microsoft Access vornehmen, auch an den Daten in der Visual FoxPro-Datenquelle vorgenommen werden, finden Sie die entsprechenden Informationen unter Abfragen und Aktualisieren von Visual FoxPro-Daten aus Access.

Abfragen und Aktualisieren von Visual FoxPro-Daten aus Microsoft Access

Wenn Sie entweder den Befehl **Tabellen verknüpfen** oder im Dialogfeld **Neue Tabelle** die Option **Tabelle verknüpfen** verwenden, können Sie Daten, die in einer Visual FoxPro-Datenbank gespeichert sind, aus einer Microsoft Access-Datenbank heraus abrufen und aktualisieren.

So binden Sie eine Visual FoxPro-Tabelle mittels Verknüpfen in eine Microsoft Access-Datenbank ein

1. Öffnen Sie eine Microsoft Access-Datenbank.
2. Klicken Sie auf der Registerkarte **Tabellen** auf **Neu**.
3. Wählen Sie im Dialogfeld **Neue Tabelle** den Eintrag **Tabelle verknüpfen** aus, und klicken Sie auf **OK**.
4. Es wird das Dialogfeld **Verknüpfen** angezeigt. Klicken Sie im Listenfeld **Dateityp** auf den Eintrag **ODBC-Datenbanken( )**.
5. Daraufhin wird das Dialogfeld **SQL-Datenquellen** angezeigt. Wählen Sie die Datenquelle aus, über die eine Verbindung zu den FoxPro-Daten hergestellt wird, die Sie abfragen möchten, und klicken Sie auf **OK**.
6. Es wird das Dialogfeld **Tabellen verknüpfen** angezeigt. Wählen Sie die Tabellen aus, die Sie abfragen und aktualisieren möchten, und klicken Sie auf **OK**.

   Die Namen der auf diese Weise eingebundenen ("verknüpften") Visual FoxPro-Tabellen werden im Microsoft Access-Datenbankfenster auf der Registerkarte **Tabellen** angezeigt.

Die Daten, die in den eingebundenen Visual FoxPro-Tabellen gespeichert sind, können Sie jetzt aus Microsoft Access heraus abfragen und aktualisieren. Alle Änderungen, die Sie an den eingebundenen Daten vornehmen, werden an die Visual FoxPro-Datenquelle gesendet.

Wie Sie vorgehen müssen, damit sich die Änderungen, die Sie in Microsoft Access vornehmen, nicht auf die Daten in der Visual FoxPro-Datenquelle auswirken, ist unter Importieren von Visual FoxPro-Daten in Microsoft Access beschrieben.

Verwenden des Visual FoxPro-ODBC-Treibers mit einer Visual Basic-Anwendung

Aus einer Visual Basic-Anwendung können Sie auf Visual FoxPro-Daten zugreifen, indem Sie ein Daten-Steuerelement (**Data**) erstellen und dieses mit einer Visual FoxPro-Datenquelle verbinden.

So verbinden Sie in Visual Basic, Version 4.0, ein Daten-Steuerelement mit Visual FoxPro-Daten

1. Erstellen Sie eine Datenquelle namens **Test**, die an die Beispieldatenbank **TasTrade** gebunden ist, die zu Visual FoxPro gehört. Bei der Standardinstallation von Visual FoxPro wird die Beispieldatenbank **TasTrade** an der folgenden Stelle abgelegt:

   ```
   c:\vfp\samples\mainsamp\data\tastrade.dbc
   ```

2. Erstellen Sie in Visual Basic, Version 4.0, eine neue Form, und ordnen Sie auf dieser Form ein Textfeld und ein Daten-Steuerelement (**Data**) an.
3. Stellen Sie die **Connect**-Eigenschaft des Daten-Steuerelements wie folgt ein:

   ```
   ODBC;DATABASE=tastrade;DSN=test
   ```

4. Stellen Sie die **RecordsetType**-Eigenschaft auf folgenden Wert ein:

   ```
   2 - Snapshot
   ```

5. Stellen Sie die **RecordSource**-Eigenschaft auf folgenden Wert ein:

   ```
   customer
   ```

6. Stellen Sie die **DataSource**-Eigenschaft des Textfelds auf den Standardnamen des Daten-Steuerelements ein:

   ```
   data1
   ```

7. Stellen Sie die **DataField**-Eigenschaft des Textfelds auf folgenden Wert ein:

   ```
   customer_id
   ```

8. Führen Sie die Form aus, und verwenden Sie das Daten-Steuerelement, um die Werte des Feldes **customer_id** zu durchlaufen, die in der Visual FoxPro-Beispieldatenbank **TasTrade** gespeichert sind.

Verwenden des Visual FoxPro-ODBC-Treibers mit einer C- oder C++ Anwendung

Beispiel

Ihre Anwendung greift auf Visual FoxPro-Daten zu, indem sie eine **SQLExecute**- oder eine **SQLExecDirect**-Anweisung an Visual FoxPro sendet. Eine solche Anweisung kann folgendes enthalten:

- SQL-Anweisungen, die Bestandteil der Visual FoxPro-Sprache sind, wie z.B. der Befehl DROP TABLE.
- Unterstützte ODBC SQL-Grammatik.
- Nicht-SQL-Elemente der Visual FoxPro-Sprache, wie z.B. unterstützte SET-Befehle.

Weitere Information zu den SQL-Anweisungen, die Bestandteil von Visual FoxPro sind, finden Sie in der Dokumentation von Visual FoxPro.

Verwenden des Visual FoxPro-ODBC-Treibers mit einer C- oder C++ Anwendung

Im folgenden Beispiel wird das ODBC C-API verwendet, um Daten abzurufen, die in der Visual FoxPro-Beispieldatenbank **TasTrade** in der Tabelle **employee** im Feld **last_name** gespeichert sind. Diese Datenbank wird mit Visual FoxPro geliefert und standardmäßig an der folgenden Stelle installiert:

```
c:\vfp\samples\mainsamp\data\tastrade.dbc
```

Das Beispiel zeigt jeweils einen Nachnamen in einem Meldungsfeld an und ermöglicht es Ihnen, durch Klicken auf **OK** den jeweils nächsten Nachnamen anzuzeigen. Es wird vorausgesetzt, daß eine Datenquelle namens **tastrade** eingerichtet wurde, die an die Datenbank **tastrade.dbc** gebunden ist.

Anmerkung Alle ODBC API-Aufrufe sollten auf Fehler geprüft werden; in diesem Beispiel wird aus Gründen der Übersichtlichkeit auf die Fehlerprüfung verzichtet.

```
//Die Firma Microsoft und ihre Lieferanten lehnen
//bezüglich dieses Beispiels jede Garantie bis zum
//größtmöglichen vom Gesetz erlaubten Umfang ab. Dies
//gilt sowohl für ausdrückliche als auch indirekte
//Garantien einschließlich, aber nicht beschränkt auf
//indirekte Garantien der Vertriebsfähigkeit und
//Eignung für bestimmte Zwecke.

#include <windows.h>
#include <sql.h>
#include <sqlext.h>
#include <stdlib.h>
#include <mbstring.h>

#define MAX_DATA 100
#define MYSQLSUCCESS(rc) ((rc==SQL_SUCCESS)||(rc==SQL_SUCCESS_WITH_INFO))

class direxec
{
   RETCODE rc;          // ODBC-Rückgabecode
   HENV henv;           // Umgebung (Environment)
   HDBC hdbc;           // Verbindungskennung (Connection handle)
   HSTMT hstmt;         // Anweisungskennung (Statement handle)
   unsigned char szData[MAX_DATA];  // Variable für Rückgabedaten
   SDWORD cbData;       // Ausgabelänge der Daten
   unsigned char chr_ds_name[SQL_MAX_DSN_LENGTH];  // Name der Datenquelle
(DSN)

public:
   direxec();           // Konstruktor
   void sqlconn();      // Reservieren der Kennungen (handles)
                        // und Verbindung herstellen.
   void sqlexec(unsigned char *);  // SQL-Anweisung ausführen
   void sqldisconn();   // Freigeben der Kennungen (handles)
                        // und Verbindung trennen.
   void error_out();    // Fehler anzeigen.
};

// Konstruktor initialisiert die Zeichenfolge
// chr_ds_name mit dem Namen der Datenquelle.
direxec::direxec()
```

```
{
   _mbscpy(chr_ds_name,(const unsigned char *)"tastrade");
}

// Reservieren der Umgebungskennung (environment handle),
// Reservieren der Verbindungskennung (connection handle),
// Verbindung zur Datenquelle herstellen und
// Reservieren der Anweisungskennung (statement handle).
void direxec::sqlconn(void)
{
   SQLAllocEnv(&henv);
   SQLAllocConnect(henv,&hdbc);
   rc=SQLConnect(hdbc,chr_ds_name,SQL_NTS,NULL,0,NULL,0);

   // Kennungen (handles) freigeben, Fehlermeldung
   // anzeigen und Funktion beenden.
   if (!MYSQLSUCCESS(rc))
   {
      SQLFreeEnv(henv);
      SQLFreeConnect(hdbc);
      error_out();
      exit(-1);
   }

   rc=SQLAllocStmt(hdbc,&hstmt);

}

// SQL-Befehl mit ODBC API-Funktion SQLExecDirect() ausführen.
void direxec::sqlexec(unsigned char * cmdstr)
{
   rc=SQLExecDirect(hstmt,cmdstr,SQL_NTS);
   if (!MYSQLSUCCESS(rc))  //Fehler
   {
      error_out();
      // Kennungen (handles) freigeben und Verbindung trennen.
      SQLFreeStmt(hstmt,SQL_DROP);
      SQLDisconnect(hdbc);
      SQLFreeConnect(hdbc);
      SQLFreeEnv(henv);
      exit(-1);
   }
   else
   {
      for (rc=SQLFetch(hstmt); rc == SQL_SUCCESS; rc=SQLFetch(hstmt))
      {
         SQLGetData(hstmt,1,SQL_C_CHAR,szData,sizeof(szData),&cbData);
         // In diesem Beispiel werden die Daten der Einfachheit
      // halber in einem Meldungsfeld angezeigt. Normalerweise
      // würde die ODBC API-Funktion SQLBindCol() verwendet, um
         // einzelne Datenspalten einer Ergebnisgruppe (result set)
         // an einen Speicherbereich zu binden. Ausführliche Infor-
         // mationen finden Sie in ODBC 2.0 Programmer's Reference
         // in Kapitel 15, "Returning Results".
         MessageBox(NULL,(const char *)szData,"ODBC",MB_OK);
      }
   }
```

```
}

// Freigeben der Anweisungskennung (statement handle),
// Verbindung trennen, Freigeben der Verbindungskennung (connection handle)
// und Freigeben der Umgebungskennung (environment handle).
void direxec::sqldisconn(void)
{
   SQLFreeStmt(hstmt,SQL_DROP);
   SQLDisconnect(hdbc);
   SQLFreeConnect(hdbc);
   SQLFreeEnv(henv);
}

// Fehlermeldung in einem Meldungsfeld anzeigen, das
// eine OK-Schaltfläche hat.
void direxec::error_out(void)
{
   unsigned char szSQLSTATE[10];
   SDWORD nErr;
   unsigned char msg[SQL_MAX_MESSAGE_LENGTH+1];
   SWORD cbmsg;

   while(SQLError(0,0,hstmt,szSQLSTATE,&nErr,msg,sizeof(msg),&cbmsg)==
SQL_SUCCESS)
   {
      wsprintf((char *)szData,"Fehler:
\nSQLSTATE=%s,Treiberfehler=%ld,msg='%s'",
         szSQLSTATE,nErr,msg);
      MessageBox(NULL,(const char *)szData,"ODBC-Fehler",MB_OK);

   }

}

int WINAPI WinMain (HANDLE hInstance, HANDLE hPrevInstance,
               LPSTR lpszCmdLine, int nCmdShow)
{
   // Eine Instanz des Objekts direxec deklarieren.
   direxec x;

   // Kennungen (handles) reservieren und Verbindung herstellen.
   x.sqlconn();

   // SQL-Befehl "SELECT last_name FROM employee" ausführen.
   x.sqlexec((UCHAR FAR *)"SELECT last_name FROM employee");

   // Kennungen (handles) freigeben und Verbindung trennen.
   x.sqldisconn();

   // Erfolg zurückgeben; Beispiel erfolgreich ausgeführt.
   return (TRUE);
}
```

Systemanforderungen

Die nachfolgend bezüglich des Betriebssystems und des Plattenspeicherplatzes angegebenen Systemanforderungen müssen mindestens erfüllt sein, um den Treiber erfolgreich installieren zu können. Nachdem Sie den Treiber installiert haben, können Sie die Anwendungssoftware wählen, mit der Sie auf Visual FoxPro-Daten zugreifen möchten.

Anforderungen für die Installation

Damit Sie den Microsoft Visual FoxPro-ODBC-Treiber installieren können, benötigen Sie:

- Windows NT, Version 3.50 oder höher, oder Windows 95
- 2 MB Speicherplatz auf der Festplatte

Informationen, wie der Treiber installiert wird, finden Sie unter Installieren des Visual FoxPro-ODBC-Treibers.

Zugreifen auf Visual FoxPro-Daten

Damit Sie auf Daten zugreifen können, die im Format von Microsoft Visual FoxPro oder im Format von Microsoft FoxPro 2.*x* vorliegen, müssen Sie über folgende Komponenten verfügen:

- ODBC-Client-Software (wird automatisch mit dem Treiber installiert)
- Microsoft Visual FoxPro-ODBC-Treiber
- Eine Anwendungssoftware folgenden Typs:
  - Microsoft Office-Anwendung, z.B. Microsoft Excel oder Word
  - ODBC-Anwendung in C oder C++
  - ODBC-Anwendung in Visual Basic, Version 4.0
- Daten, die in folgender Form vorliegen:
  - In einer Datenbank oder in einem Verzeichnis mit freien Tabellen im Format von Visual FoxPro
  - In einer Tabelle im Format von FoxPro, Version 2.0, 2.5 oder 2.6

Der Visual FoxPro-ODBC-Treiber unterstützt Zwei-Byte-Zeichensätze (DBCS = double-byte character sets). Weitere Informationen finden Sie unter Unterstützung internationaler Sprachen.

Der Treiber unterstützt keine für Windows, Version 3.1, geschriebenen 16-Bit-Anwendungen.

Unterstützte FoxPro-Versionen

Sie können über den Visual FoxPro-ODBC-Treiber auf Daten zugreifen, die in FoxPro-Tabellen gespeichert sind. Für die Daten werden folgende FoxPro-Versionen unterstützt:

- 2.0
- 2.5
- 2.6
- Visual FoxPro (alle Versionen)

Wenn Sie auf Daten zugreifen, die im Format von Visual FoxPro gespeichert sind, haben Sie die Wahl, eine Verbindung zu einer Datenbank herzustellen, unabhängig davon, ob diese Tabellen enthält, oder zu einem Verzeichnis mit freien Tabellen.

Weitere Informationen, wie Sie eine Verbindung zu einer Datenquelle herstellen, finden Sie unter Hinzufügen einer Visual FoxPro-Datenquelle.

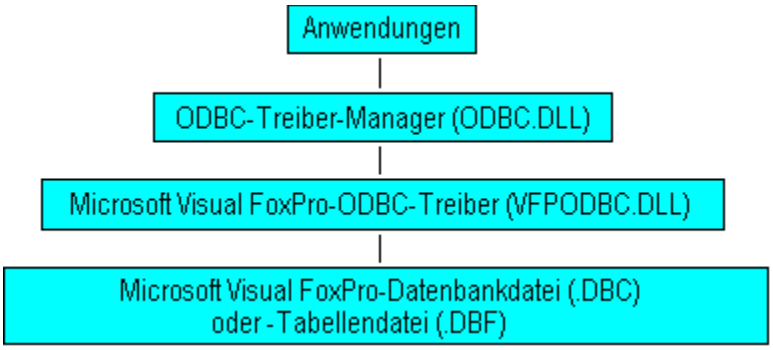
Überblick über die Architektur des Treibers

Der Microsoft Visual FoxPro-ODBC-Treiber ist ein 32-Bit-Treiber, der es Ihnen ermöglicht, eine Microsoft Visual FoxPro-Datenbank oder FoxPro-Tabellen über die ODBC-Schnittstelle (ODBC = Open Database Connectivity) zu öffnen und abzufragen. Auf FoxPro-Daten können Sie wie folgt zugreifen:

- Aus einer Microsoft Office-Anwendung, wie z.B. Microsoft Excel oder Microsoft Word, die über Microsoft Query mit der ODBC-Schnittstelle kommuniziert.
- Aus einer in Visual C++ oder C geschriebenen Anwendung, die das ODBC SDK-API verwendet.
- Aus einer Anwendung, die in Visual Basic oder Visual Basic für Applikationen geschrieben ist.

In jedem Fall geht eine Anforderung nach Informationen über die ODBC-API. Der ODBC Treiber-Manager arbeitet mit dem Visual FoxPro-ODBC-Treiber zusammen, um FoxPro-Tabellen oder -Datenbanken zu öffnen und Daten aus diesen abzurufen.

Die Architektur ist im folgenden Diagramm dargestellt:



Unterstützung für Lesezeichen (bookmarks)

Der Visual FoxPro-ODBC-Treiber unterstützt einfache Lesezeichen (bookmarks). Wenn Sie bei einem Aufruf der Funktion **SQLGetInfo** für das Argument *fInfoType* den Wert SQL_BOOKMARK_PERSISTENCE angeben, gibt die Funktion den Wert SQL_BP_SCROLL zurück.

Informationen zu Lesezeichen (bookmarks) finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

Unterstützung für Parallelzugriff (concurrency)

Der Visual FoxPro-ODBC-Treiber unterstützt schreibgeschützten Parallelzugriff (concurrency). Wenn *fOption* auf SQL_CONCURRENCY eingestellt ist, kann Ihre Anwendung die Funktion **SQLSetStmtOption** mit SQL_CONCUR_READ_ONLY für *vParam* aufrufen.

Weitere Informationen zu Parallelzugriff (concurrency) und **SQLSetStmtOption** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

Schreibgeschützter Parallelzugriff (concurrency)

Der Cursor kann nicht aktualisiert werden.

Zeilenversionsvergleiche (row versioning)

Im wesentlichen Unterstützung für TIMESTAMP-Werte, wobei zum Zeitpunkt einer Aktualisierung Zeilenversionen verglichen werden.

Unterstützte Cursor-Modelle

Der Visual FoxPro-ODBC-Treiber unterstützt sowohl Block-Cursor (block cursors; Datensatzgruppe (rowset)) als auch statische Cursor (static cursors). Statische Cursor werden für jeden Treiber unterstützt, der die Übereinstimmungsstufe ODBC Level 1 hat. Der Treiber unterstützt weder dynamische (dynamic) noch schlüsselgruppengesteuerte (keyset-driven), noch gemischte (schlüsselgruppengesteuerte und dynamische) Cursor.

Ihre Anwendung kann **SQLSetStmtOption** mit einer der SQL_CURSOR_TYPE-Optionen SQL_CURSOR_FORWARD_ONLY (Block-Cursor) oder SQL_CURSOR_STATIC (statischer Cursor) aufrufen.

Anmerkung Wenn Sie **SQLSetStmtOption** mit einer anderen SQL_CURSOR_TYPE-Option als SQL_CURSOR_FORWARD_ONLY oder SQL_CURSOR_STATIC aufrufen, gibt die Funktion SQL_SUCCESS_WITH_INFO mit SQLSTATE gleich 01S02 ("Optionswert geändert.") zurück. Der Treiber stellt alle nichtunterstützten Cursor-Modi auf SQL_CURSOR_STATIC ein.

Weitere Informationen zu den Cursor-Typen sowie zu **SQLSetStmtOption** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

Block-Cursor (block cursor)

Eine nur vorwärts durchlaufbare (forward-scrolling) schreibgeschützte (read-only) Ergebnismenge, die an den Client zurückgegeben wird. Der Client muß den für die Daten benötigten Speicherplatz verwalten.

Statischer Cursor (static cursor)

Ein Snapshot einer Datenmenge, die durch eine Abfrage definiert ist. In einem statischen Cursor schlägt sich keine der Änderungen nieder, die andere Benutzer an den zugrundeliegenden Daten vornehmen, nachdem der Cursor erstellt wurde. Der Arbeitsspeicherpuffer des Cursors wird von der ODBC-Cursor-Bibliothek (ODBC cursor library) verwaltet, die die Durchführung eines Bildlaufs vorwärts und rückwärts (forward and backward scrolling) gestattet.

Datensatzgruppe (rowset)

Datenblöcke, die in einem Cursor gespeichert sind und Zeilen entsprechen, die aus einer Datenquelle abgerufen wurden.

Datentypen

Die vom Treiber unterstützten Datentypen werden über das ODBC-API und in Microsoft Query mitgeteilt.

Datentypen in C-Anwendungen

In einer C- oder C++ Anwendung können Sie die Funktion **SQLGetTypeInfo** verwenden, um eine Liste der Datentypen abzurufen, die der Visual FoxPro-ODBC-Treiber unterstützt.

Datentypen in Anwendungen, die Microsoft Query verwenden

Wenn Ihre Anwendung Microsoft Query verwendet, um eine neue Tabelle für eine Visual FoxPro-Datenquelle zu erstellen, zeigt Microsoft Query das Dialogfeld **Neue Tabellendefinition** an. Unter **Feldbeschreibung** werden im Listenfeld **Typ** die Visual FoxPro-Felddatentypen angezeigt, die durch einzelne Zeichen dargestellt sind.

Unterstützung internationaler Sprachen

Der Microsoft Visual FoxPro-ODBC-Treiber unterstützt:

- Zwei-Byte-Zeichensätze (DBCS = double-byte character sets)
- Mehrere Sortierfolgen

Eine Sortierfolge definiert die Sortierreihenfolge für Daten, die in einer Visual FoxPro-Tabelle oder -Datenbank gespeichert sind. Der Treiber wird standardmäßig so konfiguriert, daß er die Sortierfolgen verwendet, die für die Sprachversion Ihres Betriebssystems festgelegt sind.

Eine Liste der unterstützten Sortierfolgen finden Sie unter SET COLLATE.

Gebietsschema (locale)

Die Informationen, die zu einer bestimmten Sprache und einem bestimmten Land gehören. Ein Gebietsschema gibt bestimmte Einstellungen an, wie z.B. Dezimaltrennzeichen, Datums- und Zeitformate sowie die Reihenfolge, in der Zeichen sortiert werden.

Sortierreihenfolge

Sortierreihenfolgen enthalten die Sortierregeln unterschiedlicher Gebietsschemas und ermöglichen es Ihnen, Daten entsprechend der zugehörigen Sprachen korrekt zu sortieren. In Visual FoxPro bestimmt die aktuelle Sortierreihenfolge die Ergebnisse von Zeichenfolgenvergleichen sowie die Reihenfolgen, in denen Datensätze in indizierten oder sortierten Tabellen erscheinen.

Dialogfeld "ODBC Visual FoxPro-Setup"

Ermöglicht es Ihnen, eine Visual FoxPro-Datenquelle hinzuzufügen oder die Einstellungen einer vorhandenen Visual FoxPro-Datenquelle zu ändern.

Die Optionen des Dialogfelds

Datenquellenname In dieses Feld geben Sie den Namen der Datenquelle ein.

Beschreibung In dieses Feld können Sie eine Beschreibung der Datenquelle eingeben.

Datenbanktyp Ermöglicht es Ihnen, den Typ der Datenbank anzugeben, mit der Ihre Datenquelle verbunden sein soll.

Visual FoxPro-Datenbank (.DBC) Gibt an, daß die Datenquelle mit einer Visual FoxPro-Datenbank (.DBC-Datei) sowie allen Tabellen und lokalen Ansichten verbunden werden soll, die zu der Datenbank gehören.

Verzeichnis mit freien Tabellen Gibt an, daß die Datenquelle mit einem Verzeichnis mit freien Tabellen verbunden werden soll. Alle Tabellen, die sich ebenfalls in diesem Verzeichnis befinden und zu einer Datenbank gehören, werden von den ODBC-Katalogfunktionen (catalog functions), z.B. **SQLColumns** und **SQLTables**, ignoriert. Auf Datenbanktabellen kann zugegriffen werden, indem über **SQLExecute** oder **SQLExecDirect** SQL SELECT-Anweisungen gesendet werden.

Pfad Zeigt den Pfad und den Namen der Datenbank oder des Verzeichnisses mit freien Tabellen an, mit der bzw. mit dem die Datenquelle verbunden wird.

Durchsuchen Ermöglicht es Ihnen, Ihren Computer oder Ihr Netzwerk nach der Datenbank oder dem Verzeichnis zu durchsuchen, mit der bzw. dem Sie die Datenquelle verbinden möchten.

Optionen Erweitert das Dialogfeld, so daß Sie die Optionen des Visual FoxPro-ODBC-Treibers sehen und einstellen können.

Treiber

Sortierreihenfolge Die Reihenfolge, in der Felder sortiert werden. Die Standardreihenfolgen entsprechen den Reihenfolgen, die von Ihrer Sprachversion des Betriebssystems unterstützt werden. Eine Zusammenstellung der unterstützten Sortierreihenfolgen finden Sie unter SET COLLATE.

Exklusiv Wenn dieses Kontrollkästchen aktiviert ist, öffnet der Treiber die Visual FoxPro-Datenbank im exklusiven Modus, wenn Sie über die Datenquelle auf Daten zugreifen. Solange die Datenbank im exklusiven Modus geöffnet ist, können andere Benutzer nicht auf die Datenbank oder die Tabellen der Datenbank zugreifen. Die Tabellen einer im exklusiven Modus geöffneten Datenbank werden zur gemeinsamen Nutzung (SHARED) geöffnet. Soll eine Tabelle im exklusiven Modus geöffnet werden, verwenden Sie einen SET EXCLUSIVE-Befehl. Dieses Kontrollkästchen ist deaktiviert, wenn **Datenbanktyp** auf **Verzeichnis mit freien Tabellen** eingestellt ist.

Daten im Hintergrund lesen Legt fest, ob Datensätze im Hintergrund gelesen werden (kontinuierliches Lesen, progressive fetching) oder ob Ihre Anwendung wartet, bis alle Datensätze der Datensatzgruppe gelesen sind.

Unterstützte ODBC SQL-Grammatik

Der Microsoft Visual FoxPro-ODBC-Treiber unterstützt:

- Alle SQL-Anweisungen und -Klauseln der ODBC SQL-Grammatik Minimum
- Eine weitere SQL-Anweisung der ODBC SQL-Grammatik Core

In der folgenden Tabelle sind die vom Treiber unterstützten Elemente entsprechend der ODBC SQL-Grammatikstufe aufgeführt.

Stufe	Elemente	Bestandteil
Minimum	Datendefinitionssprache (DDL = Data Definition Language)	CREATE TABLE und DROP TABLE
	Datenbearbeitungssprache (DML = Data Manipulation Language)	SELECT, INSERT, UPDATE und DELETE
	Ausdrücke (expressions)	Einfache Ausdrücke (z.B. A>B+C)
	Datentypen	CHAR, VARCHAR oder LONG VARCHAR

Zusätzlich zu der unterstützten ODBC SQL-Grammatik unterstützt der Visual FoxPro-ODBC-Treiber uneingeschränkt die Visual FoxPro-Sprachsyntax folgender Visual FoxPro-Befehle:

ALTER TABLE

CREATE TABLE

DELETE

DELETE TAG

DROP TABLE

INDEX

INSERT

SELECT

UPDATE

Registrierungseinträge

Wenn Sie den Visual FoxPro-ODBC-Treiber installieren, aktualisiert das Installationsprogramm die Registrierung Ihres Computers (HKEY_LOCAL_MACHINE\SOFTWARE\ODBC\ODBCINST.INI), indem es einen neuen Schlüssel namens **Microsoft Visual FoxPro-Treiber** einfügt. Unter diesem Schlüssel werden folgende Werte hinzugefügt:

Wertname	Werttyp	Wert
APILevel	REG_SZ	"1"
ConnectFunctions	REG_SZ	"YYN"
Driver	REG_SZ	Systempfad zur Datei VFPODBC.DLL
DriverODBCVer	REG_SZ	"02.50"
FileExtns	REG_SZ	"*.dbf,*.cdx,*.fpt"
FileUsage	REG_SZ	"1"
Setup	REG_SZ	Systempfad zur Datei VFPODBC.DLL
SQLLevel	REG_SZ	"0"

Außerdem fügt das Installationsprogramm zu dem Schlüssel HKEY_CURRENT_USER\SOFTWARE\ODBC\ODBC.INI den Schlüssel **Visual FoxPro-Dateien** hinzu, der dem standardmäßigen Visual FoxPro-Treiber entspricht. Unter diesem Schlüssel fügt das Installationsprogramm folgende Werte hinzu:

Wertname	Werttyp	Wert
Driver	REG_SZ	Systempfad zur Datei VFPODBC.DLL

Jedesmal, wenn Sie eine Visual FoxPro-ODBC-Datenquelle zu Ihrer ODBC-Konfiguration hinzufügen, wird ein neuer Schlüssel für den Namen dieser Datenquelle hinzugefügt. Die Werte, die für die Datenquelle in der Registrierung gespeichert werden, entsprechen den Werten, die Sie im Dialogfeld **ODBC Visual FoxPro-Setup** eingestellt haben:

Wertname	Werttyp	Wert
Collate	REG_SQ	Eine der unterstützten Sortierreihenfolgen (collating sequence)
Description	REG_SZ	Eine Benutzerbeschreibung der Datenquelle
Driver		Der Systempfad der Datei VFPODBC.DLL
Exclusive		"Yes" oder "No"
BackgroundFetch		"Yes" oder "No" (Daten im Hintergrund lesen)
SourceDB	REG_SZ	Der Pfad der .DBC-Datei
SourceType	REG_SZ	"DBC" oder "DBF"

Sie sollten nicht direkt auf diese Informationen zugreifen. Alle Änderungen, die in der Registrierung vorgenommen werden müssen, wenn Sie eine Datenquelle hinzufügen, ändern oder löschen, werden vom ODBC-Administrator verwaltet.

Sie können einige dieser Schlüsselwörter und Werte nutzen, wenn Sie die ODBC-API-Funktion **SQLDriverConnect** verwenden.

Unterstützte skalare Funktionen

Der Visual FoxPro-ODBC-Treiber unterstützt von den fünf Typen skalarer Funktionen (scalar functions), die in der ODBC SQL-Grammatik definiert sind, die drei folgenden:

- Zeichenfolgenfunktionen (String functions)
- Numerische Funktionen
- Zeit- und Datumsfunktionen

Zeichenfolgenfunktionen (String functions)

In der folgenden Tabelle sind die ODBC-Zeichenfolgenfunktionen zusammengestellt, die der Visual FoxPro-ODBC-Treiber unterstützt. Für die Fälle, in denen sich für gleiche Funktionen die Visual FoxPro-Grammatik und die ODBC-Syntax unterscheiden, ist jeweils das Visual FoxPro-Gegenstück aufgeführt.

ODBC-Grammatik	**Visual FoxPro-Grammatik**
ASCII (*string_exp*)	ASC *(Zeichenfolge*)
CHAR *(code*)	CHR (*ANSICode*)
CONCAT (*string_exp1, string_exp2*)	*Zeichenfolge1 + Zeichenfolge2*
DIFFERENCE (*string_exp1, string_exp2*)	
INSERT (*string_exp1, start, length, string_exp2*)	STUFF (*Zeichenfolge1, Beginn, Länge, Zeichenfolge2*)
LCASE (*string_exp*)	LOWER (*Zeichenfolge*)
LEFT (*string_exp, count*)	
LENGTH (*string_exp*)	LEN (*Zeichenfolge*)
LTRIM (*string_exp*)	
REPEAT (*string_exp, count*)	REPLICATE (*Zeichenfolge, Anzahl*)
REPLACE (*string_exp1, string_exp2, string_exp3*)	STRTRAN (*Zeichenfolge1, Zeichenfolge2, Zeichenfolge3*)
RIGHT (*string_exp, count*)	
RTRIM (*string_exp*)	
SOUNDEX (*string_exp*)	
SPACE (*count*)	
SUBSTRING (*string_exp, start, length*)	SUBSTR (*Zeichenfolge, Beginn, Länge*)
UCASE (*string_exp*)	UPPER (*Zeichenfolge*)

Numerische Funktionen

In der folgenden Tabelle sind die numerischen ODBC-Funktionen (numeric functions) zusammengestellt, die der Visual FoxPro-ODBC-Treiber unterstützt. Für die Fälle, in denen sich für gleiche Funktionen die Visual FoxPro-Grammatik und die ODBC-Syntax unterscheiden, ist jeweils das Visual FoxPro-Gegenstück aufgeführt.

ODBC-Grammatik	**Visual FoxPro-Grammatik**
ABS (*numeric_exp*)	
ACOS (*float_exp*)	
ASIN (*float_exp*)	
ATAN (*float_exp*)	
ATAN2 (*float_exp1, float_exp2*)	ATN2(*Zahl1, Zahl2*)
CEILING (*numeric_exp*)	
COS (*float_exp*)	
COT (*float_exp*)	
DEGREES (*numeric_exp*)	RTOD (*numerischer_Ausdruck*)
EXP (*float_exp*)	
FLOOR (*numeric_exp*)	
LOG (*float_exp*)	
LOG10 (*float_exp*)	
MOD (*integer_exp1, integer_exp2*)	
PI ( )	
RADIANS (*numeric_exp*)	DTOR (*numerischer_Ausdruck*)
RAND (*[integer_exp]*)	
ROUND (*numeric_exp, integer_exp*)	
SIGN (*numeric_exp*)	
SIN (*float_exp*)	
SQRT (*float_exp*)	
TAN (*float_exp*)	

Die folgenden numerischen Funktionen werden nicht unterstützt:

POWER (*numeric_exp, integer_exp*)

TRUNCATE (*numeric_exp, integer_exp*)

Zeit- und Datumsfunktionen

In der folgenden Tabelle sind die ODBC-Zeit- und -Datumsfunktionen (time and date functions) zusammengestellt, die der Visual FoxPro-ODBC-Treiber unterstützt. Für die Fälle, in denen sich für gleiche Funktionen die Visual FoxPro-Grammatik und die ODBC-Syntax unterscheiden, ist jeweils das Visual FoxPro-Gegenstück aufgeführt.

ODBC-Grammatik	**Visual FoxPro-Grammatik**
CURDATE( )	DATE( )
CURTIME( )	TIME( )
DAYNAME(*date_exp*)	CDOW(*Datumsausdruck*)
DAYOFMONTH(*date_exp*)	DAY( )
HOUR(*time_exp*)	
MINUTE(*time_exp*)	
MONTH(*time_exp*)	
MONTHNAME(*date_exp*)	CMONTH(*Datumsausdruck*)
NOW( )	DATETIME( )
SECOND(*time_exp*)	SEC(*Zeitausdruck*)
WEEK(*date_exp*)	
YEAR(*date_exp*)	

Die folgenden Zeit- und Datumsfunktionen werden nicht unterstützt:

DAYOFYEAR (*date_exp*)

QUARTER (*date_exp*)

TIMESTAMPADD (*interval, integer_exp, timestamp_exp*)

TIMESTAMPDIFF (*interval, timestamp_exp1, timestamp_exp2*)

ODBC-Escape-Sequenzen

Der Treiber unterstützt für Datums- und TIMESTAMP-Daten auch die ODBC-Escape-Sequenzen. Die Syntax einer Escape-Klausel lautet wie folgt:

--(*vendor(Microsoft),product(ODBC) d '*Wert***' *)**—
--(*vendor(Microsoft),product(ODBC) ts '*Wert***' *)**—

In dieser Syntax gibt **d** an, daß *Wert* ein Datum im Format "JJJJ-MM-TT" ("yyyy-mm-dd") ist; und **ts** gibt an, daß *Wert* ein TIMESTAMP-Wert im Format "JJJJ-MM-TT HH:MM:SS[.f…]" ("yyyy-mm-dd hh:mm:ss[.f…]") ist. Die Kurzsyntax (shorthand syntax) für Datums- und TIMESTAMP-Werte lautet wie folgt:

{**d '***Wert***'**}
{**ts '***Wert***'**}

Zum Beispiel wird in den beiden folgenden Anweisungen die Tabelle ALLTYPES aktualisiert, indem in einem unterstützten SQL UPDATE-Befehl je ein Datums- bzw. TIMESTAMP-Wert in der Kurzsyntax verwendet wird:

```
UPDATE alltypes
   SET DAT_COL={d'1968-04-28'}
   WHERE KEY=111

UPDATE alltypes
   SET DTI_COL={ts'1968-04-28 12:00:00'}
   WHERE KEY=111
```

Weitere Informationen zu den Escape-Sequenzen (escape sequences) finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

Unterstützte SET-Befehle

Ihre Anwendung kann die folgenden Visual FoxPro-SET-Befehle an eine Datenquelle senden:

SET ANSI

SET BLOCKSIZE

SET COLLATE

SET DELETED

SET EXACT

SET EXCLUSIVE

SET NULL

SET PATH

SET REPROCESS

SET UNIQUE

Thread-Unterstützung

Der Visual FoxPro-ODBC-Treiber garantiert die Integrität der Threads. Zugriffe auf Umgebungskennungen (HENV = environment handles), Verbindungskennungen (HDBC = connection handles) und Anweisungskennungen (HSTMT = statement handles) werden in geeignete Semaphore gehüllt, um andere Prozesse daran zu hindern, auf die internen Datenstrukturen des Treibers zuzugreifen und diese möglicherweise zu ändern.

In einer Anwendung, die mit mehreren Threads arbeitet, können Sie eine Funktion, die synchron auf einer HSTMT läuft, abbrechen, indem Sie **SQLCancel** über einen eigenen Thread aufrufen.

Wenn Sie mit kontinuierlichem Lesen (progressive fetching) arbeiten, verwendet der Treiber einen eigenen Thread, um die Daten zu lesen. Damit für eine Datenquelle kontinuierliches Lesen verwendet wird, müssen Sie entweder im Dialogfeld **ODBC Visual FoxPro-Setup** das Kontrollkästchen **Daten im Hintergrund lesen** aktivieren oder in Ihrer Verbindungszeichenfolge (connection string) das Attributschlüsselwort **BackgroundFetch** verwenden. Informationen zu den Attributschlüsselwörtern einer Verbindungszeichenfolge finden Sie unter Verwenden von Verbindungszeichenfolgen.

Weitere Informationen zu Threads und **SQLCancel** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

Fehlerbehebung

In den folgenden Abschnitten wird erläutert, wie das Leistungsverhalten verbessert und Probleme behoben werden können, die bei der Verwendung des Visual FoxPro-ODBC-Treibers auftreten können.

Zugreifen auf Ansichten mit Parametern

Sie können mit dem Treiber nicht auf Ansichten mit Parametern (parameterized views) zugreifen, die in einer Visual FoxPro-Datenbank gespeichert sind. Eine Ansicht mit Parametern bewirkt, daß in ihrer SQL SELECT-Anweisung eine WHERE-Klausel erstellt wird, die die herunterzuladenden Datensätze auf die Datensätze beschränkt, die den Bedingungen der WHERE-Klausel entsprechen. Für diese Bedingungen werden die Werte verwendet, die für die Parameter der Ansicht angegeben sind. Da der Treiber das Übergeben von Parametern an eine Ansicht nicht unterstützt, ist es nicht möglich, auf Ansichten mit Parametern zuzugreifen.

Der Wert eines Parameters kann:

- Zur Laufzeit bereitgestellt werden.
- Programmgesteuert an eine Ansicht übergeben werden.

Zugreifen auf Remote-Ansichten

Sie können mit dem Treiber nicht auf Remote-Ansichten einer Visual FoxPro-Datenbank zugreifen. Remote-Ansichten sind Ansichten, die entweder auf Nicht-FoxPro-Daten oder eine Kombination aus FoxPro- und Nicht-FoxPro-Daten zugreifen. Wenn Sie auf Remote-Ansichten zugreifen möchten, müssen Sie Visual FoxPro verwenden.

Löschen von Datensätzen

Sie können den Treiber zwar verwenden, um Datensätze zum Löschen zu markieren, Sie können ihn aber nicht verwenden, Datensätze endgültig aus einer Datenbank zu löschen. Wenn Sie Datensätze endgültig aus einer Tabelle löschen möchten, müssen Sie Visual FoxPro verwenden.

Verbessern des Leistungsverhaltens mit Lesen im Hintergrund

Sie können das Leistungsverhalten für umfangreiche Lesevorgänge verbessern, indem Sie die Treiberfähigkeit nutzen, Daten im Hintergrund lesen zu können. Beim Lesen im Hintergrund wird ein eigener Thread verwendet, um Daten zu lesen, die über eine bestimmte Datenquelle angefordert werden.

Sie können auf zwei Arten für eine Datenquelle erreichen, daß mit Lesen im Hintergrund (background fetching) gearbeitet wird:

- Aktivieren Sie im Dialogfeld **ODBC Visual FoxPro-Setup** das Kontrollkästchen **Daten im Hintergrund lesen**.
- Verwenden Sie in Ihrer Verbindungszeichenfolge das Attributschlüsselwort **BackgroundFetch**.

Informationen zu den Attributschlüsselwörtern einer Verbindungszeichenfolge finden Sie unter Verwenden von Verbindungszeichenfolgen.

Aktualisieren mehrstufiger Ansichten

Eine mehrstufige Ansicht (multitiered view) ist eine Ansicht, die nicht auf Basistabellen beruht, sondern auf einer oder mehreren Ansichten. Wenn Sie in einer mehrstufigen Ansicht Daten aktualisieren, werden die Aktualisierungen nur eine Ebene nach unten gereicht bis zu der Ansicht, auf der die oberste Ansicht basiert. Die Basistabellen werden dann nicht aktualisiert.

Verwenden von DDL in gespeicherten Prozeduren

In gespeicherten Visual FoxPro-Prozeduren können Sie keine DDL-Anweisungen (DDL = Data Definition Language), wie z.B. CREATE TABLE oder ALTER TABLE, verwenden.

Informationen zu den Sprachelementen, die Sie in gespeicherten Prozeduren verwenden können, finden Sie unter Unterstützung der Visual FoxPro-Sprache in Regeln, Triggern, Standardwerten und gespeicherten Prozeduren.

Verwenden positionierter Aktualisierungen

Der Treiber unterstützt keine positionierten Aktualisierungen (positioned updates). Verwenden Sie eine SQL WHERE-Klausel, um die Zeilen anzugeben, die Sie aktualisieren möchten.

Verwenden des Befehls "SET ANSI"

Wenn Sie Visual FoxPro-Entwickler sind, sollten Sie folgendes wissen: für den Treiber lautet die SET ANSI-Standardeinstellung ON, während sie für Visual FoxPro OFF lautet. Die Standardeinstellung ON für SET ANSI ermöglicht es Visual FoxPro-Datenquellen, sich so zu verhalten wie andere ODBC-Datenquellen, die normalerweise exakte Vergleiche ausführen. Sie können die Standardeinstellung ändern. Weitere Informationen finden Sie unter SET ANSI.

Überblick: Fehlermeldungen

Wenn ein Fehler auftritt, gibt der Visual FoxPro-Treiber folgende Informationen zurück:

- Die treibereigene Nummer und Meldung des Fehlers.
- Den SQL-ZUSTAND (SQLSTATE, ein ODBC-Fehlercode) und die SQL-Meldung des Fehlers.

Sie können auf diese Fehlerinformationen zugreifen, indem Sie **SQLError** aufrufen.

Treibereigene Fehler (native errors)

Für Fehler, die in der Datenquelle auftreten, gibt der Visual FoxPro-Treiber seine eigene Nummer und Meldung des Fehlers zurück. Eine Liste dieser treibereigenen Fehlernummern finden Sie unter Eigene Fehlermeldungen des Visual FoxPro-ODBC-Treibers.

SQLSTATE (ODBC-Fehlercodes)

Für Fehler, die vom Visual FoxPro-Treiber entdeckt und zurückgegeben werden, bildet der Treiber seine von ihm zurückgegebene eigene Fehlernummer auf den entsprechenden SQLSTATE-Wert ab. Gibt es zu der treibereigenen Fehlernummer keinen ODBC-Fehlercode, auf den die Nummer abgebildet werden kann, gibt der Visual FoxPro-Treiber SQLSTATE S1000 ("Allgemeiner Fehler") zurück.

Eine Liste mit den SQLSTATE-Werten, die der Visual FoxPro-ODBC-Treiber für die entsprechenden Visual FoxPro-Fehler meldet, finden Sie unter ODBC-Fehlercodes.

Syntax

Eine Fehlermeldung hat folgendes Format:

[*Hersteller***][***ODBC_Komponente***]***Fehlermeldung*

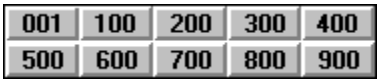
Die in eckigen Klammern ([ ]) stehenden Präfixe geben die Fehlerquelle entsprechend der Übersicht der folgenden Tabelle an.

Fehlerquelle	Präfix	Wert
Treiber-Manager	[Hersteller] [ODBC_Komponente] [Datenquelle]	[Microsoft] [ODBC-Treiber-Manager] Entfällt
Visual FoxPro-Treiber	[Hersteller] [ODBC_Komponente] [Datenquelle]	[Microsoft] [ODBC Visual FoxPro-Treiber] Entfällt

Wenn der Visual FoxPro-ODBC-Treiber z.B. die Datei EMPLOYEE.DBF nicht finden konnte, gibt er folgende Fehlermeldung zurück:

"[Microsoft][ODBC Visual FoxPro-Treiber]Datei 'employee.dbf' existiert nicht."

Eigene Fehlermeldungen des Visual FoxPro-ODBC-Treibers



In der folgenden Tabelle sind die Fehlermeldungen zusammengestellt, die zu dem Visual FoxPro-ODBC-Treiber gehören.

001

1	Option ist nicht verfügbar.
2	Eingabe-/Ausgabefehler.
3	Es wurde keine freie Kennung (handle) gefunden.
5	Kennung (handle) nicht zugewiesen.
14	???????
97	???????
99	Vorgang abgebrochen.

100

100	Zu viele Dateien offen.
101	Datei kann nicht geöffnet werden.
102	Datei kann nicht erstellt werden.
105	Fehler beim Schreiben der Datei.
107	Ungültige Schlüssellänge.
109	Satz außerhalb des Bereichs.
110	Satz nicht im Index.
111	Ungültiger Dateideskriptor.
113	Datei ist nicht geöffnet.
114	Es ist nicht genügend Plattenspeicherplatz für *Wert* verfügbar.
115	Ungültige Operation für den Cursor.
118	Indexdatei stimmt nicht mit Tabelle überein.
119	Es ist keine Tabelle geöffnet.
120	Datei ist nicht vorhanden.
121	Datei ist bereits vorhanden.
122	Es wurde kein Index für die Tabelle festgelegt.
123	Keine Tabelle.
125	Indexausdruck überschreitet maximal zulässige Länge.
127	Sie müssen einen logischen Ausdruck für die FOR- oder WHILE-Klausel angeben.
128	Kein numerischer Ausdruck.
129	Variable wurde nicht gefunden.
132	Datei wird bereits benutzt.
133	Index stimmt nicht mit Tabelle überein. Löschen Sie die Indexdatei, und erstellen Sie den Index erneut.
135	Dateiende erreicht.
136	Dateianfang erreicht.

137	Alias wurde nicht gefunden.
139	Sie müssen einen logischen Ausdruck für FILTER angeben.
142	Zyklische Beziehung.
143	Keine zu kopierenden Felder gefunden.
144	Der LOCATE-Befehl muß vor dem CONTINUE-Befehl erteilt werden.
145	Muß ein Zeichen- oder numerisches Schlüsselfeld sein.
146	Es ist nicht möglich, in schreibgeschützte Datei zu schreiben.
147	Zieltabelle wird bereits in einer Beziehung benutzt.
148	Der Ausdruck wurde erneut eingegeben, während der Filter ausgeführt wird.
149	Nicht genug Speicher für Zwischenspeicher.
150	Nicht genug Speicher für Dateitabelle.
155	Unzulässiger buffdirty-Aufruf.
156	Feldnamen nicht eindeutig.
158	Keine zu verarbeitenden Felder gefunden.
159	Numerischer Überlauf. Es sind Daten verlorengegangen.
162	Prozedur '*Wert*' wurde nicht gefunden.
165	*Wert* steht nicht in Beziehung zum aktuellen Arbeitsbereich.
170	Variable '*Wert*' wurde nicht gefunden.
171	Datei *Wert* kann nicht geöffnet werden.
173	Datei '*Wert*' ist nicht vorhanden.
174	'*Wert*' ist keine Speichervariable.
175	'*Wert*' ist keine Dateivariable.
176	'*Wert*' ist kein Array (Datenfeld).
177	Alias '*Wert*' wurde nicht gefunden.
180	Datei wurde nicht über den LOAD-Befehl in den Speicher geladen.
182	Es ist nicht genug Speicher vorhanden, um diese Operation abschließen zu können.

200

200	Syntaxfehler.
201	Zu viele Namen benutzt.
202	Programm ist zu groß.
203	Zu viele Speichervariablen.
205	Verschachtelungsfehler.
206	Rekursive Makrodefinition.
209	Zeile ist zu lang.
210	Maximale Verschachtelungstiefe für DO überschritten.
211	Es fehlt eine IF | ELSE | ENDIF-Anweisung.
212	Strukturverschachtelung ist zu tief.

213	In der FOR...ENDFOR oder DO CASE...ENDCASE-Befehlsstruktur fehlt ein Schlüsselwort.
219	Befehl enthält unbekannten Ausdruck oder unbekanntes Schlüsselwort.
221	Befehl ist nicht vollständig.
222	Unbekannter Befehl.
224	Unzulässiger Verweis auf Datenfeldindex.
227	Fehlender Ausdruck.
228	Tabellennummer ist ungültig.
229	Zu wenige Argumente.
230	Zu viele Argumente.
233	Anweisung ist im interaktiven Modus nicht zulässig.
234	Datenfeldindex außerhalb des gültigen Bereichs.
236	Sie können RESUME erst verwenden, wenn das Programm unterbrochen wurde.
238	Es wurde keine PARAMETER-Anweisung gefunden.
239	Es müssen zusätzliche Parameter angegeben werden.
240	Kein Zeichenausdruck.
250	Es sind zu viele PROCEDURE-Befehle aktiv.
252	Kompilierter Code für diese Zeile ist zu lang.
257	Zeichenfolge für Taste ist zu lang.
291	Der mit ASIN() benutzte Ausdruck ist außerhalb des gültigen Bereichs.
292	Das Argument für LOG10() darf nicht 0 oder negativ sein.
293	Der mit ACOS() benutzte Ausdruck ist außerhalb des gültigen Bereichs.
294	Die Datei FOXUSER.DBF ist ungültig.
295	Ungültiger Pfad oder Dateiname.
296	Fehler beim Lesen der Ressource.
297	Befehl ist nur im interaktiven Modus zulässig.

300

301	Operator-/Operand-Typ stimmt nicht überein.
302	Datentyp stimmt nicht überein.
305	Ausdruck liefert ungültigen Wert.
307	Es kann nicht durch 0 dividiert werden.
308	Stapelüberlauf.
337	Der PRINTJOB-Befehl kann nicht verschachtelt werden.

400

406	Drucker ist nicht bereit.
407	Ungültiges Argument für die SET-Funktion.
410	Temporäre Arbeitsdateien können nicht erstellt werden.
423	Fehler beim Erstellen des OLE-Objekts.

424 Fehler beim Kopieren des OLE-Objekts in die Zwischenablage.

462 *Wert*: interner Übereinstimmungsfehler.

465 Interner Übereinstimmungsfehler bei SQL-Pass-Through.

466 Verbindungskennung (connection handle) ist ungültig.

467 Die Eigenschaft ist für lokale Cursor ungültig.

468 Ungültige Eigenschaft für einen Tabellen-Cursor.

469 Eigenschaftenwert ist außerhalb des gültigen Bereichs.

470 Falscher Name für Eigenschaft.

471 Falsches Spaltenformat.

473 Umgebungsebenen-Eigenschaft ist ungültig.

474 Es wurde ein unzulässiger Aufruf während der Ausführung einer SQLEXEC()-Sequenz durchgeführt.

479 Ungültiger Name für Aktualisierungsspalte: *Wert*\.

489 Objektfelder können nicht in der WHERE-Bedingung einer Aktualisierungsanweisung benutzt werden. Ändern Sie die WhereType-Eigenschaft der Ansicht.

491 Es wurden keine Aktualisierungstabellen angegeben. Verwenden Sie die Tables-Eigenschaft des Cursors.

492 Für die Aktualisierungstabelle *Wert*\ sind keine Schlüsselspalten angegeben. Verwenden Sie die KeyFieldList-Eigenschaft des Cursors.

493 Fehlender SQL-Parameter.

494 Ansichtsdefinition ist verändert worden.

495 Warnung: Der durch die KeyFieldList-Eigenschaft definierte Schlüssel der Tabelle *Wert* ist nicht eindeutig.

498 SQL-SELECT-Anweisung ist ungültig.

499 SQL-Parameter *Wert* ist ungültig.

500

502 Es kann nicht in den Datensatz geschrieben werden, weil er bereits benutzt wird.

503 Datei kann nicht gesperrt werden.

508 Fehler bei der OLE-Initialisierung.

520 Es ist keine Datenbank geöffnet oder als aktuelle Datenbank festgelegt.

522 Interner Connectivity-Übereinstimmungsfehler.

523 Ausführung wurde durch den Benutzer abgebrochen.

525 Funktion wird für Remote-Tabellen nicht unterstützt.

526 Connectivity-Fehler: *Wert*

527 Die ODBC-Bibliothek (ODBC32.DLL) kann nicht geladen werden.

528 ODBC-Einsprungpunkt fehlt, *Wert*.

530 Lesen abgebrochen; die Remote-Tabelle ist

geschlossen.

532 Typkonvertierung ist nicht unterstützt.

533 Diese Eigenschaft ist schreibgeschützt.

536 Funktion wird für systemeigene Tabellen nicht unterstützt.

538 Es wird gerade eine gespeicherte Prozedur ausgeführt.

540 Sitzungsnummer ist ungültig.

541 Verbindung '*Wert*' ist belegt.

542 Einige Basistabellenfelder sind verändert worden und stimmen nicht mehr mit den Ansichtsfeldern überein. Die Eigenschaften der Ansichtsfelder können nicht festgelegt werden.

543 Die Typumwandlung, die wegen der DataType-Eigenschaft des Feldes ' *Wert* ' erforderlich ist, ist unzulässig.

544 Die DataType-Eigenschaft des Feldes '*Wert* ' ist unzulässig.

545 Der Tabellenpuffer für den Aliasnamen *Wert*\ enthält Änderungen, die noch nicht übernommen wurden.

546 Tabelle kann nicht während der Ausführung eines tabellengebundenen Ausdrucks geschlossen werden.

547 Eine leere Zeile einer Ansicht kann nicht in die Basistabelle(n) der Ansicht eingefügt werden.

548 Für die Tabelle *Wert* ist mindestens ein nichtstrukturierter Index geöffnet. Schließen Sie diesen Index, bevor Sie den BEGIN TRANSACTION-Befehl erneut ausgeben.

549 Die Datensitzung Nr. *Wert* kann nicht freigegeben werden, da sie eine oder mehrere nicht abgeschlossene Transaktionen besitzt.

550 Interner .DBC-Übereinstimmungsfehler.

557 Die Datenbank muß exklusiv geöffnet werden.

559 Eigenschaft wurde nicht gefunden.

560 Eigenschaftswert ist ungültig.

561 Datenbank ist ungültig. Bitte Gültigkeitsprüfung durchführen.

562 Objekt *Wert* konnte nicht in der Datenbank gefunden werden.

563 Ansicht *Wert* konnte nicht in der aktuellen Datenbank gefunden werden.

566 Solange die Tabellen einer Datenbank verwendet werden, kann für diese Datenbank kein PACK-Befehl ausgeführt werden.

567 Die Primärschlüssel-Eigenschaft ist ungültig; bitte überprüfen Sie die Gültigkeit der Datenbank.

570 Datenbank ist schreibgeschützt.

571 Der Name *Wert* wird bereits für ein anderes Objekt verwendet. Bitte wählen Sie einen unterschiedlichen Namen.

575	Objektname ist ungültig.
577	Auf die Tabelle *Wert* wird in einer Beziehung verwiesen.
578	Ungültiger Datenbank-Tabellenname.
579	Der Befehl kann nicht für eine Tabelle ausgegeben werden, für die es Cursor im Tabellenpufferungsmodus gibt.
580	Funktion wird nur für DBC-Tabellen unterstützt.
581	Feld *Wert* akzeptiert keine Nullwerte.
583	Die Gültigkeitsregel des Datensatzes wurde verletzt.
585	Aktualisierungskonflikt. Verwenden Sie entweder TABLEUPDATE() mit dem Parameter lErzwingen, um die Aktualisierung zu übernehmen (commit), oder TABLEREVERT(), um die Aktualisierung zurückzusetzen (roll back).
586	Funktion erfordert zeilen- oder tabellenweise Zwischenspeicherung.
587	Unzulässige Verschachtelung bei OLDVAL() oder CURVAL().
589	Tabellen- oder Zeilenpufferung erfordert, daß SET MULTILOCKS auf ON eingestellt ist.
590	BEGIN TRANSACTION-Befehl fehlgeschlagen. Verschachtelungsebene ist zu tief.
591	END TRANSACTION-Befehl kann nicht ohne einen entsprechenden BEGIN TRANSACTION-Befehl erteilt werden.
592	ROLLBACK-Befehl kann nicht ohne einen entsprechenden BEGIN TRANSACTION-Befehl erteilt werden.
593	Befehl kann nicht innerhalb einer Transaktion erteilt werden.
594	Eine Dateisperre kann nicht in einer Transaktion erfolgen, wenn bereits Sperrungen auf Datensatzebene erfolgt sind.
596	Die Tabellenpufferung ist nicht aktiviert.
597	Ansichten erfordern entweder DB_BUFOPTROW oder DB_BUFOPTTABLE.
598	Regel- und Trigger-Code muß den Transaktionsgebrauch ausgleichen.
599	Die Datensitzung Nr. *Wert* wurde gezwungen, alle Transaktionen zurückzunehmen (ROLLBACK), um eine Blockierung zu verhindern.

600

601	Aliasname wird bereits benutzt.
602	Operation ist unzulässig für ein Memo-, Objekt- oder Bildfeld.
612	Menü oder Menüeintrag ist nicht definiert.
618	Menü ist nicht mit DEFINE MENU definiert worden.
624	Menütitel ist nicht mit DEFINE PAD definiert worden.

625	Menü ist nicht mit DEFINE POPUP definiert worden.
631	Datenfeld-Dimensionen sind ungültig.
637	Datei muß exklusiv geöffnet sein, um die Memodatei zu konvertieren.
638	Feld muß ein Memofeld sein.
649	Es wurde kein passender PRINTJOB-Befehl für diesen Befehl erteilt.
651	CANCEL oder SUSPEND ist nicht zugelassen.
659	Die Tabelle besitzt Memofelder, die nicht konvertiert werden können, weil die Tabelle schreibgeschützt geöffnet wurde.
683	Indexname wurde nicht gefunden.

700

700	Datensatz wird von einem anderen Benutzer benutzt.
701	Datei muß exklusiv geöffnet sein.
702	Datei wird von einem anderen Benutzer benutzt.
703	Der Datensatz ist nicht gesperrt.
705	Zugriff auf Datei wurde verweigert.
706	.IDX-Dateien können nicht absteigend sortiert werden.
707	Strukturierte .CDX-Datei wurde nicht gefunden.
708	Datei ist bereits in einem anderen Arbeitsbereich geöffnet.
712	Feldname ist bereits vorhanden oder ungültig.
714	Fenster '*Wert*' wurde nicht definiert.
718	Datei ist schreibgeschützt.
722	Präprozessorausdruck ist ungültig.
734	Eigenschaft *Wert* wurde nicht gefunden.
737	*Wert* ist eine Methode, Ereignis oder Objekt.
738	Eigenschaft *Wert* ist keine Methode oder Ereignis.
740	*Wert* ist eine schreibgeschützte Eigenschaft.
748	Diese Datei ist nicht mit der aktuellen Version von Visual FoxPro kompatibel.
750	Datei wurde in einer späteren als der aktuellen Version von Visual FoxPro erstellt.
763	Eigenschaft *Wert* ist bereits vorhanden.
773	Datenbankobjekttyp ist ungültig.
784	Dieses Objekt wurde aus einer Basisklasse abgeleitet und hat keine Oberklasse (ParentClass).

800

802	SQL: Tabelle kann nicht gefunden werden.
872	Zu viele Spalten.
879	Kein Primärschlüssel.
884	Die Eindeutigkeit des Indexes *Wert* wird verletzt.
885	Nur strukturierte Indizes können als potentieller Index

definiert werden.

886 Index akzeptiert keine Nullwerte.

887 Unzulässige Rekursion bei der Regelauswertung.

888 Indexname ist zu lang.

900

901 Wert, Typ oder Anzahl der Funktionsargumente ist ungültig.

902 Fehler bei der Auswertung eines Ausdrucks.

903 Zeichenfolge ist zu lang.

904 ** oder ^ Domänenfehler.

905 LOG(): Null oder negative Zahl als Argument benutzt.

906 SQRT()-Argument darf nicht negativ sein.

912 Operation ist für ein Objektfeld nicht zulässig.

914 Codeseitennummer ist ungültig.

915 Sortierfolge '*Wert*' wurde nicht gefunden.

918 Dateiname ist zu lang.

922 Volume existiert nicht.

923 Objekt *Wert* wurde nicht gefunden.

924 *Wert* ist kein Objekt.

925 Unbekanntes Element *Wert*.

928 Diese Anweisung ist nur innerhalb einer Klassendefinition zulässig.

929 *Wert* kann nur innerhalb einer Methode benutzt werden.

930 *Wert* kann nicht erneut definiert werden.

931 Anweisung befindet sich nicht in einer Prozedur.

934 Diese Anweisung ist nur innerhalb einer Methode zulässig.

935 Das aktuelle Objekt erbt nichts von der Klasse *Wert*.

937 Prozedurdatei '*Wert*' wurde nicht gefunden.

938 Objekt ist nicht in einem *Wert*-Container enthalten.

939 WITH/ENDWITH-Fehler.

940 Ausdruck ist nicht außerhalb von WITH/ENDWITH zulässig.

941 Fehlercode ist ungültig.

942 Objekte können nicht Datenfeldern zugewiesen werden.

943 Element *Wert* kann nicht zu einem Objekt ausgewertet werden.

945 Das aktuelle Objekt ist freigegeben worden.

947 Ausdruck ist zu komplex.

951 Es kann kein Objekt freigegeben werden, das gerade benutzt wird.

955 WIN.INI/Registrierung ist beschädigt.

957 Fehler beim Zugriff auf den Drucker-Spooler.

959 Ungültige Koordinaten.

960	Ungültige erneute Definition der Variablen *Wert*.
971	Es kann nicht kompiliert werden, solange der aktuelle COMPILE-Befehl nicht abgeschlossen ist.
972	Datenfeld *Wert* wird benutzt.
974	Datenfelder können nicht Datenfeldelementen zugewiesen werden.
976	Rückverweis kann nicht aufgelöst werden.
988	Währungswert ist außerhalb des gültigen Bereichs.
990	Abbrechen.
999	Funktion ist nicht implementiert.

ODBC-Fehlercodes

In der folgenden Tabelle ist zusammengestellt, welcher Visual FoxPro-Fehlercode auf welchen ODBC-Fehlercode (SQLSTATE-Wert) abgebildet wird. Die abgebildeten SQLSTATE-Werte werden von den Funktionen **SQLExecDirect** und **SQLPrepare** zurückgegeben. Es werden keine SQLSTATE-Werte anderer ODBC-API-Funktionen abgebildet, da **SQLExecDirect** und **SQLPrepare** die einzigen Funktionen sind, die auf die Visual FoxPro-Engine zugreifen.

Weitere Informationen zu den ODBC-Fehlercodes (ODBC Error Codes) finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLSTATE	Visual FoxPro-Fehlercode
S1001	149 150 182 202 308
1004	159
37000	132 200 219 221 222 227 229 230 498 499 713 901
22005	301 302
22012	307
23000	581 583 884 886 988
S0001	121 571
S0002	173 120 123 295 562 563 802
S0012	683
S0021	156 712

S0022	158
	806
S1000	100
	101
	102
	105
	107
	109
	110
	111
	113
	114
	115
	118
	119
	125
	133
	135
	136
	137
	145
	146
	171
	173
	177
	201
	205
	239
	240
	252
	257
	296
	305
	407
	410
	462
	502
	503
	520
	538
	550
	561
	567
	570
	575
	578
	580
	585
	602
	702
	705
	707
	708
	718
	750

872
879
887
888
912
914
915
918
922
923
947
976
999

Unterstützung der Visual FoxPro-Sprache in Regeln, Triggern, Standardwerten und gespeicherten Prozeduren

Sie können den Visual FoxPro-ODBC-Treiber nicht dazu verwenden, Visual FoxPro-Regeln, -Trigger, -Standardwerte oder gespeicherte Prozeduren zu erstellen. Ihre Anwendung kann aber, wenn sie Visual FoxPro-Daten, die in einer Datenbank gespeichert sind, hinzufügt, aktualisiert oder löscht, auf vorhandene Regeln, Trigger, Standardwerte oder gespeicherte Prozeduren zurückgreifen.

In der folgenden Tabelle sind die Visual FoxPro-Befehle und -Funktionen zusammengestellt, die vom Visual FoxPro-ODBC-Treiber unterstützt werden, wenn diese Befehle oder Funktionen in Regeln, Triggern, Standardwerten oder gespeicherten Prozeduren vorkommen.

Wenn Ihre Anwendung auf Daten zugreift, deren Regeln, Trigger, Standardwerte oder gespeicherte Prozeduren irgendwelche anderen Visual FoxPro-Befehle oder -Funktionen aufrufen, meldet der Treiber einen Fehler. Eine Liste der Befehle und Funktionen, die der Treiber nicht unterstützt, finden Sie unter Nichtunterstützte Visual FoxPro-Befehle und -Funktionen.

Tip Wenn Sie in Ihre Regeln, Trigger oder gespeicherten Prozeduren bedingten Code einfügen möchten, der die Befehle festlegt, die ausgeführt werden sollen, wenn die Regeln, Trigger oder gespeicherten Prozeduren vom Treiber aufgerufen werden, können Sie die Funktion VERSION( ) verwenden. Die Funktion VERSION( ) gibt "Visual FoxPro-ODBC-Treiber <*Version*>" zurück, wenn sie vom Treiber aufgerufen wird.

Die Visual FoxPro-Befehle und -Funktionen, die in Regeln, Triggern, Standardwerten und gespeicherten Prozeduren unterstützt werden

$-Operator	%-Operator	&-Befehl
&&-Befehl	*-Befehl	=-Befehl

A

ABS( )-Funktion	ACOPY( )-Funktion	ACOS( )-Funktion
ADATABASES( )-Funktion	ADBOBJECTS( )-Funktion	ADEL( )-Funktion
AELEMENT( )-Funktion	AERROR( )-Funktion	AFIELDS( )-Funktion
AINS( )-Funktion	ALEN( )-Funktion	ALIAS( )-Funktion
ALLTRIM( )-Funktion	APPEND-Befehl	APPEND FROM-Befehl
APPEND FROM ARRAY-Befehl	APPEND GENERAL-Befehl	APPEND MEMO-Befehl
ASC( )-Funktion	ASCAN( )-Funktion	ASIN( )-Funktion
ASORT( )-Funktion	ASUBSCRIPT( )-Funktion	AT( )-Funktion
ATAN( )-Funktion	ATC( )-Funktion	ATCLINE( )-Funktion
ATLINE( )-Funktion	ATN2( )-Funktion	AUSED( )-Funktion
AVERAGE-Befehl		

B

BEGIN TRANSACTION-Befehl	BETWEEN( )-Funktion	BITAND( )-Funktion
BITCLEAR( )-Funktion	BITLSHIFT( )-Funktion	BITNOT( )-Funktion

C

D

E

M

MAX( )-Funktion	MDX( )-Funktion	MDY( )-Funktion
MEMLINES( )-Funktion	MESSAGE( )-Funktion	MIN( )-Funktion
MINUTE( )-Funktion	MLINE( )-Funktion	_MLINE-Systemvariable
MOD( )-Funktion	MONTH( )-Funktion	MTON( )-Funktion

N

NDX( )-Funktion	NORMALIZE( )-Funktion	NOTE-Befehl
NTOM( )-Funktion	NVL( )-Funktion	

O

OCCURS( )-Funktion	OLDVAL( )-Funktion	ON ERROR-Befehl
ON KEY-Befehl	ON( )-Funktion	OPEN DATABASE-Befehl
ORDER( )-Funktion	OS( )-Funktion	

P

PACK-Befehl	PAD( )-Funktion	PADL( )- | PADR( )- | PADC( )-Funktion
PARAMETERS-Befehl	PARAMET ERS( )-Funktion	PAYMENT( )-Funktion
PI( )-Funktion	PRIMARY( )-Funktion	PRIVATE-Befehl
PROCEDURE-Befehl	PROGRAM( )-Funktion	PROPER( )-Funktion
PUBLIC-Befehl	PV( )-Funktion	

Q

R

RAND( )-Funktion	RAT( )-Funktion	RATLINE( )-Funktion
RECALL-Befehl	RECCOUNT( )-Funktion	RECNO( )-Funktion
RECSIZE( )-Funktion	REFRESH( )-Funktion	REGIONAL-Befehl
RELATION( )-Funktion	REMOVE TABLE-Befehl	REPLACE-Befehl
REPLACE FROM ARRAY-Befehl	REPLICATE( )-Funktion	RETURN-Befehl
RIGHT( )-Funktion	RLOCK( )-Funktion	ROLLBACK-Befehl
ROUND( )-Funktion	RTOD( )-Funktion	RTRIM( )-Funktion

S

SCAN ... ENDSCAN-Befehl	SCATTER-Befehl	SEC( )-Funktion
SECONDS( )-Funktion	SEEK-Befehl	SEEK( )-Funktion
SELECT-Befehl	SELECT-SQL-Befehl	SELECT( )-Funktion

X

Y

Z

Nichtunterstützte Visual FoxPro-Befehle und -Funktionen

In der folgenden Tabelle sind die Visual FoxPro-Befehle und -Funktionen zusammengestellt, die zwar von Microsoft Visual FoxPro aber nicht vom Visual FoxPro-ODBC-Treiber unterstützt werden.

Wenn Ihre Anwendung auf Daten zugreift, deren Regeln, Trigger, Standardwerte oder gespeicherten Prozeduren diese Visual FoxPro-Befehle oder -Funktionen aufrufen, meldet der Treiber einen Fehler.

Anmerkung Eventuell möchten Sie in Visual FoxPro ein Programm schreiben, das in Ihrer Datenbank die Regeln, Trigger, Standardwerte und gespeicherten Prozeduren nach nichtunterstützten Befehlen und Funktionen durchsucht. Informationen zu einer Textdateiversion der folgenden Tabelle, die Sie für dieses Programm verwenden könnten, finden Sie in der Datei INFO.TXT, die mit dem Treiber geliefert wurde.

Nichtunterstützte Visual FoxPro-Befehle und -Funktionen

#DEFINE ... #UNDEF-Präprozessoranweisung	#IF ... #ENDIF-Präprozessoranweisung	#IFDEF | #IFNDEF-Präprozessoranweisung
#INCLUDE-Präprozessoranweisung	::-Geltungsbereich-Auflösungs-Operator	!-Befehl (siehe RUN- | !-Befehl)
?- | ??-Befehl	???-Befehl	\- | \\-Befehl
@ ... BOX-Befehl	@ ... CLASS-Befehl	@ ... CLEAR-Befehl
@ ... EDIT-Befehl - Bearbeitungsfelder	@ ... FILL-Befehl	@ ... GET
@ ... MENU-Befehl	@ ... PROMPT-Befehl	@ ... SAY-Befehl
@ ... SCROLL-Befehl	@ ... TO-Befehl	

A

ACCEPT-Befehl	ACLASS( )-Funktion	ACTIVATE MENU-Befehl
ACTIVATE POPUP-Befehl	ACTIVATE SCREEN-Befehl	ACTIVATE WINDOW-Befehl
ActivateCell Method	ADD CLASS-Befehl	ADIR( )-Funktion
AFONT( )-Funktion	AINSTANCE( )-Funktion	_ALIGNMENT System Memory Variable
AMEMBERS( )-Funktion	APRINTERS( )-Funktion	ASELOBJ( )-Funktion
ASSIST-Befehl		

B

BAR( )-Funktion	BARCOUNT( )-Funktion	BARPROMPT( )-Funktion
_BEAUTIFY System Memory Variable	_BOX System Memory Variable	BROWSE-Befehl
_BROWSER System Memory Variable	BUILD APP-Befehl	BUILD EXE-Befehl
BUILD PROJECT-Befehl	_BUILDER System Memory Variable	

C

_CALCVALUE System Memory Variable
CALL-Befehl
CANCEL-Befehl
CAPSLOCK( )-Funktion
CHANGE-Befehl
CHRSAW( )-Funktion
_CLIPTEXT System Memory Variable
CLOSE MEMO-Befehl
CNTBAR( )-Funktion
CNTPAD( )-Funktion
COL( )-Funktion
COMPILE-Befehl
COMPILE DATABASE-Befehl
COMPILE FORM-Befehl
COMPOBJ( )-Funktion
Container Object
Control Object
_CONVERTER System Memory Variable
COPY FILE-Befehl
CREATE CLASS-Befehl
CREATE CLASSLIB-Befehl
CREATE COLOR SET-Befehl
CREATE-Befehl
CREATE CONNECTION-Befehl
CREATE DATABASE-Befehl
CREATE FORM-Befehl
CREATE FROM-Befehl
CREATE LABEL-Befehl
CREATE MENU-Befehl
CREATE PROJECT-Befehl
CREATE QUERY-Befehl
CREATE REPORT-Befehl
CREATE SCREEN-Befehl
CREATE SQL VIEW-Befehl
CREATE TRIGGER-Befehl
CREATE VIEW-Befehl
CREATEOBJECT( )-Funktion
_CUROBJ System Memory Variable

D

_DBLCLICK System Memory Variable
DBSETPROP( )-Funktion
DDE-Funktions
DEACTIVATE MENU-Befehl
DEACTIVATE POPUP-Befehl
DEACTIVATE WINDOW-Befehl
DECLARE - DLL-Befehl
DEFINE BAR-Befehl
DEFINE BOX-Befehl
DEFINE CLASS-Befehl
DEFINE MENU-Befehl
DEFINE PAD-Befehl
DEFINE POPUP-Befehl
DEFINE WINDOW-Befehl
DELETE CONNECTION-Befehl
DELETE DATABASE-Befehl
DELETE TRIGGER-Befehl
DELETE VIEW-Befehl
_DIARYDATE System Memory Variable
DIR-Befehl
DIRECTORY-Befehl
DISPLAY-Befehl
DISPLAY CONNECTIONS-Befehl
DISPLAY DATABASE-Befehl
DISPLAY DLLS-Befehl
DISPLAY FILES-Befehl
DISPLAY MEMORY-Befehl
DISPLAY OBJECTS-Befehl
DISPLAY PROCEDURES-Befehl
DISPLAY STATUS-Befehl
DISPLAY STRUCTURE-Befehl
DISPLAY TABLES-Befehl
DISPLAY VIEWS-Befehl
DO FORM-Befehl

E

EDIT-Befehl
EJECT-Befehl
EJECT PAGE-Befehl
EXPORT-Befehl
EXTERNAL-Befehl

F

FILER-Befehl
FIND-Befehl
FKLABEL( )-Funktion
FKMAX( )-Funktion
FONTMETRIC( )-Funktion
_FOXDOC System Memory Variable
_FOXGRAPH System Memory Variable

G

_GENGRAPH System Memory Variable
_GENMENU System Memory Variable
_GENPD System Memory Variable
_GENSCRN System Memory Variable
_GENXTAB System Memory Variable
GETBAR( )-Funktion
GETCOLOR( )-Funktion
GETDIR( )-Funktion
GETFILE( )-Funktion
GETFONT( )-Funktion
GETOBJECT( )-Funktion
GETPAD( )-Funktion
GETPICT( )-Funktion
GETPRINTER( )-Funktion

H

HELP-Befehl
HIDE MENU-Befehl
HIDE POPUP-Befehl
HIDE WINDOW-Befehl

I

IMESTATUS( )-Funktion
IMPORT-Befehl
_INDENT System Memory Variable
INDEX ON-Befehl
INKEY( )-Funktion
INPUT-Befehl
INSERT-Befehl
INSMODE( )-Funktion
ISCOLOR( )-Funktion
ISMOUSE( )-Funktion

J

JOIN-Befehl

K

KEYBOARD-Befehl

L

LABEL-Befehl
LASTKEY( )-Funktion
LIST-Befehls
LIST CONNECTIONS-Befehl
_LMARGIN System Memory Variable
LOAD-Befehl
LOCFILE( )-Funktion

M

MCOL( )-Funktion
MDOWN( )-Funktion
MEMORY( )-Funktion
MENU-Befehl
MENU TO-Befehl
MENU( )-Funktion
MESSAGEBOX( )-Funktion
MODIFY CLASS-Befehl
MODIFY-Befehl-Befehl
MODIFY CONNECTION-Befehl
MODIFY DATABASE-Befehl
MODIFY FILE-Befehl
MODIFY FORM-Befehl
MODIFY GENERAL-Befehl
MODIFY LABEL-Befehl
MODIFY MEMO-Befehl
MODIFY MENU-Befehl
MODIFY PROCEDURE-Befehl
MODIFY PROJECT-Befehl
MODIFY QUERY-Befehl
MODIFY REPORT-Befehl
MODIFY SCREEN-Befehl
MODIFY STRUCTURE-Befehl
MODIFY VIEW-Befehl
MODIFY WINDOW-Befehl
MOUSE-Befehl
MOVE POPUP-Befehl
MOVE WINDOW-Befehl
MRKBAR( )-Funktion
MRKPAD( )-Funktion
MROW( )-Funktion
MWINDOW( )-Funktion

N

NUMLOCK( )-Funktion

O

OBJNUM( )-Funktion
OBJTOCLIENT( )-Funktion
OBJVAR( )-Funktion
ON APLABOUT-Befehl
ON BAR-Befehl
ON ESCAPE-Befehl
ON EXIT BAR-Befehl
ON EXIT MENU-Befehl
ON EXIT PAD-Befehl
ON EXIT POPUP-Befehl
ON KEY =-Befehl
ON KEY LABEL-Befehl
ON MACHELP-Befehl
ON PAD-Befehl
ON PAGE-Befehl
ON READERROR-Befehl
ON SELECTION BAR-Befehl
ON SELECTION MENU-Befehl
ON SELECTION PAD-Befehl
ON SELECTION POPUP-Befehl
ON SHUTDOWN-Befehl

P

PACK DATABASE-Befehl
_PADVANCE System Memory Variable
_PAGENO System Memory Variable
_PBPAGE System Memory Variable
PCOL( )-Funktion
_PCOLNO System Memory Variable
_PCOPIES System Memory Variable
_PDRIVER System Memory Variable
_PDSETUP System Memory Variable
_PECODE System Memory Variable
_PEJECT System Memory Variable
PEMSTATUS( )-Funktion
_PEPAGE System Memory Variable
_PLENGTH System Memory Variable
_PLINENO System Memory Variable
_PLOFFSET System
PLAY MACRO-Befehl
POP KEY-Befehl

Q

R

S

SET CLEAR-Befehl | SET CLOCK-Befehl | SET COLOR OF-Befehl
SET COLOR OF SCHEME-Befehl | SET COLOR SET-Befehl | SET COLOR TO-Befehl
SET COMPATIBLE-Befehl | SET CONFIRM-Befehl | SET CONSOLE-Befehl
SET CPCOMPILE | SET CPDIALOG | SET CURRENCY-Befehl
SET CURSOR-Befehl | SET DATASESSION-Befehl | SET DEBUG-Befehl
SET DECIMALS-Befehl | SET DELIMITERS-Befehl | SET DEVELOPMENT-Befehl
SET DEVICE-Befehl | SET DISPLAY-Befehl | SET DOHISTORY-Befehl
SET ECHO-Befehl | SET ESCAPE-Befehl | SET FORMAT-Befehl
SET FUNCTION-Befehl | SET HEADINGS-Befehl | SET HELP-Befehl
SET HELPFILTER-Befehl | SET INTENSITY-Befehl | SET KEY-Befehl
SET KEYCOMP-Befehl | SET LOGERRORS-Befehl | SET MACDESKTOP-Befehl
SET MACHELP-Befehl | SET MACKEY-Befehl | SET MARGIN-Befehl
SET MARK OF-Befehl | SET MARK TO-Befehl | SET MEMOWIDTH-Befehl
SET MESSAGE-Befehl | SET MOUSE-Befehl | SET ODOMETER-Befehl
SET OLEOBJECT-Befehl | SET PALETTE-Befehl | SET PDSETUP-Befehl
SET POINT-Befehl | SET PRINTER-Befehl | SET READBORDER-Befehl
SET REFRESH-Befehl | SET RESOURCE-Befehl | SET SAFETY-Befehl
SET SCOREBOARD-Befehl | SET SECONDS-Befehl | SET SEPARATOR-Befehl
SET SHADOWS-Befehl | SET SKIP OF-Befehl | SET SPACE-Befehl
SET STATUS-Befehl | SET STATUS BAR-Befehl | SET STEP-Befehl
SET STICKY-Befehl | SET SYSFORMATS-Befehl | SET SYSMENU-Befehl
SET TALK-Befehl | SET TEXTMERGE-Befehl | SET TEXTMERGE DELIMITERS-Befehl
SET TOPIC-Befehl | SET TOPIC ID-Befehl | SET TRBETWEEN-Befehl
SET TYPEAHEAD-Befehl | SET VIEW-Befehl | SET WINDOW OF MEMO-Befehl
SET XCMDFILE-Befehl | _SHELL System Memory Variable | SHOW GET-Befehl
SHOW GETS-Befehl | SHOW MENU-Befehl | SHOW OBJECT-Befehl
SHOW POPUP-Befehl | SHOW WINDOW-Befehl | SIZE POPUP-Befehl
SIZE WINDOW-Befehl | SKPBAR( )-Funktion | SKPPAD( )-Funktion

Unterstützte ODBC-APIs

Der Visual FoxPro-ODBC-Treiber unterstützt die folgenden APIs:

- Die gesamte Core Level-API.
- Die gesamte Level 1-API.
- Den überwiegenden Teil der Level 2-API

Umfassende Informationen zu allen ODBC API-Funktionen finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

In jedem API-Thema finden Sie neben einer kurzen Übersicht alle Einzelheiten, die Visual FoxPro-spezifisch sind. Für einige Funktionen ist zu beachten, daß sie sich abhängig davon, ob die Datenquelle als Verbindung zu einem Verzeichnis mit freien Tabellen (.DBF-Dateien) oder als Verbindung zu einer Visual FoxPro-Datenbank (.DBC-Datei) definiert ist, unterschiedlich verhalten. Verschiedene Operationen werden nur für Datenbankverbindungen unterstützt.

Unterstützung der Core Level-API

Unterstützung der Level 1-API

Unterstützung der Level 2-API

Unterstützung der Core Level-API

Es werden alle Funktionen der ODBC Core Level-API unterstützt. Zu jeder dieser Funktionen finden Sie eine kurze Beschreibung sowie alle Besonderheiten, soweit sie Visual FoxPro betreffen.

SQLAllocConnect

SQLAllocEnv

SQLAllocStmt

SQLBindCol

SQLCancel

SQLColAttributes

SQLConnect

SQLDescribeCol

SQLDisconnect

SQLError

SQLExecDirect

SQLExecute

SQLFetch

SQLFreeConnect

SQLFreeEnv

SQLFreeStmt

SQLGetCursorName

SQLNumResultCols

SQLPrepare

SQLRowCount

SQLSetCursorName

SQLTransact

Unterstützung der Level 1-API

Es werden alle Funktionen der ODBC Level 1-API unterstützt. Zu jeder dieser Funktionen finden Sie eine kurze Beschreibung sowie alle Besonderheiten, soweit sie Visual FoxPro betreffen.

SQLBindParameter

SQLColumns

SQLDriverConnect

SQLGetConnectOption

SQLGetData

SQLGetFunctions

SQLGetInfo

SQLGetStmtOption

SQLGetTypeInfo

SQLParamData

SQLPutData

SQLSetConnectOption

SQLSetStmtOption

SQLSpecialColumns

SQLStatistics

SQLTables

Unterstützung der Level 2-API

Die folgenden Funktionen der ODBC Level 2-API werden vollständig oder teilweise unterstützt:

SQLDataSources

SQLDrivers

SQLExtendedFetch

SQLMoreResults

SQLNumParams

SQLParamOptions

SQLPrimaryKeys

SQLSetPos

SQLSetScrollOptions (wird teilweise unterstützt)

Die folgenden Funktionen der ODBC Level 2-API werden nicht unterstützt:

SQLBrowseConnect

SQLColumnPrivileges

SQLDescribeParam

SQLForeignKeys

SQLNativeSql

SQLProcedureColumns

SQLProcedures

SQLTablePrivileges

SQLAllocConnect

Unterstützung: Vollständig ODBC API-Übereinstimmung: Core Level

Reserviert innerhalb der Umgebung, die von der Umgebungskennung (environment handle, *henv*) angegeben wird, Arbeitsspeicher für eine Verbindungskennung (connection handle, *hdbc*). Der Treiber-Manager verarbeitet diesen Aufruf und ruft die Treiberfunktion **SQLAllocConnect** immer dann auf, wenn **SQLConnect**, **SQLBrowseConnect**, oder **SQLDriverConnect** aufgerufen wird.

Weitere Informationen zu **SQLAllocConnect** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLAllocEnv

Unterstützung: Vollständig ODBC API-Übereinstimmung: Core Level

Reserviert Arbeitsspeicher für eine Umgebungskennung *(*environment handle, *henv)* und initialisiert das ODBC-CLI (Call Level Interface), so daß das CLI von einer Anwendung verwendet werden kann.

Weitere Informationen zu **SQLAllocEnv** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLAllocStmt

Unterstützung: Vollständig ODBC API-Übereinstimmung: Core Level

Reserviert Arbeitsspeicher für eine Anweisungskennung (statement handle, *hstmt*) und verknüpft die Anweisungskennung mit der Verbindung, die von *hdbc* angegeben wird. Der Treiber-Manager übergibt diesen Aufruf an den Treiber, der dann den Arbeitsspeicher für die *hstmt*-Struktur reserviert.

Weitere Informationen zu **SQLAllocStmt** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLBindCol

Unterstützung: Vollständig ODBC API-Übereinstimmung: Core Level

Weist Speicherplatz für eine Ergebnisspalte (result column) zu und gibt den Typ des Ergebnisses an. Wenn **SQLFetch** oder **SQLExtendedFetch** aufgerufen wird, schreibt der Treiber die Daten aller gebundenen Spalten (bound columns) in die zugewiesenen Bereiche. Unter **SQLGetTypeInfo** finden Sie eine Tabelle, der Sie entnehmen können, wie die ODBC- und die Visual FoxPro-Datentypen aufeinander abgebildet werden.

Weitere Informationen zu **SQLBindCol** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLBindParameter

Unterstützung: Vollständig ODBC API-Übereinstimmung: Level 1

Verknüpft einen Puffer mit einem Parametermarkierer (parameter marker) einer SQL-Anweisung. Der Visual FoxPro-ODBC-Treiber unterstützt Eingabeparameter (input parameters) wie vom Argument *fParamType* angegeben.

Weitere Informationen zu **SQLBindParameter** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLCancel

Unterstützung: Vollständig ODBC API-Übereinstimmung: Core Level

Bricht die Verarbeitung ab, die für eine Anweisungskennung (statement handle, *hstmt*) ausgeführt wird.

Weitere Informationen zu **SQLCancel** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLColAttributes

Unterstützung: Vollständig ODBC API-Übereinstimmung: Core Level

Gibt Deskriptor-Informationen für eine Spalte einer Ergebnismenge (result set) zurück; die Deskriptor-Informationen werden als Zeichenfolge, als deskriptorabhängiger 32-Bit-Wert oder als ganze Zahl zurückgegeben.

Anmerkung **SQLColAttributes** kann nicht verwendet werden, um Informationen über die jeweilige Bookmark-Spalte (Spalte 0) zurückzugeben.

Der Visual FoxPro-ODBC-Treiber unterstützt alle *fDescType*-Werte. In der folgenden Tabelle wird erläutert, wie der Treiber für bestimmte Werte implementiert ist:

fDescType	**Erläuterung**
SQL_COLUMN_AUTO_INCREMENT	Gibt FALSE zurück: Visual FoxPro hat keine Zählerfelder (counter field).
SQL_COLUMN_CASE_SENSITIVE	Gibt immer TRUE zurück, wenn die Spalte den Typ Zeichen (Character) hat.
SQL_COLUMN_LABEL	Gibt den Spaltennamen zurück, der auch von SQL_COLUMN_NAME zurückgegeben wird.
SQL_COLUMN_MONEY	Gibt TRUE zurück, wenn die Spalte den Typ Währung (Currency) hat (wird in der Visual FoxPro-Sprache durch ein "Y" gekennzeichnet).
SQL_COLUMN_OWNER_NAME	Gibt immer eine leere Zeichenfolge zurück.
SQL_COLUMN_SEARCHABLE	Gibt SQL_UNSEARCHABLE zurück für Spalten, die den Typ Objekt (General) haben; solche Spalten können nicht in einer WHERE-Klausel verwendet werden. Gibt SQL_SEARCHABLE zurück für Spalten, die den Typ Zeichen (Character) oder Memo haben und für die NOCPTRANS nicht angegeben ist; solche Spalten können in einer WHERE-Klausel mit jedem Vergleichsoperator verwendet werden. Gibt SQL_ALL_EXCEPT_LIKE zurück für alle Spalten, die einen anderen Typ haben; solche Spalten können in einer WHERE-Klausel mit jedem Vergleichsoperator außer LIKE verwendet werden.

Weitere Informationen zu **SQLColAttributes** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLColumns

Unterstützung: Vollständig ODBC API-Übereinstimmung: Level 1

Erstellt eine Tabelle als Ergebnismenge (result set), in der Informationen zu den Spalten der angegebenen Tabellen stehen.

Weitere Informationen zu **SQLColumns** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLConnect

Unterstützung: Vollständig ODBC API-Übereinstimmung: Core Level

Stellt eine Verbindung zu einer Datenquelle her. Diese Datenquelle kann entweder eine Datenbank oder ein Verzeichnis mit Tabellen sein. Der Visual FoxPro-ODBC-Treiber ignoriert die Argumente *szUID*, *cbUID*, *szAuthStr* und *cbAuthStr*.

Weitere Informationen zu **SQLConnect** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLDataSources

Unterstützung: Vollständig ODBC API-Übereinstimmung: Level 2

Listet die Namen von Datenquellen auf.

Weitere Informationen zu **SQLDataSources** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLDescribeCol

Unterstützung: Vollständig ODBC API-Übereinstimmung: Core Level

Gibt für die angegebene Ergebnisspalte den Namen, den Typ, die Genauigkeit (precision), die Dezimalstellenanzahl (scale) und die Zulässigkeit von Nullwerten (nullability) zurück.

Weitere Informationen zu **SQLDescribeCol** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLDisconnect

Unterstützung: Vollständig ODBC API-Übereinstimmung: Core Level

Trennt eine Verbindung.

Weitere Informationen zu **SQLDisconnect** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLDriverConnect

Unterstützung: Vollständig ODBC API-Übereinstimmung: Level 1

Stellt eine Verbindung zu einer vorhandenen Datenquelle her, die entweder eine Datenbank oder ein Verzeichnis mit freien Tabellen sein kann. Die ODBC-Attributschlüsselwörter UID und PWD werden ignoriert. In der folgenden Tabelle sind die auch unterstützten Attributschlüsselwörter aufgeführt.

ODBC-Attributschlüsselwort	Attributwert
DSN	
UID	Wird vom Visual FoxPro-ODBC-Treiber ignoriert, verursacht aber keinen Fehler.
PWD	Wird vom Visual FoxPro-ODBC-Treiber ignoriert, verursacht aber keinen Fehler.
Driver	Der Name und die Position (Ordner) des Visual FoxPro-ODBC-Treibers; implementiert vom Treiber-Manager.

Attributschlüsselwörter des Visual FoxPro-ODBC-Treibers	Attributwert
BackgroundFetch	"Yes" oder "No".
Collate	"Machine" oder eine andere Sortierreihenfolge. Eine Liste der unterstützten Sortierreihenfolgen finden Sie unter SET COLLATE.
Description	
Exclusive	"Yes" oder "No".
SourceDB	Ein vollständig angegebener Pfad zu einem Verzeichnis, das freie Tabellen enthält, oder der absolute Pfad und der Dateiname einer Datenbank.
SourceType	"DBC" oder "DBF"
Version	

Ist der Name der Datenquelle nicht angegeben, fordert der Treiber-Manager den Benutzer auf, diese Information einzugeben (abhängig davon, wie das Argument *fDriverCompletion* eingestellt ist), und setzt dann die Verarbeitung fort. Werden noch weitere Informationen benötigt, zeigt der Visual FoxPro-ODBC-Treiber das Eingabedialogfeld an.

Weitere Informationen zu **SQLDriverConnect** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLDrivers

Unterstützung: Vollständig ODBC API-Übereinstimmung: Level 2

Listet die Beschreibung und die Attributschlüsselwörter eines Treibers auf.

Weitere Informationen zu **SQLDrivers** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLError

Unterstützung: Vollständig ODBC API-Übereinstimmung: Core Level

Gibt Fehler- oder Statusinformationen zum letzten Fehler zurück. Der Treiber verwaltet eine Liste aus Fehlern, die er abhängig davon, wie **SQLError** aufgerufen wird, für eines der Argumente *hstmt*, *hdbc* oder *henv* zurückgibt. Die Fehlerliste wird nach jeder Anweisung entsprechend bereinigt.

In der folgenden Tabelle sind die **SQLError**-Argumente und -Rückgabewerte beschrieben, die der Treiber verwendet.

SQLError-Argument	Beschreibung des Rückgabewerts
szSQLState	Der SQLSTATE-Wert, der dem Fehler zugeordnet ist.
PfNativeError	Ein Wert ungleich null zeigt eine eigene Fehlermeldung des Visual FoxPro-ODBC-Treibers an. Der Wert Null zeigt an, daß der Fehler vom Treiber entdeckt und auf den entsprechenden ODBC-Fehlercode abgebildet wurde.
szErrorMsg	Der Text des treibereigenen Fehlers oder des ODBC-Fehlers.
pcbErrorMsg	Die Länge der Meldung plus die Länge der Bezeichner.

Weitere Informationen zu den Fehlermeldungen des Treibers finden Sie unter Überblick: Fehlermeldungen. Weitere Informationen zu **SQLError** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLExecDirect

Unterstützung: Vollständig ODBC API-Übereinstimmung: Core Level

Führt eine neue, vorbereitbare SQL-Anweisung aus. Wenn die Anweisung Parameter enthält, verwendet der Visual FoxPro-ODBC-Treiber die aktuellen Werte der zu den Parametermarkierern (parameter marker) gehörenden Variablen.

Wenn Sie einen Stapelverarbeitungsbefehl (Batch-Befehl) erstellen möchten, um gleichzeitig mehrere SQL-Anweisungen zu übergeben, müssen Sie die SQL-Anweisungen des Stapels durch Semikolons (;) voneinander trennen.

Wenn der Name einer Tabelle, einer Ansicht oder eines Feldes Leerzeichen enthält, müssen Sie diesen Namen in Gravis (`) setzen. Enthält Ihre Datenbank z.B. eine Tabelle namens `Tabelle 1` und diese wiederum das Feld `Feld 1`, müssen Sie jedes Element des Bezeichners so einschließen, wie nachstehend gezeigt:

```
SELECT `Tabelle 1`.`Feld 1`, `Tabelle 1`.`Feld 2` FROM `Tabelle 1`
```

Weitere Informationen zu **SQLExecDirect** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLExecute

Unterstützung: Vollständig ODBC API-Übereinstimmung: Core Level

Führt eine vorbereitete SQL-Anweisung aus (eine Anweisung, die bereits mit **SQLPrepare** vorbereitet wurde). Wenn die Anweisung Parameter enthält, verwendet der Visual FoxPro-ODBC-Treiber die aktuellen Werte der zu den Parametermarkierern (parameter marker) gehörenden Variablen.

Weitere Informationen zu **SQLExecute** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLExtendedFetch

Unterstützung: Vollständig ODBC API-Übereinstimmung: Level 2

Ähnlich wie **SQLFetch**, gibt aber mehrere Zeilen zurück, wobei für jede Spalte ein Datenfeld (array) verwendet wird. Die Ergebnismenge (result set) ist vorwärts durchlaufbar (forward-scrollable). Ist der Cursor nicht als Vorwärts-Cursor (forward-only cursor), sondern als statischer Cursor (static cursor) definiert, ist die Ergebnismenge auch rückwärts durchlaufbar (backward-scrollable).

Der Visual FoxPro-ODBC-Treiber gibt standardmäßig keine Zeilen einer FoxPro-Tabelle zurück, die als gelöscht markiert sind. Zeilen, die zum Löschen markiert, aber noch nicht aus einer Tabelle gelöscht sind, werden nicht in den Cursor einer Ergebnisgruppe aufgenommen. Mit dem Befehl SET DELETED können Sie dieses Verhalten ändern.

Weitere Informationen zu **SQLExtendedFetch** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLFetch

Unterstützung: Vollständig ODBC API-Übereinstimmung: Core Level

Ruft eine Zeile einer Ergebnismenge (result set) ab und schreibt sie in die Positionen, die mit den vorhergehenden Aufrufen von **SQLBindCol** festgelegt wurden. Bereitet den Treiber darauf vor, Daten mit **SQLGetData** aus einer ungebundenen Spalte (unbound column) abrufen zu können.

Weitere Informationen zu **SQLFetch** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLFreeConnect

Unterstützung: Vollständig ODBC API-Übereinstimmung: Core Level

Gibt eine Verbindungskennung (connection handle) sowie den gesamten Arbeitsspeicher frei, der für die Kennung reserviert war.

Weitere Informationen zu **SQLFreeConnect** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLFreeEnv

Unterstützung: Vollständig ODBC API-Übereinstimmung: Core Level

Schließt den Visual FoxPro-ODBC-Treiber und gibt den gesamten Arbeitsspeicher frei, der für den Treiber reserviert war.

Weitere Informationen zu **SQLFreeEnv** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLFreeStmt

Unterstützung: Vollständig ODBC API-Übereinstimmung: Core Level

Beendet die Verarbeitung, die zur angegebenen Anweisungskennung (statement handle, *hstmt*) gehört, schließt alle offenen Cursor, die mit *hstmt* verknüpft sind, verwirft Ergebnisse, die noch nicht abgerufen wurden, und gibt optional alle Ressourcen frei, die der Anweisungskennung zugeordnet sind.

Weitere Informationen zu **SQLFreeStmt** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLGetConnectOption

Unterstützung: Teilweise ODBC API-Übereinstimmung: Level 1

Gibt die aktuelle Einstellung einer Verbindungsoption zurück. Diese Funktion wird nicht vollständig unterstützt: Der Treiber unterstützt zwar alle Werte, die für das Argument *fOption* möglich sind; die meisten der *vParam*-Werte für den *fOption*-Wert SQL_TXN_ISOLATION unterstützt er aber nicht.

Eine vollständige Liste der Werte, die das Argument *fOption* annehmen kann, finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide* unter **SQLSetConnectOption**.

In der folgenden Tabelle sind nur die Werte des Arguments *fOption* beschrieben, bei denen sich der Visual FoxPro-ODBC-Treiber wegen der Art, wie **SQLGetConnectOption** implementiert ist, abweichend verhält:

fOption	**Bemerkungen**
SQL_AUTOCOMMIT	Wenn Sie SQL_AUTOCOMMIT_OFF angeben, muß Ihre Anwendung Transaktionen explizit mit **SQLTransact** übernehmen (commit) oder zurücksetzen (roll back); der Visual FoxPro-ODBC-Treiber übernimmt eine transaktionsfähige Anweisung nach deren Beendigung nicht automatisch. Der Treiber startet eine Transaktion, wenn die Anweisung transaktionsfähig ist.
SQL_CURRENT_QUALIFIER	Kann ein vollständig angegebener Name einer Datenbank (.DBC-Datei) oder ein vollständiger Pfad eines Verzeichnisses sein, das Tabellen (.DBF-Dateien) enthält.
SQL_LOGINTIMEOUT	Gibt die folgende Fehlermeldung zurück: "Treiber nicht in der Lage, diese Operation durchzuführen".
SQL_CURSORS	Gibt die folgende Fehlermeldung zurück: "Treiber nicht in der Lage, diese Operation durchzuführen".
SQL_PACKET_SIZE	Gibt die folgende Fehlermeldung zurück: "Treiber nicht in der Lage, diese Operation durchzuführen".
SQL_TXN_ISOLATION	Der Treiber unterstützt nur SQL_TXN_READ_COMMITTED; Die folgenden Werte für *vParam* werden nicht unterstützt: SQL_TXN_READ_UNCOMMITTED; SQL_TXN_REAPEATABLE_READ; SQL_TXN_SERIALIZABLE

Weitere Informationen zu **SQLGetConnectOption** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLGetCursorName

Unterstützung: Vollständig ODBC API-Übereinstimmung: Core Level

Gibt den Namen des Cursors zurück, dem die angegebene Anweisungskennung (statement handle, *hstmt*) zugeordnet ist. Die Funktion **SQLGetCursorName** wurde in die Visual FoxPro-ODBC-Treiber-API aufgenommen, da sie Teil der ODBC Core Level-API ist; sie kann nicht mit anderen API-Funktionen verwendet werden, da der Treiber positionierte Aktualisierungen (positioned updates) nicht unterstützt.

Weitere Informationen zu **SQLGetCursorName** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLGetData

Unterstützung: Vollständig ODBC API-Übereinstimmung: Level 1

Ruft den Wert ab, der in dem angegebenen Feld des aktuellen Datensatzes einer Ergebnismenge (result set) steht.

Weitere Informationen zu **SQLGetData** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLGetFunctions

Unterstützung: Vollständig ODBC API-Übereinstimmung: Level 1

Gibt für alle unterstützten Funktionen TRUE zurück.

Der Visual FoxPro-ODBC-Treiber unterstützt alle ODBC Core Level- und Level 1-API-Funktionen. In der folgenden Tabelle ist angegeben, welche Level 2-Funktionen der Treiber unterstützt:

fFunction	**Unterstützt**
SQL_API_SQLBROWSECONNECT	Nein
SQL_API_SQLCOLUMNPRIVELEGES	Nein
SQL_API_SQLDATASOURCES	Ja
SQL_API_SQLDESCRIBEPARAM	Nein
SQL_API_SQLDRIVERS	Ja
SQL_API_SQLEXTENDEDFETCH	Ja
SQL_API_SQLFOREIGNKEYS	Nein
SQL_API_SQLMORERESULTS	Ja
SQL_API_SQLNATIVESQL	Nein
SQL_API_SQLNUMPARAMS	Ja
SQL_API_SQLPARAMOPTIONS	Ja
SQL_API_SQLPRIMARYKEYS	Ja
SQL_API_SQLPROCEDURECOLUMNS	Nein
SQL_API_SQLPROCEDURES	Nein
SQL_API_SQLSETPOS	Ja
SQL_API_SQLSETSCROLLOPTIONS	Ja
SQL_API_SQLTABLEPRIVILEGES	Nein

Weitere Informationen zu **SQLGetFunctions** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLGetInfo

Unterstützung: Vollständig ODBC API-Übereinstimmung: Level 1

Gibt allgemeine Informationen zum Visual FoxPro-ODBC-Treiber sowie zu der Datenquelle zurück, der die angegebene Verbindungskennung (connection handle, *hdbc*) zugeordnet ist. In der folgenden Liste ist für jeden Wert, den das Argument *fInfoType* annehmen kann, der entsprechende Rückgabewert angegeben. Bei einigen Rückgabewerten finden Sie weitere Erläuterungen.

Weitere Informationen zu **SQLGetInfo** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

A

SQL_ACCESSIBLE_PROCEDURES gibt "N" zurück.

SQL_ACCESSIBLE_TABLES gibt "Y" zurück.

SQL_ACTIVE_CONNECTIONS gibt 0 zurück.

SQL_ACTIVE_STATEMENTS gibt 0 zurück.

SQL_ALTER_TABLE gibt zurück
SQL_AT_ADD_COLUMN
SQL_AT_DROP_COLUMN

B

SQL_BOOKMARK_PERSISTENCE gibt SQL_BP_SCROLL zurück.

C

SQL_COLUMN_ALIAS gibt "Y" zurück.

SQL_CONCAT_NULL_BEHAVIOR gibt SQL_CB_NULL zurück.

SQL_CONVERT_BIGINT gibt 0 zurück. Der Visual FoxPro-ODBC-Treiber unterstützt BigInt nicht.

SQL_CONVERT_BINARY gibt 0 zurück.

SQL_CONVERT_BIT gibt 0 zurück.

SQL_CONVERT_CHAR gibt 0 zurück.

SQL_CONVERT_DATE gibt 0 zurück.

SQL_CONVERT_DECIMAL gibt 0 zurück.

SQL_CONVERT_DOUBLE gibt 0 zurück.

SQL_CONVERT_FLOAT gibt 0 zurück.

SQL_CONVERT_INTEGER gibt 0 zurück.

SQL_CONVERT_LONGVARBINARY gibt 0 zurück.

SQL_CONVERT_LONGVARCHAR gibt 0 zurück.

SQL_CONVERT_NUMERIC gibt 0 zurück.

SQL_CONVERT_REAL gibt 0 zurück.

SQL_CONVERT_SMALLINT gibt 0 zurück.

SQL_CONVERT_TIME gibt 0 zurück.

SQL_CONVERT_TIMESTAMP gibt 0 zurück.

SQL_CONVERT_TINYINT gibt 0 zurück.

SQL_CONVERT_VARBINARY gibt 0 zurück.

SQL_CONVERT_VARCHAR gibt 0 zurück.

SQL_CONVERT_FUNCTIONS gibt 0 zurück.

SQL_CORRELATION_NAME gibt SQL_CN_ANY zurück.

SQL_CURSOR_COMMIT_BEHAVIOR gibt SQL_CB_PRESERVE zurück.

SQL_CURSOR_ROLLBACK_BEHAVIOR gibt SQL_CB_PRESERVE zurück.

D

SQL_DATA_SOURCE_NAME gibt den Wert zurück, der als DSN an **SQLConnect** oder **SQLDriverConnect** übergeben wurde; es wird eine leere Zeichenfolge zurückgegeben, wenn kein Datenquellenname (DSN) angegeben wurde.

SQL_DATA_SOURCE_READ_ONLY gibt "N" zurück.

SQL_DATABASE_NAME gibt den vollständigen, gemäß UNC (Universal Naming Convention) formulierten Pfad der aktuellen Datenbank zurück, wenn die Datenquelle eine Datenbank ist. Ist die Datenquelle ein Verzeichnis mit Tabellen, gibt die Funktion den Pfad des Verzeichnisses zurück.

SQL_DBMS_NAME gibt "Visual FoxPro" zurück.

SQL_DBMS_VER gibt "03.00.0000" zurück.

SQL_DEFAULT_TXN_ISOLATION gibt SQL_TXN_READ_COMMITTED zurück. Unsauberes Lesen (dirty read) ist nicht möglich, aber nichtwiederholbares Lesen (nonrepeatable read) und Phantome sind möglich.

SQL_DRIVER_HDBC ist durch den Treiber-Manager implementiert.

SQL_DRIVER_HENV ist durch den Treiber-Manager implementiert.

SQL_DRIVER_HLIB ist durch den Treiber-Manager implementiert.

SQL_DRIVER_HSTMT ist durch den Treiber-Manager implementiert.

SQL_DRIVER_NAME gibt "VFPODBC.DLL" zurück.

SQL_DRIVER_ODBC_VER gibt "02.50" (SQL_SPEC_MAJOR, SQL_SPEC_MINOR) zurück.

SQL_DRIVER_VER gibt "01.00.0000" zurück.

E

SQL_EXPRESSIONS_IN_ORDERBY gibt "N" zurück.

F

SQL_FETCH_DIRECTION gibt zurück
- SQL_FD_FETCH_NEXT
- SQL_FD_FETCH_FIRST
- SQL_FD_FETCH_LAST
- SQL_FD_FETCH_PRIOR
- SQL_FD_FETCH_ABSOLUTE
- SQL_FD_FETCH_RELATIVE
- SQL_FD_FETCH_BOOKMARK

SQL_FILE_USAGE gibt SQL_FILE_QUALIFIER für beide Datenquellen zurück: für Datenbanken (.DBC-Dateien) und freie Tabellen(.DBF-Dateien).

G-H

SQL_GETDATA_EXENSIONS gibt zurück
- SQL_GD_ANY_COLUMN
- SQL_GD_ANY_BLOCK
- SQL_GD_ANY_BOUND
- SQL_GD_ANY_ORDER.

SQL_GROUP_BY gibt SQL_GB_NO_RELATION zurück.

I-J

SQL_IDENTIFIER_CASE gibt SQL_IC_MIXED zurück.
SQL_IDENTIFIER_QUOTE_CHAR gibt ` zurück.

K

SQL_KEYWORDS gibt "" zurück.

L

SQL_LIKE_ESCAPE_CLAUSE gibt "N" zurück.
SQL_LOCK_TYPES gibt SQL_LCK_NO_CHANGE zurück.

M

SQL_MAX_BINARY_LITERAL_LEN gibt 0 zurück
SQL_MAX_CHAR_LITERAL_LEN gibt 254 zurück.
SQL_MAX_COLUMN_NAME_LEN gibt 128 zurück.
SQL_MAX_COLUMNS_IN_GROUP_BY gibt 16 zurück.
SQL_MAX_COLUMNS_IN_ORDER_BY gibt 16 zurück.
SQL_MAX_COLUMNS_IN_INDEX gibt 0 zurück.
SQL_MAX_COLUMNS_IN_SELECT gibt 254 zurück.
SQL_MAX_COLUMNS_IN_TABLE gibt 254 zurück.
SQL_MAX_CURSOR_NAME_LEN gibt 254 zurück.
SQL_MAX_INDEX_SIZE gibt 0 zurück.
SQL_MAX_OWNER_NAME_LEN gibt 0 zurück.
SQL_MAX_PROCEDURE_NAME_LEN gibt 0 zurück. Der Visual FoxPro-ODBC-Treiber ermöglicht es nicht, direkt auf gespeicherte Visual FoxPro-Prozeduren (stored procedures) zuzugreifen.
SQL_MAX_QUALIFIER_NAME_LEN gibt die maximale Länge zurück, die ein Betriebssystempfad haben kann.
SQL_MAX_ROW_SIZE gibt 254^2 zurück.
SQL_MAX_ROW_SIZE_INCLUDES_LONG gibt "N" zurück.
SQL_MAX_STATEMENT_LEN gibt 8192 zurück.
SQL_MAX_TABLE_NAME_LEN gibt 128 zurück.
SQL_MAX_TABLES_IN_SELECT gibt 16 zurück.
SQL_MAX_USER_NAME_LEN gibt 0 zurück.
SQL_MULT_RESULT_SETS gibt "Y" zurück.
SQL_MULTIPLE_ACTIVE_TXN gibt "Y" zurück. Mehrere Verbindungen können gleichzeitig Transaktionen geöffnet haben.

N

SQL_NEED_LONG_DATA_LEN gibt "N" zurück.
SQL_NON_NULLABLE_COLUMNS gibt SQL_NNC_NON_NULL zurück.
SQL_NULL_COLLATION gibt SQL_NC_LOW zurück.
SQL_NUMERIC_FUNCTIONS gibt alle Funktionen außer der Funktion SQL_FN_NUM_POWER zurück, die nicht vom Visual FoxPro-ODBC-Treiber unterstützt wird. Die folgenden Funktionen werden unterstützt:

SQL_FN_NUM_ABS
SQL_FN_NUM_ACOS
SQL_FN_NUM_ASIN
SQL_FN_NUM_ATAN
SQL_FN_NUM_ATAN2
SQL_FN_NUM_CELING
SQL_FN_NUM_COS
SQL_FN_NUM_COT
SQL_FN_NUM_DEGREES
SQL_FN_NUM_EXP
SQL_FN_NUM_FLOOR
SQL_FN_NUM_LOG
SQL_FN_NUM_LOG10
SQL_FN_NUM_MOD
SQL_FN_NUM_PI
SQL_FN_NUM_RADIANS
SQL_FN_NUM_RAND
SQL_FN_NUM_ROUND
SQL_FN_NUM_SIGN
SQL_FN_NUM_SIN
SQL_FN_NUM_SQRT
SQL_FN_NUM_TAN

O

SQL_ODBC_API_CONFORMANCE gibt SQL_OAC_LEVEL1 zurück.

SQL_ODBC_SAG_CLI_CONFORMANCE gibt SQL_OSCC_COMPLIANT zurück.

SQL_ODBC_SQL_CONFORMANCE gibt SQL_OSC_MINIMUM zurück. Die SQL-Grammatikstufe Minimum (minimum SQL grammar) wird unterstützt.

SQL_ODBC_SQL_OPT_IEF gibt "N" zurück.

SQL_ODBC_VER ist durch den Treiber-Manager implementiert.

SQL_ORDER_BY_COLUMNS_IN_SELECT gibt "N" zurück.

SQL_OUTER_JOINS gibt "N" zurück.

SQL_OWNER_TERM gibt "" zurück. Der Visual FoxPro-ODBC-Treiber unterstützt für seine Objekte keine Besitzer (owners).

SQL_OWNER_USAGE gibt 0 zurück. Der Visual FoxPro-ODBC-Treiber unterstützt für seine Objekte keine Besitzer (owners).

P

SQL_POS_OPERATIONS gibt SQL_POS_POSITION zurück.

SQL_POSITIONED_STATEMENTS gibt 0 zurück.

SQL_PROCEDURE_TERM gibt "" zurück.

SQL_PROCEDURES gibt "N" zurück.

Q

SQL_QUALIFIER_LOCATION gibt SQL_QL_START zurück.

SQL_QUALIFIER_NAME_SEPARATOR gibt "!" oder "\" zurück. Für eine Datenquelle, die eine Datenbank ist, dient ein "!" als Trennzeichen zwischen dem Datenbank- und einem Tabellennamen; für eine Datenquelle, die ein Verzeichnis mit freien Tabellen ist, dient ein "\" als Trennzeichen.

SQL_QUALIFIER_TERM gibt "database" oder "directory" zurück. Der Bezeichner "database" wird für

eine Datenquelle verwendet, die eine Datenbank ist; "directory" wird für eine Datenquelle verwendet, die ein Verzeichnis mit freien Tabellen ist.

SQL_QUALIFIER_USAGE unterstützt SQL_QU_PRIVILEGE_DEFINITION nicht; sie gibt zurück:
SQL_QU_DML_STATEMENT
SQL_QU_TABLE_DEFINITION

SQL_QUOTED_IDENTIFIER_CASE gibt SQL_IC_MIXED zurück.

R

SQL_ROW_UPDATES gibt "N" zurück. Der Visual FoxPro-ODBC-Treiber unterstützt nur statische (static) und Vorwärts-Cursor (forward cursors).

S

SQL_SCROLL_CONCURRENCY gibt SQL_SCCO_READ_ONLY zurück.

SQL_SCROLL_OPTIONS gibt zurück
SQL_SO_STATIC
SQL_SO_READONLY

SQL_SEARCH_PATTERN_ESCAPE gibt "\" zurück.

SQL_SERVER_NAME gibt "" zurück.

SQL_SPECIAL_CHARACTERS gibt "~@#$%^" zurück.

SQL_STATIC_SENSITIVITY gibt 0 zurück. Der Visual FoxPro-ODBC-Treiber unterstützt keine positionierten Aktualisierungen (positioned updates).

SQL_STRING_FUNCTIONS unterstützt SQL_FN_STR_INSERT, SQL_FN_STR_LOCATE, SQL_FN_STR_LOCATE_2 und SQL_FN_STR_SOUNDEX nicht.
Sie gibt zurück
SQL_FN_STR_ASCII
SQL_FN_STR_CHAR
SQL_FN_STR_CONCAT
SQL_FN_STR_DIFFERENCE
SQL_FN_STR_LCASE
SQL_FN_STR_LEFT
SQL_FN_STR_LENGTH
SQL_FN_STR_LTRIM
SQL_FN_STR_REPEAT
SQL_FN_STR_REPLACE
SQL_FN_STR_RIGHT
SQL_FN_STR_RTRIM
SQL_FN_STR_SUBSTRING
SQL_FN_STR_UCASE
SQL_FN_STR_SPACE

SQL_SUBQUERIES gibt zurück
SQL_SQ_CORRELATED_SUBQUERIES
SQL_SQ_COMPARISON
SQL_SQ_EXISTS
SQL_SQ_IN
SQL_SQ_QUANTIFIED

SQL_SYSTEM_FUNCTIONS gibt zurück
SQL_FN_SYS_DBNAME
SQL_FN_SYS_IFNULL
aber nicht
SQL_FN_SYS_USERNAME

T

SQL_TABLE_TERM gibt "table" zurück.

SQL_TIMEDATE_ADD_INTERVALS gibt zurück
- SQL_FN_TSI_ SECOND
- SQL_FN_TSI_MINUTE
- SQL_FN_TSI_HOUR
- SQL_FN_TSI_DAY
- SQL_FN_TSI_MONTH
- SQL_FN_TSI_YEAR

Nicht SQL_FN_TSI_FRAC_SECOND, SQL_FN_TSI_WEEK und SQL_FN_TSI_QUARTER.

SQL_TIMEDATE_DIFF_INTERVALS gibt zurück
- SQL_FN_TSI_ SECOND
- SQL_FN_TSI_MINUTE
- SQL_FN_TSI_HOUR
- SQL_FN_TSI_DAY
- SQL_FN_TSI_MONTH
- SQL_FN_TSI_YEAR

SQL_TIMEDATE_FUNCTIONS unterstützt nicht SQL_FN_TD_QUARTER, SQL_FN_TD_TIMESTAMPADD, SQL_FN_TD_DAYOFYEAR und SQL_FN_TD_WEEK. Sie gibt zurück
- SQL_FN_TD_CURDATE
- SQL_FN_TD_CURTIME
- SQL_FN_TD_DAYNAME
- SQL_FN_TD_DAYOFMONTH
- SQL_FN_TD_DAYOFWEEK
- SQL_FN_TD_HOUR
- SQL_FN_TD_MINUTE
- SQL_FN_TD_MONTH
- SQL_FN_TD_MONTHNAME
- SQL_FN_TD_NOW
- SQL_FN_TD_SECOND
- SQL_FN_TD_TIMESTAMPDIFF
- SQL_FN_TD_YEAR

SQL_TXN_CAPABLE gibt SQL_TC_DML zurück.

SQL_TXN_ISOLATION_OPTION gibt SQL_TXN_READ_COMMITTED zurück.

U-Z

SQL_UNION gibt zurück
- SQL_U_UNION
- SQL_U_UNION_ALL

SQL_USER_NAME gibt <Leerzeichen> zurück.

SQLGetStmtOption

Unterstützung: Vollständig ODBC API-Übereinstimmung: Level 1

Gibt die aktuelle Einstellung einer Option einer Anweisung zurück.

fOption	**Rückgabewert**
SQL_GET_BOOKMARK	32-Bit-Ganzzahl (integer), die das Lesezeichen (bookmark) des aktuellen Datensatzes ist.
SQL_ROW_NUMBER	32-Bit-Ganzzahl, die die Position angibt, die der aktuelle Datensatz in der Ergebnismenge (result set) einnimmt.
SQL_TRANSLATE_DLL	Fehler: "Treiber nicht in der Lage, diese Operation durchzuführen".

Der Visual FoxPro-ODBC-Treiber hat keine Umwandlungs-DLL-Dateien (translation DLLs).

Weitere Informationen zu **SQLGetStmtOption** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLGetTypeInfo

Unterstützung: Vollständig ODBC API-Übereinstimmung: Level 1

Gibt Informationen zu den Datentypen zurück, die von einer Datenquelle unterstützt werden. Der Treiber gibt die Informationen in einer SQL-Ergebnismenge (result set) zurück. In der folgenden Tabelle sind die ODBC-Datentypen und die entsprechenden Visual FoxPro-Datentypen gegenübergestellt.

ODBC-Typ	Visual FoxPro-Typ
SQL_BIGINT	Nicht unterstützt. Visual FoxPro hat keinen 64-Bit-Datentyp.
SQL_BIT	Logisch
SQL_CHAR	Zeichen (Character)
SQL_DATE	Datum
SQL_DECIMAL	Numerisch
SQL_DOUBLE	Double
SQL_FLOAT	Double
SQL_INTEGER	Integer
SQL_LONGVARBINARY	Memo (binär)
SQL_LONGVARCHAR	Memo
SQL_NUMERIC	Numerisch*, Währung (Currency), Gleitkomma (Float)
SQL_REAL	Double
SQL_SMALLINT	Integer
SQL_TIME	Nicht unterstützt. Visual FoxPro hat den Datentyp *Zeit* (*time*) nicht.
SQL_TIMESTAMP	DatumZeit (DateTime)
SQL_TINYINT	Integer
SQL_VARBINARY	Memo (binär)*, Objekt (General)
SQL_VARCHAR	Zeichen (Character)

*Standardtyp

Weitere Informationen zu den Visual FoxPro-Datentypen finden Sie unter CREATE TABLE. Weitere Informationen zu **SQLGetTypeInfo** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLMoreResults

Unterstützung: Vollständig ODBC API-Übereinstimmung: Level 2

Ermittelt, ob weitere, noch nicht abgerufene Ergebnisse für eine Anweisungskennung (statement handle, *hstmt*) vorliegen, die SELECT-, UPDATE-, INSERT- oder DELETE-Anweisungen enthält. Ist dies der Fall, initialisiert **SQLMoreResults** die Verarbeitung dieser Ergebnisse.

Weitere Informationen zu **SQLMoreResults** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLNumParams

Unterstützung: Vollständig ODBC API-Übereinstimmung: Level 2

Gibt die Anzahl der Parameter einer SQL-Anweisung zurück. Die Anzahl der Parameter stimmt normalerweise mit der Anzahl der Fragezeichen in der SQL-Anweisung überein, die an **SQLPrepare** übergeben wurde.

Weitere Informationen zur SQL-Grammatik finden Sie unter Unterstützte ODBC SQL-Grammatik. Weitere Informationen zu **SQLNumParams** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLNumResultCols

Unterstützung: Vollständig ODBC API-Übereinstimmung: Core Level

Gibt die Anzahl der Spalten zurück, die der Cursor einer Ergebnismenge (result set) enthält.

Weitere Informationen zu **SQLNumResultCols** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLParamData

Unterstützung: Vollständig ODBC API-Übereinstimmung: Level 1

Wird zusammen mit **SQLPutData** verwendet, um während der Ausführungszeit einer Anweisung Parameterdaten bereitzustellen.

Weitere Informationen zu **SQLParamData** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLParamOptions

Unterstützung: Vollständig ODBC API-Übereinstimmung: Level 1

Ermöglicht es einer Anwendung, mehrere Werte für eine Gruppe von Parametern anzugeben, die mit **SQLBindParameter** zugewiesen wurden. Dies ist immer dann hilfreich, wenn viele Einfügevorgänge (inserts) oder andere Vorgänge ausgeführt werden müssen, für die die Datenquelle dieselbe SQL-Anweisung mehrfach mit unterschiedlichen Parameterwerten verarbeiten muß. Zum Beispiel kann eine Anwendung drei Wertegruppen für eine Parametergruppe angeben, die zu einer INSERT-Anweisung gehört, und dann die INSERT-Anweisung einmal aufrufen, um die drei Einfügevorgänge auszuführen.

Weitere Informationen zu **SQLParamOptions** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLPrepare

Unterstützung: Vollständig ODBC API-Übereinstimmung: Core Level

Bereitet eine SQL-Anweisung vor, indem sie ermittelt, wie die Anweisung optimiert werden kann. Die SQL-Anweisung wird vor ihrer Ausführung von **SQLExecDirect** kompiliert.

Wenn der Name einer Tabelle, einer Ansicht oder eines Feldes Leerzeichen enthält, müssen Sie diesen Namen in Gravis (`) setzen. Enthält Ihre Datenbank z.B. eine Tabelle namens `Tabelle 1` und diese wiederum das Feld `Feld 1`, müssen Sie jedes Element des Bezeichners so einschließen, wie nachstehend gezeigt:

```
SELECT * FROM `Tabelle 1`.`Feld 1`
```

Weitere Informationen zu **SQLPrepare** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLPrimaryKeys

Unterstützung: Vollständig ODBC API-Übereinstimmung: Level 2

Gibt die Namen der Spalten zurück, die den Primärschlüssel einer Tabelle bilden. Die Funktion **SQLPrimaryKeys** ist im Visual FoxPro-ODBC-Treiber so implementiert, daß folgende Einschränkungen gelten:

- Die Argumente *szTableOwner* und *cbTableOwner* werden ignoriert.
- Sie funktioniert nur für Datenquellen, die Datenbanken sind. Ist die Datenquelle ein Verzeichnis mit freien Tabellen, gibt der Treiber folgende Fehlermeldung zurück: "Diese Funktion wird vom Treiber nicht unterstützt".

Weitere Informationen zu **SQLPrimaryKeys** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLPutData

Unterstützung: Vollständig ODBC API-Übereinstimmung: Level 1

Ermöglicht es einer Anwendung, während der Ausführungszeit einer Anweisung Daten für einen Parameter oder eine Spalte an den Treiber zu senden.

Weitere Informationen zu **SQLPutData** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLRowCount

Unterstützung: Vollständig ODBC API-Übereinstimmung: Core Level

Gibt die Anzahl der Zeilen zurück, die von der letzten UPDATE-, INSERT- oder DELETE-Anweisung betroffen waren.

Weitere Informationen zu **SQLRowCount** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLSetConnectOption

Unterstützung: Teilweise ODBC API-Übereinstimmung: Level 1

Stellt Optionen ein, die bestimmte Aspekte einer Verbindung festlegen. Diese Funktion wird nicht vollständig unterstützt: Der Treiber unterstützt zwar alle Werte, die für das Argument *fOption* möglich sind; die meisten der *vParam*-Werte für den *fOption*-Wert SQL_TXN_ISOLATION unterstützt er aber nicht.

Eine vollständige Liste der Werte, die das Argument *fOption* annehmen kann, finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

In der folgenden Tabelle sind nur die Werte des Arguments *fOption* beschrieben, bei denen sich der Visual FoxPro-ODBC-Treiber wegen der Art, wie **SQLGetConnectOption** implementiert ist, abweichend verhält:

fOption	**Bemerkungen**
SQL_AUTOCOMMIT	Wenn Sie SQL_AUTOCOMMIT_OFF angeben, muß Ihre Anwendung Transaktionen explizit mit **SQLTransact** übernehmen (commit) oder zurücksetzen (roll back); der Visual FoxPro-ODBC-Treiber übernimmt eine transaktionsfähige Anweisung nach deren Beendigung nicht automatisch. Der Treiber startet eine Transaktion, wenn die Anweisung transaktionsfähig ist.
SQL_CURRENT_QUALIFIER	Kann ein vollständig angegebener Name einer Datenbank oder ein vollständiger Pfad zu einem Verzeichnis sein, das freie Tabellen enthält.
SQL_LOGINTIMEOUT	Gibt die folgende Fehlermeldung zurück: "Treiber nicht in der Lage, diese Operation durchzuführen".
SQL_CURSORS	Gibt die folgende Fehlermeldung zurück: "Treiber nicht in der Lage, diese Operation durchzuführen".
SQL_PACKET_SIZE	Gibt die folgende Fehlermeldung zurück: "Treiber nicht in der Lage, diese Operation durchzuführen".
SQL_TXN_ISOLATION	Der Treiber unterstützt nur SQL_TXN_READ_COMMITTED; Die folgenden Werte für *vParam* werden nicht unterstützt: SQL_TXN_READ_UNCOMMITTED; SQL_TXN_REAPEATABLE_READ; SQL_TXN_SERIALIZABLE

Weitere Informationen zu **SQLSetConnectOption** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLSetCursorName

Unterstützung: Vollständig ODBC API-Übereinstimmung: Core Level

Verknüpft den Namen eines Cursors mit einer aktiven Anweisungskennung (statement handle, *hstmt*). Die Funktion **SQLSetCursorName** wurde in die Visual FoxPro-ODBC-Treiber-API aufgenommen, da sie Teil der ODBC Core Level-API ist; sie kann nicht mit anderen API-Funktionen verwendet werden, da der Treiber positionierte Aktualisierungen (positioned updates) nicht unterstützt.

Weitere Informationen zu **SQLSetCursorName** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLSetPos

Unterstützung: Vollständig ODBC API-Übereinstimmung: Level 2

Legt in einer Datensatzgruppe die Position des Cursors fest. Sie können **SQLSetPos** zusammen mit **SQLGetData** verwenden, um Zeilen ungebundener Spalten abzurufen, nachdem Sie den Cursor auf eine bestimmte Zeile der Datensatzgruppe (rowset) gesetzt haben.

Weitere Informationen zu **SQLSetPos** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLSetScrollOptions

Unterstützung: Teilweise ODBC API-Übereinstimmung: Level 2

Stellt Optionen ein, die das Verhalten eines Cursors festlegen, der mit einer Anweisungskennung (statement handle, *hstmt*) verknüpft ist.

Der Visual FoxPro-ODBC-Treiber unterstützt nur SQL_CONCUR_READ_ONLY; den *fConcurrency*-Wert SQL_CONCUR_ROWVER unterstützt er nicht. Der Treiber wandelt SQL_KEYSET_SIZE, SQL_CURSOR_DYNAMIC und SQL_CURSOR_KEYSET_DRIVEN in SQL_SCROLL_STATIC um und gibt die Warnung ODBC_01S02 aus.

Weitere Informationen zu **SQLSetScrollOptions** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLSetStmtOption

Unterstützung: Vollständig ODBC API-Übereinstimmung: Level 1

Stellt Optionen ein, die zu einer Anweisungskennung (statement handle, *hstmt*) gehören.

fOption	**Zulässige Werte**	**Erläuterungen**
SQL_ASYNC_ENABLE	SQL_ASYNC_ENABLE_OFF	Wenn Sie versuchen, diese *fOption* einzustellen, gibt der Treiber folgende Fehlermeldung zurück: "Treiber nicht in der Lage, diese Operation durchzuführen". Visual FoxPro unterstützt keine asynchrone Ausführung.
SQL_BIND_TYPE	SQL_BIND_BY_COLUMN oder ein 32-Bit-Wert, der die Länge der Struktur oder eine Instanz eines Puffers angibt, an den die Ergebnisspalten gebunden werden sollen.	
SQL_CONCURRENCY	SQL_CONCUR_READ_ONLY SQL_CONCUR_LOCK SQL_CONCUR_VALUES	Der Treiber unterstützt SQL_CONCUR_ROWVER nicht, da Visual FoxPro keine Zeilenversionsvergleiche (row versioning) vornimmt, die auf TIMESTAMP-Werten beruhen.
SQL_CURSOR_TYPE	SQL_CURSOR_FORWARD_ONLY SQL_CURSOR_STATIC	Der Treiber unterstützt SQL_CURSOR_KEYSET_DRIVEN und SQL_CURSOR_DYNAMIC nicht; weitere Informationen finden Sie unter **SQLSetScrollOptions**.
SQL_KEYSET_SIZE	Fehler: "Treiber nicht in der Lage, diese Operation durchzuführen".	Visual FoxPro unterstützt das Cursor-Modell nicht, das auf Schlüsselmengen (keysets) basiert.
SQL_MAX_LENGTH	0	Wenn Sie versuchen, *fOption* auf diesen Wert einzustellen, gibt der Treiber folgende Fehlermeldung zurück: "Treiber nicht in der Lage, diese Operation durchzuführen".
SQL_MAX_ROWS	0	Wenn Sie versuchen, *fOption* auf diesen Wert einzustellen, gibt der Treiber folgende Fehlermeldung zurück: "Treiber nicht in der Lage, diese Operation durchzuführen".
SQL_NOSCAN	SQL_NOSCAN_OFF	
SQL_QUERY_TIMEOUT	0	Wenn Sie versuchen, *fOption* auf diesen Wert einzustellen, gibt der Treiber folgende Fehlermeldung zurück: "Treiber nicht in der Lage, diese Operation durchzuführen".
SQL_RETRIEVE_DATA	SQL_RD_ON, SQL_RD_OFF	
SQL_ROWSET_SIZE	1 bis 4.294.967.296	
SQL_SIMULATE_CURSOR	Fehler: "Treiber nicht in der Lage,	

	diese Operation durchzuführen“.
SQL_USE_BOOKMARKS	SQL_UB_OFF SQL_UB_ON

Weitere Informationen zu **SQLSetStmtOption** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer’s Reference and SDK Guide*.

SQLSpecialColumns

Unterstützung: Vollständig ODBC API-Übereinstimmung: Level 1

Ermittelt die optimale Gruppe der Spalten, die eine Zeile der Tabelle eindeutig kennzeichnen.

Der Visual FoxPro-ODBC-Treiber gibt Informationen zu den Spalten zurück, die den Primärschlüssel der FoxPro-Tabelle bilden (siehe **SQLPrimaryKeys**). War *fColType* beim Aufruf der Funktion auf SQL_ROWVER eingestellt, werden keine Spalten zurückgegeben. **SQLSpecialColumns** funktioniert nur für Datenquellen, die Datenbanken sind.

Weitere Informationen zu **SQLSpecialColumns** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLStatistics

Unterstützung: Vollständig ODBC API-Übereinstimmung: Level 1

Ruft eine Liste mit einer Statistik für eine Tabelle sowie Informationen zu den Indizes ab, die zu der Tabelle gehören. Der Treiber gibt die Informationen als Ergebnismenge (result set) zurück.

Weitere Informationen zu **SQLStatistics** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLTables

Unterstützung: Vollständig ODBC API-Übereinstimmung: Level 1

Gibt eine Liste mit den Tabellennamen zurück, die von dem Parameter in der **SQLTables**-Anweisung angegeben sind. Ist kein Parameter angegeben, gibt die Funktion die Namen der Tabellen zurück, die in der aktuellen Datenquelle gespeichert sind. Der Treiber gibt die Informationen als Ergebnismenge (result set) zurück.

Aufrufe des Datentyps Aufzählung erhalten keinen Ergebnismengeneintrag für Remote-Ansichten oder lokale parametrisierte Ansichten. Ein Aufruf von **SQL Tables** mit einem eindeutigen Tabellennamen findet Entsprechungen für eine solche Ansicht, sofern vorhanden. Dies ermöglicht der eingesetzten API, Namenskonflikte zu überprüfen, bevor eine neue Tabelle erstellt wird.

Anmerkung Der Visual FoxPro-ODBC-Treiber unterscheidet selbst dann zwischen Datenbanktabellen und freien Tabellen, wenn Tabellen beider Typen in demselben Verzeichnis Ihres Computers gespeichert sind. Ist Ihre Datenquelle ein Verzeichnis mit freien Tabellen, gibt der Visual FoxPro-ODBC-Treiber keinen der Namen der Tabellen zurück, die einer Datenbank zugeordnet sind.

Weitere Informationen zu **SQLTables** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

SQLTransact

Unterstützung: Vollständig ODBC API-Übereinstimmung: Core Level

Löst einen Übernahmevorgang (commit) oder Rücksetzvorgang (roll back) aus: entweder für alle aktiven Vorgänge auf allen Anweisungskennungen (statement handle, *hstmt*), die mit einer Verbindung verknüpft sind; oder für alle Verbindungen, die mit der Verbindungskennung (connection handle, *hdbc*) verknüpft sind. **SQLTransact** funktioniert nur für Datenquellen, die Datenbanken sind.

Schlägt ein Übernahmevorgang im manuellen Modus fehl, bleibt die Transaktion aktiv, und Sie haben die Wahl, die Transaktion zurückzusetzen oder den Übernahmevorgang erneut zu versuchen. Schlägt ein Übernahmevorgang im automatischen Transaktionsmodus fehl, wird die Transaktion automatisch zurückgesetzt; die Transaktion kann nicht inaktiv sein.

Weitere Informationen zu **SQLTransact** finden Sie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

Datenbank

In Visual FoxPro gilt für eine Datenbankdatei folgendes: Sie hat die Erweiterung .DBC und kann eine oder mehrere Tabellen enthalten.

Datenbanktabelle

In Visual FoxPro eine Tabelle, die zu einer Datenbank gehört. Vergleiche freie Tabelle.

Freie Tabelle

In Visual FoxPro eine Tabelle, die nicht zu einer Datenbank gehört.

Eine .DBF-Datei, die mit FoxPro, Version 2.x, erstellt wurde, ist eine freie Tabelle, sofern sie nicht in eine Visual FoxPro-Tabelle umgewandelt und zu einer Visual FoxPro-Datenbank hinzugefügt wurde. Vergleiche Datenbanktabelle.

Vorbereitbare SQL-Anweisung

Eine SQL-Anweisung, die noch nicht von **SQLPrepare** verarbeitet wurde.

Tabelle

In Visual FoxPro werden Datensätze in einer Tabelle gespeichert. Jede Zeile einer Tabelle entspricht einem Datensatz, und die Spalten einer Tabelle entsprechen den Datenfeldern eines Datensatzes. Jede Visual FoxPro-Tabelle wird in einer eigenen Datei gespeichert, die die Erweiterung .DBF hat. Visual FoxPro-Tabellen können einer Datenbank zugeordnet werden.

FoxPro-Tabellen, die in einem Version 2.*x*-Format vorliegen, sind keiner Datenbank zugeordnet.

ALTER TABLE - SQL-Befehl

Siehe auch

Ermöglicht es, die Struktur einer Tabelle aus einem Programm heraus zu ändern.

Syntax

ALTER TABLE *Tabellenname1*
  ADD | ALTER [COLUMN] *Feldname1*
    Feldtyp [(*nFeldbreite* [, *nDezimalstellen*])]
    [NULL | NOT NULL]
    [CHECK *lAusdruck1* [ERROR *zMeldung1*]]
    [DEFAULT *aAusdruck1*]
    [PRIMARY KEY | UNIQUE]
    [REFERENCES *Tabellenname2* [TAG *Indexname1*]]
    [NOCPTRANS]
- Oder -
ALTER TABLE *Tabellenname1*
  ALTER [COLUMN] *Feldname2*
    [NULL | NOT NULL]
    [SET DEFAULT *aAusdruck2*]
    [SET CHECK *lAusdruck2* [ERROR *zMeldung2*]]
    [DROP DEFAULT]
    [DROP CHECK]
- Oder -
ALTER TABLE *Tabellenname1*
  [DROP [COLUMN] *Feldname3*]
  [SET CHECK *lAusdruck3* [ERROR *zMeldung3*]]
  [DROP CHECK]
  [ADD PRIMARY KEY *aAusdruck3* TAG *Indexname2*]
  [DROP PRIMARY KEY]
  [ADD UNIQUE *aAusdruck4* [TAG *Indexname3*]]
  [DROP UNIQUE TAG *Indexname4*]
  [ADD FOREIGN KEY [*aAusdruck5*] TAG *Indexname4*
    REFERENCES *Tabellenname2* [TAG *Indexname5*]]
  [DROP FOREIGN KEY TAG *Indexname6* [SAVE]]
  [RENAME COLUMN *Feldname4* TO *Feldname5*]
  [NOVALIDATE]

Argumente

Tabellenname1 Gibt den Namen der Tabelle an, deren Struktur geändert werden soll.

ADD [COLUMN] *Feldname1* Gibt den Namen des Feldes an, das hinzugefügt werden soll.

ALTER [COLUMN] *Feldname1* Gibt den Namen eines bereits vorhandenen Feldes an, das geändert werden soll.

Feldtyp [(*nFeldbreite* [, *nDezimalstellen*]]) Gibt für das neue oder zu ändernde Feld folgendes an: den Feldtyp, die Feldbreite sowie die Anzahl der Dezimalstellen des Feldes.

Feldtyp ist ein Buchstabe, der den Datentyp des Feldes bestimmt. Für einige Felddatentypen ist es erforderlich, daß Sie entweder *nFeldbreite* oder *nDezimalstellen* bzw. beide Argumente angeben.

nFeldbreite und *nDezimalstellen* werden bei den Feldtypen D, T, I, Y, L, M, und G ignoriert. Ist das Argument *nDezimalstellen* bei einem der Typen N, F oder B nicht angegeben, wird es standardmäßig auf 0 (keine Dezimalstellen) eingestellt.

NULL | NOT NULL Gibt an, ob das Feld Nullwerte enthalten darf oder nicht.

Ist keine der Klauseln NULL bzw. NOT NULL angegeben, bestimmt die aktuelle Einstellung von SET NULL, ob in dem Feld Nullwerte zulässig sind. Dies gilt allerdings nicht, wenn Sie zwar keine der Klauseln NULL bzw. NOT NULL, aber eine der Klauseln PRIMARY KEY oder UNIQUE angegeben haben. In diesem Fall wird die Einstellung von SET NULL ignoriert und das Feld standardmäßig auf NOT NULL eingestellt.

CHECK *lAusdruck1* Gibt für das Feld eine Gültigkeitsregel an. *lAusdruck1* muß ein logischer Ausdruck und kann sowohl eine benutzerdefinierte Funktion als auch eine gespeicherte Prozedur sein. Es ist zu beachten, daß diese Gültigkeitsregel auch dann angewendet wird, wenn ein leerer Datensatz angefügt wird. Es ergibt sich ein Fehler, wenn die Gültigkeitsregel es nicht zuläßt, daß das Feld eines angefügten Datensatzes einen leeren Wert enthält.

ERROR *zMeldung1* Gibt die Fehlermeldung an, die angezeigt wird, wenn die Gültigkeitsregel des Feldes einen Fehler verursacht.

DEFAULT *aAusdruck1* Gibt für das Feld einen Standardwert an. Der Datentyp von *aAusdruck1* und der Datentyp des Feldes müssen identisch sein.

PRIMARY KEY Erstellt auf Basis des Feldes einen Primärindex, der denselben Namen hat wie das Feld.

UNIQUE Erstellt auf Basis des Feldes einen potentiellen Primärindex, der denselben Namen hat wie das Feld.

Anmerkung Potentielle Primärindizes (die erstellt werden, wenn die Klausel UNIQUE angegeben ist, die aus Gründen der ANSI-Kompatibilität Bestandteil der Befehle ALTER TABLE und CREATE TABLE ist) sind nicht dasselbe wie Indizes, die mit einem INDEX-Befehl erstellt werden, wenn in diesem die Klausel UNIQUE angegeben ist. Ein mit einem INDEX-Befehl sowie dessen UNIQUE-Klausel erstellter Index gestattet Duplikatindexwerte. Demgegenüber gestatten potentielle Primärindizes keine Duplikatindexwerte.

In einem Feld, auf dem ein Primär- oder potentieller Primärindex basiert, sind weder Nullwerte noch Duplikatwerte (Duplikatdatensätze) zulässig.

Wenn Sie mit der Klausel ADD COLUMN ein neues Feld erstellen, meldet Visual FoxPro keinen Fehler, wenn Sie einen Primär- oder potentiellen Primärindex für ein Feld erstellen, das Nullwerte unterstützt. Visual FoxPro meldet allerdings einen Fehler, wenn Sie versuchen, einen Null- oder Duplikatwert in ein Feld einzugeben, das als Primär- oder potentieller Primärindex verwendet wird.

Für den Fall, daß Sie ein bereits vorhandenes Feld ändern und der Ausdruck des Primär- oder potentiellen Primärindexes aus Feldern der Tabelle besteht, prüft Visual FoxPro die Felder daraufhin, ob sie Nullwerte oder Duplikatwerte enthalten. Ist dies der Fall, meldet Visual FoxPro einen Fehler, und an der Tabelle werden keine Änderungen vorgenommen.

REFERENCES *Tabellenname2* TAG *Indexname1* Gibt die Master-Tabelle an, zu der eine persistente Beziehung hergestellt wird. TAG *Indexname1* gibt den Namen des Indexes der Master-Tabelle an, auf dem die Beziehung basiert. Der Name eines Indexes kann aus bis zu 10 Zeichen bestehen.

NOCPTRANS Verhindert für Zeichen- oder Memo-Felder, daß sie gemäß einer anderen Codeseite umgewandelt werden. Wird eine Tabelle an eine andere Codeseite angepaßt, werden die Felder, für die NOCPTRANS angegeben ist, nicht umgewandelt. NOCPTRANS kann nur für Zeichen- bzw. Memo-Felder angegeben werden.

Im folgenden Beispiel wird unter dem Namen TABELLE1 eine Tabelle erstellt, die aus zwei Zeichen- und zwei Memo-Feldern besteht. Sowohl für das zweite Zeichen-Feld (`zeichen2`) als auch für das zweite Memo-Feld (`memo2`) ist NOCPTRANS angegeben, damit diese Felder nicht umgewandelt werden.

```
CREATE TABLE tabelle1 (zeichen1 C(10), zeichen2 C(10) NOCPTRANS,;
   memo1 M, memo2 M NOCPTRANS)
```

ALTER [COLUMN] *Feldname2* Gibt den Namen eines bereits vorhandenen Feldes an, das geändert werden soll.

SET DEFAULT *aAusdruck2* Gibt für das bereits vorhandene Feld den neuen Standardwert an. *aAusdruck2* muß denselben Datentyp haben wie das Feld.

SET CHECK *lAusdruck2* Gibt für das bereits vorhandene Feld die neue Gültigkeitsregel an. *lAusdruck2* muß ein logischer Ausdruck und kann eine benutzerdefinierte Funktion oder eine gespeicherte Prozedur sein.

ERROR *zMeldung2* Gibt die Fehlermeldung an, die angezeigt wird, wenn die Gültigkeitsregel des Feldes einen Fehler verursacht. Diese Meldung wird nur angezeigt, wenn innerhalb eines Datenblatt- oder Bearbeitungsfensters Änderungen an Daten vorgenommen werden.

DROP DEFAULT Löscht den Standardwert des bereits vorhandenen Feldes.

DROP CHECK Löscht die Gültigkeitsregel des bereits vorhandenen Feldes.

DROP [COLUMN] *Feldname3* Gibt den Namen des Feldes an, das aus der Tabelle gelöscht werden soll. Neben dem Feld selbst werden auch dessen Standardwert sowie Gültigkeitsregel aus der Tabelle gelöscht.

Für den Fall, daß Index- oder Triggerausdrücke auf das Feld verweisen, werden diese Ausdrücke ungültig, sobald das Feld gelöscht wurde. Sollte diese Situation eintreten, wird kein Fehler gemeldet, wenn das Feld gelöscht wird. Statt dessen werden die dann ungültigen Index- oder Triggerausdrücke während der Laufzeit Fehler verursachen.

SET CHECK *lAusdruck3* Gibt die Gültigkeitsregel der Tabelle an. *lAusdruck3* muß ein logischer Ausdruck und kann eine benutzerdefinierte Funktion oder eine gespeicherte Prozedur sein.

ERROR *zMeldung3* Gibt die Fehlermeldung an, die angezeigt wird, wenn die Gültigkeitsregel der Tabelle einen Fehler verursacht. Diese Meldung wird nur angezeigt, wenn innerhalb eines Datenblatt- oder Bearbeitungsfensters Änderungen an Daten vorgenommen werden.

DROP CHECK Löscht die Gültigkeitsregel der Tabelle.

ADD PRIMARY KEY *aAusdruck3* TAG *Indexname2* Fügt zu der Tabelle einen Primärindex hinzu. *aAusdruck3* gibt den Ausdruck des Primärindexes und *Indexname2* gibt den Namen des Primärindexes an. Der Name eines Indexes kann aus bis zu 10 Zeichen bestehen. Ist TAG *Indexname2* nicht angegeben und *aAusdruck3* gleich dem Namen eines Feldes, hat der Primärindex denselben Namen wie das in *aAusdruck3* angegebene Feld.

DROP PRIMARY KEY Löscht den Primärindex sowie dessen Indexnamen. Da eine Tabelle immer nur einen Primärindex (Primärschlüssel) haben kann, muß der Name des Primärindexes nicht angegeben werden. Wird ein Primärindex gelöscht, werden auch alle persistenten Beziehungen gelöscht, die auf diesem Primärindex basieren.

ADD UNIQUE *aAusdruck4* [TAG *Indexname3*] Fügt zu der Tabelle einen potentiellen Primärindex hinzu. *aAusdruck4* gibt den Ausdruck des potentiellen Primärschlüssels, *Indexname3* den Namen des potentiellen Primärindexes an. Der Name eines Indexes kann aus bis zu 10 Zeichen bestehen. Wenn Sie TAG *Indexname3* nicht angeben und *aAusdruck4* nur aus dem Namen eines Feldes besteht, hat der potentielle Primärindex denselben Namen wie das in *aAusdruck4* angegebene Feld.

DROP UNIQUE TAG *Indexname4* Löscht einen potentiellen Primärindex sowie dessen Indexnamen. Da eine Tabelle mehrere potentielle Primärindizes haben kann, müssen Sie den Namen des potentiellen Primärindexes angeben.

ADD FOREIGN KEY [*aAusdruck5*] TAG *Indexname4* Fügt einen Fremdschlüssel zu der Tabelle hinzu. *aAusdruck5* gibt den Ausdruck des Fremdschlüssels, *Indexname4* den Namen des Fremdindexes an. Der Name eines Indexes kann aus bis zu 10 Zeichen bestehen.

REFERENCES *Tabellenname2* [TAG *Indexname5*] Gibt die Master-Tabelle an, zu der eine persistente Beziehung hergestellt werden soll. Soll die Beziehung auf einem bereits vorhandenen Index der Master-Tabelle basieren, müssen Sie TAG *Indexname5* angeben. Der Name eines Indexes kann aus bis zu 10 Zeichen bestehen. Wenn Sie TAG *Indexname5* nicht angeben, wird die Beziehung über den Primärindex der Master-Tabelle hergestellt.

DROP FOREIGN KEY TAG *Indexname6* [SAVE] Löscht einen Fremdschlüssel, der in dem Index *Indexname6* verwaltet wird. Wenn Sie SAVE nicht angeben, wird der Index aus der strukturierten Indexdatei gelöscht. Wenn Sie dies verhindern möchten, müssen Sie SAVE angeben.

RENAME COLUMN *Feldname4* TO *Feldname5* Ermöglicht es Ihnen, den Namen eines Feldes der Tabelle zu ändern. *Feldname4* gibt den Namen des Feldes an, das umbenannt werden soll.

Feldname5 gibt den neuen Namen des Feldes an.

Vorsicht Gehen Sie äußerst sorgfältig vor, wenn Sie Tabellenfelder umbenennen. Es kann sein, daß in Indexausdrücken, feld- und tabellenbezogenen Gültigkeitsregeln, Befehlen, Funktionen usw. auf die ursprünglichen Feldnamen verwiesen wird.

NOVALIDATE Gibt an, daß Visual FoxPro bezüglich der Struktur der Tabelle Änderungen gestattet, die eventuell die Integrität der in der Tabelle abgelegten Daten verletzen. Standardmäßig verhindert Visual FoxPro, daß mit einem ALTER TABLE-Befehl die Struktur einer Tabelle so geändert wird, daß die Integrität der in der Tabelle abgelegten Daten beschädigt wird. Wenn Sie dieses standardmäßige Verhalten außer Kraft setzen möchten, müssen Sie NOVALIDATE angeben.

Bemerkungen

Mit ALTER TABLE kann die Struktur einer Tabelle geändert werden, die noch nicht zu einer Datenbank hinzugefügt wurde. Visual FoxPro meldet einen Fehler, wenn Sie die Struktur einer freien Tabelle ändern und dabei eine der DEFAULT-, FOREIGN KEY-, PRIMARY KEY-, REFERENCES- oder SET-Klauseln angeben.

Ein ALTER TABLE-Befehl legt die jeweilige Tabelle eventuell neu an, indem er einen neuen Tabellenvorspann erstellt und neue Einträge an den Tabellenvorspann anfügt. Zum Beispiel wird eine Tabelle neu angelegt, wenn der Typ oder die Breite eines Feldes geändert wurde.

Sobald eine Tabelle neu erstellt ist, werden für jedes Feld, dessen Typ oder Breite sich geändert hat, die feldbezogenen Gültigkeitsregeln angewendet. Wenn Sie den Typ oder die Breite eines Feldes einer Tabelle ändern, wird die tabellenbezogene Gültigkeitsregel angewendet.

Wenn Sie Änderungen an den feldbezogenen Gültigkeitsregeln oder an der tabellenbezogenen Gültigkeitsregel einer Tabelle vornehmen, die bereits Datensätze enthält, gleicht Visual FoxPro die vorhandenen Daten gegen die neuen Regeln ab und gibt eine Warnung aus, sobald es die erste Verletzung einer feld- oder tabellenbezogenen Gültigkeitsregel oder eine Triggerverletzung entdeckt hat.

Wenn sich die Tabelle, an der Sie Änderungen vornehmen, in einer Datenbank befindet, ist bei Verwendung von ALTER TABLE - SQL die exklusive Nutzung der Datenbank erforderlich. Um eine Datenbank zur exklusiven Nutzung zu öffnen, müssen Sie OPEN DATABASE mit EXCLUSIVE angeben.

Siehe auch

CREATE TABLE - SQL

INDEX

CREATE TABLE - SQL-Befehl

Siehe auch

Erstellt eine Tabelle mit den angegebenen Feldern.

Der Visual FoxPro-ODBC-Treiber unterstützt die Visual FoxPro-spezifische Sprachsyntax dieses Befehls. Treiberspezifische Informationen finden Sie unter Treiberhinweise weiter unten.

Syntax

CREATE TABLE | DBF *Tabellenname1* [NAME *LangerTabellenname*]
[FREE] (*Feldname1 Feldtyp* [(*nFeldbreite* [, *nDezimalstellen*])]
[NULL | NOT NULL]
[CHECK *lAusdruck1* [ERROR *zMeldung1*]]
[DEFAULT *aAusdruck1*]
[PRIMARY KEY | UNIQUE]
[REFERENCES *Tabellenname2* [TAG *Indexname1*]]
[NOCPTRANS]
[, *Feldname2* ...]
[, PRIMARY KEY *aAusdruck2* TAG *Indexname2*
|, UNIQUE *aAusdruck3* TAG *Indexname3*]
[, FOREIGN KEY *aAusdruck4* TAG *Indexname4* [NODUP]
REFERENCES *Tabellenname3* [TAG *Indexname5*]]
[, CHECK *lAusdruck2* [ERROR *zMeldung2*]])
| FROM ARRAY *Datenfeldname*

Argumente

CREATE TABLE | DBF *Tabellenname1* Gibt den Namen der zu erstellenden Tabelle an. Die Optionen TABLE und DBF sind identisch.

NAME *LangerTabellenname* Gibt einen langen Namen für die Tabelle an. Einen langen Tabellennamen können Sie nur angeben, wenn eine Datenbank geöffnet ist, da lange Tabellennamen in Datenbanken gespeichert werden.

Lange Namen können aus bis zu 128 Zeichen bestehen und können in Datenbanken anstelle der kurzen Dateinamen verwendet werden.

FREE Gibt an, daß die Tabelle zu keiner offenen Datenbank hinzugefügt werden soll. FREE ist nicht erforderlich, wenn keine Datenbank geöffnet ist.

(Feldname1 Feldtyp [(*nFeldbreite* [, *nDezimalstellen*])] Gibt den Feldnamen, den Feldtyp, die Feldbreite und die Genauigkeit der Feldwerte (Anzahl der Dezimalstellen) an.

Feldtyp gibt in Form eines einzelnen Buchstabens den Datentyp des Feldes an. Bei einigen Felddatentypen müssen Sie *nFeldbreite* oder *nDezimalstellen* oder beides angeben.

nFeldbreite und *nDezimalstellen* werden ignoriert, wenn Sie als Feldtyp D, T, I, Y, L, M, oder G angeben. Für *nDezimalstellen* gilt standardmäßig der Wert 0 (keine Dezimalstellen), wenn Sie für die Feldtypen N, F und B *nDezimalstellen* nicht angeben.

NULL Gibt an, daß das Feld Nullwerte enthalten kann.

NOT NULL Gibt an, daß das Feld keine Nullwerte enthalten darf.

Wenn Sie NULL und NOT NULL nicht angeben, bestimmt die aktuelle Einstellung von SET NULL, ob im Feld Nullwerte zulässig sind. Wenn NULL und NOT NULL fehlen und Sie die PRIMARY KEY- oder UNIQUE-Klausel angeben, wird dagegen die aktuelle Einstellung von SET NULL ignoriert, und der Standardwert NOT NULL wird für das Feld verwendet.

CHECK *lAusdruck1* Gibt eine Gültigkeitsregel für das Feld an. *lAusdruck1* kann eine benutzerdefinierte Funktion sein. Beachten Sie bitte, daß beim Anfügen eines leeren Datensatzes geprüft wird, ob die Gültigkeitsregel erfüllt ist. Ein Fehler wird angezeigt, wenn die Gültigkeitsregel

in den Feldern eines angefügten Datensatzes keine Leerwerte zuläßt.

ERROR *zMeldung1* Gibt die Fehlermeldung an, die Visual FoxPro anzeigt, wenn die Gültigkeitsregel für das Feld verletzt und ein Fehler erzeugt wird. Die Meldung wird nur angezeigt, wenn Daten in einem Datenblatt- oder Bearbeitungsfenster bearbeitet werden.

DEFAULT *aAusdruck1* Gibt einen Standardwert für das Feld an. Der Datentyp von *aAusdruck1* muß mit dem Datentyp des Feldes identisch sein.

PRIMARY KEY Erstellt einen Primärindex für das Feld. Der Primärindex hat denselben Namen wie das Feld.

UNIQUE Erstellt einen potentiellen Primärindex für das Feld. Der potentielle Primärindex hat denselben Namen wie das Feld.

Anmerkung Potentielle Primärindizes, die durch Angeben der Option UNIQUE im Befehl CREATE TABLE oder ALTER TABLE - SQL erstellt wurden, sind nicht mit den Indizes identisch, die mit der Option UNIQUE des Befehls INDEX erstellt wurden. Bei einem Index, der mit der Option UNIQUE des Befehls INDEX erstellt wurde, sind doppelt vorhandene Indexwerte zulässig, bei potentiellen Primärindizes sind doppelt vorhandene Indexwerte nicht zulässig. Weitere Informationen zur Klausel UNIQUE des Befehls INDEX finden Sie unter INDEX.

Nullwerte und doppelt vorhandene Datensätze sind in einem Feld, das als Primärindex oder als potentieller Primärindex verwendet wird, nicht zulässig. Trotzdem erzeugt Visual FoxPro keinen Fehler, wenn Sie einen Primärindex oder einen potentiellen Primärindex für ein Feld erstellen, das Nullwerte unterstützt. Visual FoxPro erzeugt erst dann einen Fehler, wenn Sie versuchen, einen Nullwert oder einen bereits vorhandenen Wert in ein Feld einzugeben, das als Primärindex oder als potentieller Primärindex verwendet wird.

REFERENCES *Tabellenname2* [TAG *Indexname1*] Gibt die Master-Tabelle an, zu der eine persistente Beziehung hergestellt wird. Wenn Sie TAG *Indexname1*nicht angeben, wird die Beziehung mit Hilfe des Primärindexes der Master-Tabelle hergestellt. Wenn die Master-Tabelle keinen Primärindex hat, erzeugt Visual FoxPro einen Fehler.

Geben Sie TAG *Indexname1* an, um eine Beziehung herzustellen, die auf einem vorhandenen Index der Master-Tabelle basiert. Indexnamen können aus bis zu 10 Zeichen bestehen.

Die Master-Tabelle darf keine freie Tabelle sein.

NOCPTRANS Unterbindet die Umsetzung von Zeichen- und Memo-Feldern in eine andere Codeseite. Wird die Tabelle in eine andere Codeseite konvertiert, bleiben die Felder, für die Sie NOCPTRANS angegeben haben, von der Umsetzung ausgenommen. NOCPTRANS kann nur für Zeichen- und Memo-Felder angegeben werden.

Im folgenden Beispiel wird eine Tabelle mit dem Namen TABELLE1 erstellt. Diese Tabelle enthält zwei Zeichen- und zwei Memo-Felder. Für das zweite Zeichen-Feld, ZEICHEN2, und das zweite Memo-Feld, MEMO2, wurde NOCPTRANS angegeben, um die Umsetzung zu unterbinden.

```
CREATE TABLE tabelle1 (zeichen1 C(10), zeichen2 C(10) NOCPTRANS,;
   memo1 M, memo2 M NOCPTRANS)
```

PRIMARY KEY *aAusdruck2* TAG *Indexname2* Gibt einen zu erstellenden Primärindex an. *aAusdruck2* gibt ein beliebiges Feld oder eine beliebige Kombination von Feldern der Tabelle an. TAG *Indexname2* gibt den Namen des Primärindexes an, der erstellt wird. Indexnamen können aus bis zu 10 Zeichen bestehen.

Da eine Tabelle immer nur einen Primärindex haben kann, dürfen Sie diese Klausel nicht angeben, wenn Sie bereits einen Primärindex für ein Feld erstellt haben. Visual FoxPro erzeugt einen Fehler, wenn Sie mehr als eine PRIMARY KEY-Klausel in einem CREATE TABLE-Befehl angeben.

UNIQUE *aAusdruck3* TAG *Indexname3* Erstellt einen potentiellen Primärindex. *aAusdruck3* gibt ein beliebiges Feld oder eine beliebige Kombination von Feldern der Tabelle an. Wenn Sie jedoch bereits einen Primärindex mit einer der PRIMARY KEY-Klauseln erstellt haben, dürfen Sie das Feld, das für den Primärindex angegeben wurde, hier nicht erneut angeben. TAG *Indexname3* gibt einen Namen für den potentiellen Primärindex an, der erstellt wird. Indexnamen können aus bis zu 10 Zeichen bestehen.

Eine Tabelle darf mehrere potentielle Primärindizes haben.

FOREIGN KEY *aAusdruck4* TAG *Indexname4* [NODUP] Erstellt einen Fremdindex (keinen Primärindex) und stellt eine Beziehung zu einer Master-Tabelle her. *aAusdruck4* gibt den Fremdindexausdruck und *Indexname4* den Namen des Fremdindexes an, der erstellt wird. Indexnamen können aus bis zu 10 Zeichen bestehen. Geben Sie NODUP an, um einen Fremdindex zu erstellen, der ein potentieller Primärindex ist.

Sie können für die Tabelle mehrere Fremdindizes erstellen, aber in den Fremdindexausdrücken müssen unterschiedliche Felder der Tabelle angegeben werden.

REFERENCES *Tabellenname3* [TAG *Indexname5*] Gibt die Master-Tabelle an, zu der eine persistente Beziehung hergestellt wird. Geben Sie TAG *Indexname5* an, um eine Beziehung herzustellen, die auf einem Index der Master-Tabelle basiert. Indexnamen können aus bis zu 10 Zeichen bestehen. Wenn Sie TAG *Indexname5* nicht angeben, wird die Beziehung standardmäßig mit Hilfe des Primärindexes der Master-Tabelle hergestellt.

CHECK *aAusdruck2* [ERROR *zMeldung2*] Gibt die Gültigkeitsregel für die Tabelle an. ERROR *zMeldung2* gibt die Fehlermeldung an, die Visual FoxPro anzeigt, wenn die Gültigkeitsregel der Tabelle verletzt wird. Die Meldung wird nur angezeigt, wenn Daten in einem Datenblatt- oder Bearbeitungsfenster bearbeitet werden.

FROM ARRAY *Datenfeldname* Gibt den Namen eines vorhandenen Datenfelds an, dessen Inhalt der Name, der Typ, die Feldbreite und die Anzahl der Dezimalstellen der einzelnen Felder der Tabelle ist. Der Inhalt des Datenfelds kann mit der Funktion AFIELDS( ) definiert werden.

Bemerkungen

Die neue Tabelle wird in dem Arbeitsbereich mit der niedrigsten verfügbaren Nummer geöffnet, und der Zugriff auf die Tabelle kann über ihren Aliasnamen erfolgen. Die neue Tabelle wird immer zur exklusiven Benutzung geöffnet, unabhängig von der aktuellen Einstellung von SET EXCLUSIVE.

Wenn eine Datenbank geöffnet ist und Sie die FREE-Klausel nicht angeben, wird die neue Tabelle zur Datenbank hinzugefügt. Sie können keine neue Tabelle mit demselben Namen wie eine Tabelle in der Datenbank erstellen.

Wenn eine Datenbank geöffnet ist, benötigt CREATE TABLE-SQL exklusiven Zugriff auf die Datenbank. Geben Sie in OPEN DATABASE das Schlüsselwort EXCLUSIVE an, um eine Datenbank zur exklusiven Benutzung zu öffnen.

Wenn beim Erstellen der neuen Tabelle keine Datenbank geöffnet ist und Sie die NAME-, CHECK-, DEFAULT-, FOREIGN KEY-, PRIMARY KEY- oder REFERENCES-Klauseln angeben, wird ein Fehler erzeugt.

Beachten Sie, daß in der CREATE TABLE-Syntax bestimmte CREATE TABLE-Optionen durch Kommas getrennt sind. Außerdem ist zu beachten, daß NULL-, NOT NULL-, CHECK-, DEFAULT-, PRIMARY KEY- und UNIQUE-Klauseln innerhalb der Klammern stehen müssen, die die Spaltendefinitionen umschließen.

Treiberhinweise

Wenn Ihre Anwendung eine in ODBC SQL formulierte CREATE TABLE-Anweisung an eine Visual FoxPro-Datenquelle sendet, wandelt der Visual FoxPro-ODBC-Treiber diese Anweisung entsprechend der folgenden Syntax in einen Visual FoxPro CREATE TABLE-Befehl um:

ODBC-Syntax	**Visual FoxPro-Syntax**
CREATE TABLE *base-table-name* (*column-identifier data type* [NOT NULL] [,*column-identifier data type* [NOT NULL] ...)	CREATE TABLE *Tabellenname1* [NAME *LangerTabellenname*] (*Feldname1 Feldtyp* [(*nFeldbreite* [, *nDezimalstellen*])] [NOT NULL])

Wenn Sie den Treiber verwenden, um eine Tabelle zu erstellen, schließt der Treiber die Tabelle,

unmittelbar nachdem er sie erstellt hat. Dadurch ist es anderen Benutzern möglich, auf die Tabelle zuzugreifen. In diesem Punkt geht der Treiber anders vor als Visual FoxPro, das eine neue Tabelle nach ihrer Erstellung im Exklusivmodus geöffnet läßt. Wird über Ihre Datenquelle allerdings eine gespeicherte Prozedur ausgeführt, die eine CREATE TABLE-Anweisung enthält, bleibt die neue Tabelle geöffnet.

Ist die Datenquelle eine Datenbank (.DBC-Datei), erstellt der Visual FoxPro-ODBC-Treiber eine Tabelle, deren Name *LangerTabellenname* mit dem Namen *base-table-name* identisch ist.

Verwenden von DDL (Data Definition Language)

An folgenden Stellen dürfen Sie keine DDL-Anweisungen angeben:

- In gestapelten SQL-Anweisungen (batched SQL statements), die eine Transaktion erfordern.
- Im Anschluß an eine gerade ausgeführte Anweisung, wenn folgendes zutrifft: die Anweisung erfordert eine Transaktion; es wird nicht im Modus Automatisch Übernehmen (auto-commit mode, Auto-Commit-Modus) gearbeitet; und die Anwendung hat noch nicht **SQLTransact** aufgerufen.

Wenn Sie z.B. eine temporäre Datei erstellen möchten, sollten Sie diese Tabelle erstellen, bevor Sie die Anweisung ausgeben, die eine Transaktion erfordert. Wenn Sie die entsprechende CREATE TABLE-Anweisung in eine gestapelte SQL-Anweisung einfügen, die eine Transaktion erfordert, meldet der Treiber einen Fehler.

Siehe auch

ALTER TABLE - SQL

Datentypen

INSERT - SQL

SELECT - SQL

DELETE - SQL-Befehl

Siehe auch

Markiert Datensätze als gelöschte Datensätze.

Der Visual FoxPro-ODBC-Treiber unterstützt die Visual FoxPro-spezifische Sprachsyntax dieses Befehls. Treiberspezifische Informationen finden Sie unter Treiberhinweise weiter unten.

Syntax

DELETE FROM [*Datenbankname!*]*Tabellenname*
[WHERE *Filterbedingung1* [AND | OR *Filterbedingung2* ...]]

Argumente

FROM [*Datenbankname!*]*Tabellenname* Gibt die Tabelle an, in der Datensätze zum Löschen markiert werden sollen.

Datenbankname! gibt den Namen der Datenbank an, zu der die Tabelle gehört. Diese Angabe ist erforderlich, wenn die Datenbank, zu der die Tabelle gehört, nicht mit der Datenbank identisch ist, die für die Datenquelle angegeben ist. Setzen Sie das Ausrufezeichen (!) als Trennzeichen zwischen den Datenbank- und den Tabellennamen.

WHERE *Filterbedingung1* [AND | OR *Filterbedingung2* ...] Gibt an, daß Visual FoxPro nur bestimmte Datensätze zum Löschen markieren soll.

Filterbedingung gibt die Kriterien an, die Datensätze erfüllen müssen, um zum Löschen markiert zu werden. Sie können beliebig viele Filterbedingungen angeben und diese mit den Operatoren AND oder OR verbinden. Sie können auch den Operator NOT verwenden, um den Wert eines logischen Ausdrucks umzukehren, oder EMPTY( ), um auf ein leeres Feld zu überprüfen.

Bemerkungen

Wenn SET DELETED auf ON gesetzt ist, werden die zum Löschen markierten Datensätze von Befehlen mit einem Gültigkeitsbereich ignoriert.

DELETE - SQL arbeitet mit Datensatzsperrung, wenn in Tabellen, die für gemeinsamen Zugriff geöffnet sind, mehrere Datensätze zum Löschen markiert werden. Dadurch werden Konflikte beim Zugriff auf Datensätze in Netzwerkumgebungen reduziert, das Leistungsverhalten kann sich dadurch jedoch verschlechtern. Das beste Leistungsverhalten erzielen Sie, wenn Sie die Tabelle zur exklusiven Benutzung öffnen.

Treiberhinweise

Wenn Ihre Anwendung eine in ODBC SQL formulierte DELETE-Anweisung an die Datenquelle sendet, wandelt der Visual FoxPro-ODBC-Treiber den Befehl in einen Visual FoxPro-DELETE-Befehl um, ohne sonstige Änderungen vorzunehmen.

Siehe auch

SET DELETED

DELETE TAG-Befehl

Siehe auch

Löscht einen oder mehrere Indexnamen aus einer Mehrfachindexdatei (.CDX).

Syntax

DELETE TAG *Indexname1* [OF *CDXDateiname1*]
[, *Indexname2* [OF *CDXDateiname2*]] ...

- oder -

DELETE TAG ALL [OF *CDXDateiname*]

Argumente

Indexname1 OF *CDXDateiname1* [, *Indexname2*
[OF *CDXDateiname2*]] ... Gibt den Indexnamen an, der aus einer Mehrfachindexdatei gelöscht werden soll. Sie können mehrere Indexnamen mit einem DELETE TAG-Befehl gleichzeitig löschen, indem Sie eine Liste von durch Kommas getrennten Indexnamen angeben. Enthalten die offenen Indexdateien zwei oder mehr identische Indexnamen, können Sie einen Indexnamen aus einer bestimmten Indexdatei löschen, indem Sie OF *CDXDateiname* angeben.

ALL [OF *CDXDateiname*] Löscht alle Indexnamen aus einer Mehrfachindexdatei. Wenn die aktuelle Tabelle über eine strukturierte Mehrfachindexdatei verfügt, werden alle Indexnamen aus der Indexdatei gelöscht, die Indexdatei wird vom Datenträger gelöscht und das Kennzeichen im Dateikopf, das anzeigt, daß es eine zugehörige strukturierte Mehrfachindexdatei gibt, wird ebenfalls gelöscht. Verwenden Sie ALL zusammen mit OF *CDXDateiname*, um alle Indexnamen aus einer anderen offenen Mehrfachindexdatei als der strukturierten Mehrfachindexdatei zu löschen.

Bemerkungen

Mit INDEX erstellte Mehrfachindexdateien enthalten Indexnamen, die den Indexeinträgen entsprechen. Mit DELETE TAG werden ein oder mehrere Indexnamen aus geöffneten Mehrfachindexdateien gelöscht. Sie können Indexnamen nur aus Mehrfachindexdateien löschen, die im aktuellen Arbeitsbereich geöffnet sind. Wenn Sie alle Indexnamen aus einer Mehrfachindexdatei löschen, wird die gesamte Datei vom Datenträger gelöscht.

Visual FoxPro sucht einen Indexnamen zuerst in der strukturierten Mehrfachindexdatei (sofern eine geöffnet ist). Wird der Indexname dort nicht gefunden, sucht Visual FoxPro den Indexnamen in den anderen offenen Mehrfachindexdateien.

Siehe auch

INDEX

DROP TABLE-Befehl

Löscht eine Tabelle sowohl aus der Datenbank, die in der Datenquelle angegeben ist, als auch von dem Datenträger.

Der Visual FoxPro-ODBC-Treiber unterstützt die Visual FoxPro-spezifische Sprachsyntax dieses Befehls. Treiberspezifische Informationen finden Sie unter Treiberhinweise weiter unten.

Syntax

DROP TABLE *Tabellenname* | *Dateiname* | ?

Einstellungen

Tabellenname Gibt die Tabelle an, die sowohl aus der Datenbank, die in der Datenquelle angegeben ist, als auch vom Datenträger gelöscht werden soll.

Dateiname Gibt die freie Tabelle an, die vom Datenträger gelöscht werden soll.

? Zeigt das Dialogfeld **Löschen** an, in dem Sie den Namen der Tabelle auswählen können, die gelöscht werden soll.

Bemerkungen

Wenn DROP TABLE ausgegeben wird, werden neben der Tabelle auch alle zu ihr gehörigen Primärindizes, Standardwerte und Gültigkeitsregeln gelöscht. DROP TABLE kann sich außerdem auf andere Tabellen der in der Datenquelle angegebenen Datenbank auswirken. Dies ist dann der Fall, wenn es für diese anderen Tabellen Regeln oder Beziehungen gibt, die sich auf die zu löschende Tabelle beziehen. Solche Regeln und Beziehungen sind nicht mehr gültig, sobald die Tabelle aus der Datenbank gelöscht wurde.

Treiberhinweise

Wenn Ihre Anwendung eine in ODBC SQL formulierte DROP TABLE-Anweisung an eine Visual FoxPro-Datenquelle sendet, wandelt der Visual FoxPro-ODBC-Treiber diese Anweisung entsprechend der folgenden Syntax in einen Visual FoxPro-DROP TABLE-Befehl um:

ODBC-Syntax	Datenquelle	Visual FoxPro Syntax
DROP TABLE *base-table-name*	Datenbank (.DBC-Datei)	REMOVE TABLE *Tabellenname* DELETE
	Verzeichnis mit freien Tabellen (.DBF-Dateien)	ERASE *dbfName* ERASE *cdxName* ERASE *fptName*

INDEX-Befehl

Siehe auch

Erstellt eine Indexdatei, so daß das Anzeigen von und der Zugriff auf die Datensätze einer Tabelle in einer logischen Reihenfolge erfolgen kann.

Syntax

INDEX ON *aAusdruck* TO *IDXDateiname* | TAG *Indexname* [OF *CDXDateiname*]
[FOR *lAusdruck*]
[COMPACT]
[ASCENDING | DESCENDING]
[UNIQUE | CANDIDATE]
[ADDITIVE]

Argumente

aAusdruck Gibt einen Indexausdruck (Schlüssel) an, der den Namen eines Feldes oder die Namen mehrerer Felder der aktuellen Tabelle enthalten kann. Für jeden Datensatz der Tabelle wird ein auf dem Indexausdruck basierender Index erstellt und in der Indexdatei gespeichert. Visual FoxPro verwendet diesen Index, um die Datensätze der Tabelle anzuzeigen und darauf zuzugreifen.

Anmerkung *aAusdruck* kann auch eine Speichervariable, ein Datenfeldelement oder ein Feld oder ein Feldausdruck aus einer Tabelle in einem anderen Arbeitsbereich sein. Dies wird allerdings nicht empfohlen. Memo-Felder können nicht allein in Indexdateiausdrücken verwendet werden, sondern müssen mit anderen Zeichenausdrücken kombiniert werden. Wenn Sie auf einen Index zugreifen, der eine Variable oder ein Feld enthält, die bzw. das nicht mehr vorhanden ist oder nicht gefunden wird, zeigt Visual FoxPro eine Fehlermeldung an.

Wenn Sie versuchen, einen Index mit einem Schlüssel variabler Länge zu erstellen, wird der Schlüssel mit Leerzeichen aufgefüllt. Schlüssel variabler Länge werden in Visual FoxPro nicht unterstützt.

Sie können einen Indexschlüssel mit der Länge 0 erstellen. Ein Indexschlüssel mit der Länge 0 wird z.B. erstellt, wenn der Indexausdruck eine Teilzeichenfolge eines leeren Memo-Feldes ist. Ein Indexschlüssel der Länge 0 verursacht einen Fehler. Wenn Visual FoxPro einen Index erstellt, werden die Felder des ersten Datensatzes der Tabelle ausgewertet. Ist ein Feld leer, müssen möglicherweise temporäre Daten in das Feld des ersten Datensatzes eingegeben werden, um zu verhindern, daß ein Indexschlüssel mit der Länge 0 erstellt wird.

TO *IDXDateiname* Erstellt eine .IDX-Indexdatei. Die Indexdatei erhält die Standarderweiterung .IDX.

TAG *Indexname* [OF *CDXDateiname*] Erstellt eine Mehrfachindexdatei. Eine Mehrfachindexdatei ist eine Indexdatei, die beliebig viele Indizes (Indexeinträge) enthalten kann. Jeder Index ist durch einen eindeutigen Indexnamen gekennzeichnet. Indexnamen müssen mit einem Buchstaben oder einem Unterstrich anfangen und können aus einer beliebigen Kombination von bis zu 10 Buchstaben, Zahlen und Unterstrichen bestehen. Die Anzahl der Indizes in einer Mehrfachindexdatei ist nur durch den verfügbaren Arbeitsspeicher und Speicherplatz begrenzt.

Mehrfachindexdateien werden immer komprimiert, so daß Sie beim Erstellen einer solchen Datei COMPACT nicht explizit angeben müssen. Mehrfachindexdateien erhalten die Dateinamenerweiterung .CDX.

Es können zwei Typen von Mehrfachindexdateien erstellt werden, strukturierte und nichtstrukturierte.

Strukturierte Mehrfachindexdateien Sie können mit TAG *Indexname* eine strukturierte Mehrfachindexdatei erstellen, indem Sie die optionale Klausel OF *CDXDateiname* nicht angeben. Eine strukturierte Mehrfachindexdatei hat immer denselben Namen wie die Tabelle und wird beim Öffnen der Tabelle automatisch geöffnet.

Nichtstrukturierte Mehrfachindexdateien Sie erstellen eine nichtstrukturierte

Mehrfachindexdatei, indem Sie nach OF *CDXDateiname* die Klausel TAG *Indexname* angeben. Im Gegensatz zu einer strukturierten Mehrfachindexdatei muß eine nichtstrukturierte Mehrfachindexdatei explizit mit SET INDEX oder der INDEX-Klausel in USE geöffnet werden.

Wurde bereits eine Mehrfachindexdatei erstellt und geöffnet, wird ein Index an die Mehrfachindexdatei angefügt, wenn Sie INDEX mit TAG *Indexname* eingeben.

FOR *lAusdruck* Gibt eine Bedingung an. Es können nur die Datensätze angezeigt und es kann nur auf die Datensätze zugegriffen werden, die die Filterbedingung *lAusdruck* erfüllen. In der Indexdatei werden nur für die Datensätze Indexwerte erstellt, die die Filterbedingung erfüllen.

Ein INDEX ... FOR *lAusdruck*-Befehl wird mit der Rushmore-Technologie optimiert, wenn *lAusdruck* ein optimierbarer Ausdruck ist. Das beste Leistungsverhalten erzielen Sie, wenn Sie in der FOR-Klausel einen optimierbaren Ausdruck verwenden.

COMPACT Erstellt eine komprimierte .IDX-Datei.

ASCENDING Gibt an, daß die .CDX-Datei in aufsteigender Reihenfolge erstellt wird. Standardmäßig werden .CDX-Indizes in aufsteigender Reihenfolge erstellt. Sie können ASCENDING als Hinweis auf die Indexreihenfolge der Indexdatei angeben. Geben Sie DESCENDING an, wenn Sie eine Tabelle in absteigender Reihenfolge indizieren möchten.

DESCENDING Gibt an, daß die .CDX-Datei in absteigender Reihenfolge erstellt wird. Beim Erstellen von .IDX-Dateien können Sie DESCENDING nicht angeben.

UNIQUE Gibt an, daß von mehreren Datensätzen, die denselben Indexwert haben, nur der erste Datensatz in einer .IDX-Datei oder in einem .CDX-Index berücksichtigt werden soll. Mit UNIQUE können Sie verhindern, daß Datensätze mit doppelt vorhandenem Schlüsselwert angezeigt werden oder daß auf solche Datensätze zugegriffen wird. Weitere Datensätze mit identischem Indexwert werden nicht in die Indexdatei aufgenommen. Die UNIQUE-Option des Befehls INDEX ist mit der Ausführung von SET UNIQUE ON vor INDEX oder REINDEX identisch.

Wenn ein mit UNIQUE erstellter Index (als Einfachindexdatei oder aus einer Mehrfachindexdatei) aktiv ist und ein Datensatz mit doppelt vorhandenem Indexwert so geändert wird, daß sich sein Indexwert ändert, wird der Index aktualisiert. Es ist dann jedoch nicht möglich, auf den nächsten Datensatz mit dem ursprünglich doppelt vorhandenen Indexwert zuzugreifen oder diesen Datensatz anzuzeigen, bis die Datei mit REINDEX neu indiziert wurde.

CANDIDATE Erstellt einen strukturierten potentiellen Primärindex. Das Schlüsselwort CANDIDATE kann nur beim Erstellen eines Indexes verwendet werden, der in einer strukturierten Mehrfachindexdatei verwaltet wird. Andernfalls zeigt Visual FoxPro eine Fehlermeldung an.

Bei einem mit CANDIDATE erstellten Index kann das Feld oder die Kombination von Feldern, das bzw. die im Indexausdruck *aAusdruck* angegeben wurden, keine Duplikatwerte enthalten. Da bei einem mit CANDIDATE erstellten Index Duplikatwerte nicht möglich sind, ist ein solcher Index gewissermaßen ein "Kandidat" für einen Primärindex.

Visual FoxPro erzeugt einen Fehler, wenn Sie einen potentiellen Primärindex für ein Feld oder eine Kombination von Feldern erstellen, die bereits Duplikatwerte enthalten.

ADDITIVE Bewirkt, daß alle zuvor geöffneten Indexdateien geöffnet bleiben. Wenn Sie beim Erstellen einer oder mehrerer Indexdateien für eine Tabelle mit INDEX die ADDITIVE-Klausel nicht angeben, werden alle zuvor geöffneten Indexdateien (außer der strukturierten Mehrfachindexdatei) geschlossen.

Bemerkungen

Datensätze einer Tabelle, für die es eine Indexdatei gibt, werden in der Reihenfolge angezeigt, und es wird in der Reihenfolge auf sie zugegriffen, die durch den Indexausdruck vorgegeben ist. Die physikalische Reihenfolge der Datensätze in der Tabelle ändert sich durch eine Indexdatei nicht.

Indextypen

In Visual FoxPro können Sie zwei Typen von Indexdateien erstellen:

- Strukturierte .CDX-Indexdateien, die mehrere Indizes enthalten.

- .IDX-Indexdateien, die nur einen Index enthalten.

Außerdem können Sie eine strukturierte Mehrfachindexdatei erstellen, die automatisch zusammen mit der Tabelle geöffnet wird.

Tip Da strukturierte Mehrfachindexdateien automatisch zusammen mit der Tabelle geöffnet werden, ist dieser Indextyp besonders beliebt.

Geben Sie COMPACT an, um komprimierte .IDX-Indexdateien zu erstellen. Mehrfachindexdateien werden immer komprimiert.

Indexreihenfolge sowie Aktualisieren von Indizes

Die Reihenfolge, in der die Datensätze der Tabelle angezeigt oder in der auf deren Datensätze zugegriffen wird, wird von nur einer Indexdatei (der Hauptindexdatei) oder einem Index (dem Hauptindex) gesteuert. Bestimmte Befehle (z.B. SEEK) verwenden die Hauptindexdatei oder den Hauptindex bei der Suche nach Datensätzen. Es werden jedoch alle offenen .IDX- und .CDX-Indexdateien aktualisiert, wenn an der Tabelle Änderungen vorgenommen werden.

Benutzerdefinierte Funktionen (UDF = User-Defined Functions)

Ein Indexausdruck kann zwar eine benutzerdefinierte Funktion enthalten, Sie sollten in Indexausdrücken jedoch keine benutzerdefinierten Funktionen einsetzen. Durch Einbeziehen einer benutzerdefinierten Funktion in einen Indexausdruck verlängert sich die Zeit, die zum Erstellen und Aktualisieren des Indexes benötigt wird. Außerdem werden Indexaktualisierungen möglicherweise nicht ausgeführt, wenn eine benutzerdefinierte Funktion in einem Indexausdruck verwendet wird.

Wenn Sie eine benutzerdefinierte Funktion in einem Indexausdruck verwenden, muß Visual FoxPro die benutzerdefinierte Funktion finden können. Wenn Visual FoxPro einen Index erstellt, wird der Indexausdruck in der Indexdatei gespeichert, aber der Indexausdruck enthält nur einen Verweis auf die benutzerdefinierte Funktion.

Siehe auch

ALTER TABLE - SQL

DELETE TAG

SET COLLATE

SET UNIQUE

INSERT - SQL-Befehl

Siehe auch

Fügt an das Ende einer Tabelle einen Datensatz an, der die angegebenen Feldwerte enthält.

Der Visual FoxPro-ODBC-Treiber unterstützt die Visual FoxPro-spezifische Sprachsyntax dieses Befehls. Treiberspezifische Informationen finden Sie unter Treiberhinweise weiter unten.

Syntax

INSERT INTO *dbf_Name* [(*Feldname1* [, *Feldname2*, ...])]
VALUES (*aAusdruck1* [, *aAusdruck2*, ...])

Argumente

INSERT INTO *dbf_Name* Gibt den Namen der Tabelle an, an die der neue Datensatz angefügt wird. *dbf_Name* kann einen Pfad enthalten und ein Namensausdruck sein.

Wenn die angegebene Tabelle nicht geöffnet ist, wird sie in einem neuen Arbeitsbereich exklusiv geöffnet, und der neue Datensatz wird angefügt. Der neue Arbeitsbereich wird nicht ausgewählt, sondern der aktuelle Arbeitsbereich bleibt ausgewählt.

Wenn die angegebene Tabelle offen ist, fügt INSERT den neuen Datensatz an die Tabelle an. Wenn die Tabelle in einem anderen Arbeitsbereich als dem aktuellen geöffnet ist, wird dieser nach dem Anfügen des Datensatzes nicht ausgewählt, sondern der aktuelle Arbeitsbereich bleibt ausgewählt.

[(*Feldname1* [, *Feldname2* [, ...]])] Gibt die Namen der Felder des neuen Datensatzes an, in die die Werte eingefügt werden.

VALUES (*aAusdruck1* [, *aAusdruck2* [, ...]]) Gibt die Feldwerte an, die in den neuen Datensatz eingefügt werden. Wenn Sie die Feldnamen nicht angeben, müssen Sie Feldwerte in der Reihenfolge angeben, die durch die Struktur der Tabelle vorgegeben ist.

Bemerkungen

Der neue Datensatz enthält die Daten, die in der VALUES-Klausel aufgelistet sind.

Treiberhinweise

Wenn Ihre Anwendung eine in ODBC SQL formulierte INSERT-Anweisung an die Datenquelle sendet, wandelt der Visual FoxPro-ODBC-Treiber den Befehl in einen Visual FoxPro-INSERT-Befehl um, ohne sonstige Änderungen vorzunehmen.

Siehe auch

CREATE TABLE - SQL

SELECT - SQL

SELECT - SQL-Befehl

Siehe auch

Ruft Daten aus einer oder mehreren Tabellen ab.

Der Visual FoxPro-ODBC-Treiber unterstützt die Visual FoxPro-spezifische Sprachsyntax dieses Befehls. Treiberspezifische Informationen finden Sie unter Treiberhinweise weiter unten.

Syntax

SELECT [ALL | DISTINCT]
[*Tabellenaliasname*.] *Select_Element* [AS *Spaltenname*]
[, [*Tabellenaliasname*.] *Select_Element* [AS *Spaltenname*] ...]
FROM [*Datenbankname!*]*Tabelle* [*Lokaler_Aliasname*]
[, [*Datenbankname!*]*Tabelle* [*Lokaler_Aliasname*] ...]
[WHERE *Verknüpfungsbedingung* [AND *Verknüpfungsbedingung*
…]
[AND | OR *Filterbedingung* [AND | OR *Filterbedingung* ...]]]]
[GROUP BY *Gruppenspalte* [, *Gruppenspalte* ...]]
[HAVING *Filterbedingung*]
[UNION [ALL] *SELECTBefehl*]
[ORDER BY *Sortierobjekt* [ASC | DESC] [, *Sortierobjekt* [ASC | DESC] ...]]

Argumente

[ALL | DISTINCT]
[*Tabellenaliasname*.] *Select_Element* [AS *Spaltenname*]
[, [*Tabellenaliasname*.] *Select_Element* [AS *Spaltenname*] ...] Die SELECT-Klausel gibt die Felder, Konstanten und Ausdrücke an, die in den Abfrageergebnissen angezeigt werden sollen.

ALL, die Standardeinstellung, zeigt alle Zeilen in den Abfrageergebnissen an.

DISTINCT sorgt dafür, daß in den Abfrageergebnissen keine Duplikatzeilen vorkommen.

Anmerkung Pro SELECT-Klausel können Sie DISTINCT nur einmal verwenden.

Tabellenaliasname ermöglicht es, gleiche Elementnamen zu unterscheiden. Für jedes in *Select_Element* angegebene Element wird in den Abfrageergebnissen eine Spalte erstellt. Besitzen mehrere Elemente denselben Namen, müssen Sie vor den Namen der Elemente den jeweiligen Tabellenaliasnamen sowie einen Punkt angeben, um zu verhindern, daß Spalten dupliziert werden.

Select_Element gibt ein Element an, das in den Abfrageergebnissen enthalten ist. Ein Element kann folgendes sein:

- Der Name eines Feldes einer Tabelle, die in der FROM-Klausel angegeben ist.
- Eine Konstante, die angibt, daß der gleiche Konstantenwert in allen Zeilen der Abfrageergebnisse angezeigt werden soll.
- Ein Ausdruck, der der Name einer benutzerdefinierten Funktion sein kann.

Benutzerdefinierte Funktionen mit SELECT

Obwohl es von Vorteil sein kann, benutzerdefinierte Funktionen in der SELECT-Klausel zu verwenden, sollten Sie jedoch die folgenden Einschränkungen berücksichtigen:

- Die Geschwindigkeit von SELECT-Operationen kann durch die Geschwindigkeit begrenzt werden, mit der benutzerdefinierte Funktionen ausgeführt werden. Die Bearbeitung großer Datenmengen, in die benutzerdefinierte Funktionen einbezogen sind, kann u.U. besser durchgeführt werden, wenn Sie API-Funktionen und benutzerdefinierte Funktionen verwenden, die in C oder Assembler geschrieben sind.
- Der einzige zuverlässige Weg, Werte an benutzerdefinierte Funktionen, die über SELECT

aufgerufen werden, zu übergeben, ist der über die Argumentliste, die beim Aufruf der Funktion übergeben wird.

- Wenn Sie beim Ausprobieren verschiedener Dinge auf eine vermutlich unzulässige Vorgehensweise stoßen, die in einer bestimmten Version von FoxPro korrekt funktioniert, ist nicht sichergestellt, daß diese Vorgehensweise auch in späteren Versionen noch korrekt funktionieren wird.

Trotz dieser Einschränkungen sind benutzerdefinierte Funktionen in der SELECT-Klausel akzeptabel. Vergessen Sie dabei jedoch nicht, daß die Verwendung von SELECT die Leistung herabsetzen kann.

Die folgenden Feldfunktionen können in einem Select-Element verwendet werden, das entweder ein Feld oder ein Ausdruck ist, der ein Feld betrifft:

- AVG(*Select_Element*); berechnet den Durchschnittswert der Daten einer numerischen Spalte.
- COUNT(*Select_Element*); ermittelt die Anzahl der in einer Spalte ausgewählten Elemente. COUNT(*) zählt die Anzahl der Zeilen einer Abfrageausgabe.
- MIN(*Select_Element*); ermittelt den kleinsten Wert von *Select_Element* in einer Spalte.
- MAX(*Select_Element*); ermittelt den größten Wert von *Select_Element* in einer Spalte.
- SUM(*Select_Element*); ermittelt die Gesamtsumme einer Spalte, die numerische Daten enthält.

Sie können Feldfunktionen nicht verschachteln.

AS *Spaltenname* Gibt die Überschrift einer Spalte in der Abfrageausgabe an. Dies ist sinnvoll, wenn *Select_Element* ein Ausdruck ist oder eine Feldfunktion enthält und Sie der Spalte einen aussagekräftigen Namen geben möchten. *Spaltenname* kann ein Ausdruck sein, darf aber keine Zeichen enthalten, die in Tabellenfeldnamen nicht erlaubt sind (z.B. Leerzeichen).

FROM [*Datenbankname!*]*Tabelle* [*Lokaler_Aliasname*]
[, [*Datenbankname!*]*Tabelle* [*Lokaler_Aliasname*] ...] Listet die Tabellen auf, die die von der Abfrage abzurufenden Daten enthalten. Ist keine Tabelle geöffnet, zeigt Visual FoxPro das Dialogfeld **Öffnen** an, so daß Sie angeben können, wo sich die Datei befindet. Sobald die Tabelle einmal geöffnet ist, bleibt sie auch nach Beendigung der Abfrage geöffnet.

Datenbankname! gibt den Namen einer Datenbank an, die nicht mit der Datenbank identisch ist, die für die Datenquelle angegeben ist. Sie müssen diesen Namen angeben, wenn die Datenbank, die die Tabelle enthält, und die für die Datenquelle angegebene Datenbank nicht identisch sind. Setzen Sie dabei zwischen den Datenbanknamen und den Tabellennamen das Ausrufezeichen (!) als Trennzeichen.

Lokaler_Aliasname gibt einen temporären Namen für die in *Tabelle* benannte Tabelle an. Bei Angabe eines lokalen Aliasnamens müssen Sie diesen während der gesamten SELECT-Anweisung anstelle des Tabellennamens verwenden. Der lokale Aliasname hat keine Auswirkungen auf die Visual FoxPro-Umgebung.

WHERE *Verknüpfungsbedingung* [AND *Verknüpfungsbedingung* ...]
[AND | OR *Filterbedingung* [AND | OR *Filterbedingung* ...]] Teilt Visual FoxPro mit, daß nur bestimmte Datensätze in den Abfrageergebnissen enthalten sein sollen. WHERE ist erforderlich, wenn Sie Daten aus mehreren Tabellen abfragen möchten.

Verknüpfungsbedingung gibt die Felder an, über die die Tabellen, die in der FROM-Klausel angegeben sind, verbunden werden. Werden in einer Abfrage mehrere Tabellen angegeben, sollten Sie eine Verknüpfungsbedingung für jede Tabelle nach der ersten angeben.

Wichtig Denken Sie beim Erstellen von Verknüpfungsbedingungen an die folgenden Punkte:

- Wenn Sie in einer Abfrage zwei Tabellen ohne Verknüpfungsbedingung angeben, wird jeder Datensatz der ersten Tabelle solange mit jedem Datensatz der zweiten Tabelle verknüpft, wie die Filterbedingungen zutreffen. Hierdurch können sehr umfangreiche Ergebnisse entstehen.
- Beim Verknüpfen von Tabellen, die leere Felder enthalten, ist Vorsicht geboten, da Visual FoxPro auch leere Felder vergleicht. Wenn Sie beispielsweise CUSTOMER.ZIP und INVOICE.ZIP verknüpfen, wobei CUSTOMER 100 leere Postleitzahlen und INVOICE 400 leere Postleitzahlen enthält, werden im Abfrageergebnis aufgrund der leeren Felder 40.000

zusätzliche Datensätze ausgegeben. Verwenden Sie die Funktion EMPTY( ), um leere Datensätze nicht in das Abfrageergebnis aufzunehmen.

Mehrere Verknüpfungsbedingungen müssen mit dem Operator AND verbunden werden. Die Verknüpfungsbedingungen haben folgendes Format:

Feldname1 Vergleich Feldname2

Feldname1 ist der Name eines Feldes einer Tabelle, *Feldname2* ist der Name eines Feldes einer anderen Tabelle, und *Vergleich* ist einer der folgenden Operatoren.

Operator	Vergleich
=	Gleich
==	Genau gleich
LIKE	SQL LIKE
<>, !=, #	Ungleich
>	Größer als
>=	Größer als oder gleich
<	Kleiner als
<=	Kleiner als oder gleich

Wenn Sie den Operator = mit Zeichenfolgen verwenden, hängt die Wirkungsweise von der SET ANSI-Einstellung ab. Ist SET ANSI auf OFF eingestellt, behandelt Visual FoxPro Zeichenfolgenvergleiche genau wie in Xbase. Ist SET ANSI auf ON eingestellt, werden Zeichenfolgenvergleiche entsprechend den ANSI-Regeln behandelt. Weitere Informationen, wie Visual FoxPro Zeichenfolgenvergleiche vornimmt, finden Sie unter SET ANSI und SET EXACT.

Filterbedingung gibt die Kriterien an, die Datensätze erfüllen müssen, um zum Abfrageergebnis hinzugefügt zu werden. Sie können in einer Abfrage beliebig viele Filterbedingungen angeben und diese mit den Operatoren AND und OR verbinden. Sie können auch den Operator NOT zur Umkehrung eines logischen Ausdrucks verwenden oder mit der Option EMPTY( ) nach einem leeren Feld suchen. Für *Filterbedingung* können Sie eines der Formate der folgenden Beispiele verwenden:

Beispiel 1 *Feldname1 Vergleich Feldname2*

```
customer.cust_id = orders.cust_id
```

Beispiel 2 *Feldname Vergleich Expression*

```
payments.amount >= 1000
```

Beispiel 3 *Feldname Vergleich* ALL (*Unterabfrage*)

```
company < ALL ;
(SELECT company FROM customer WHERE country = "Großbritannien")
```

Bei Angabe von ALL in der Filterbedingung muß das Feld die Vergleichsbedingung für alle Werte erfüllen, die von der Unterabfrage festgelegt werden, bevor der Datensatz in die Abfrageergebnisse aufgenommen wird.

Beispiel 4 *Feldname Vergleich* ANY | SOME (*Unterabfrage*)

```
company < ANY ;
(SELECT company FROM customer WHERE country = "Großbritannien")
```

Enthält die Filterbedingung ANY oder SOME, muß das Feld die Vergleichsbedingung für wenigstens einen der Werte erfüllen, die von der Unterabfrage festgelegt werden.

Im folgenden Beispiel wird geprüft, ob die Feldwerte innerhalb des angegebenen Wertebereichs liegen.

Beispiel 5 *Feldname* [NOT] BETWEEN *Anfangsbereich* AND *Endbereich*

```
customer.postalcode BETWEEN 90000 AND 99999
```

Im folgenden Beispiel wird geprüft, ob mindestens eine Zeile die Kriterien der Unterabfrage erfüllt.

Wenn die Filterbedingung EXISTS enthält, wird sie als .T. (wahr) ausgewertet, es sei denn, die Unterabfrage hat als Ergebnis eine leere Menge.

Beispiel 6 [NOT] EXISTS (*Unterabfrage*)

```
EXISTS ;
(SELECT * FROM orders WHERE customer.postalcode =
   orders.postalcode)
```

Beispiel 7 *Feldname* [NOT] IN *Wertegruppe*

```
customer.postalcode NOT IN ("98052","98072","98034")
```

Bei Angabe von IN in der Filterbedingung muß das Feld einen der angegebenen Werte enthalten, bevor der Datensatz zu dem Abfrageergebnis hinzugefügt wird.

Beispiel 8 *Feldname* [NOT] IN (*Unterabfrage*)

```
customer.cust_id IN ;
(SELECT orders.cust_id FROM orders WHERE orders.city="Dortmund")
```

In diesem Fall muß das Feld einen der Werte enthalten, die von der Unterabfrage zurückgegeben werden, bevor der entsprechende Datensatz zu dem Abfrageergebnis hinzugefügt wird.

Beispiel 9 *Feldname* [NOT] LIKE *zAusdruck*

```
customer.country NOT LIKE "Großbritannien"
```

Diese Filterbedingung sucht nach jedem Feld, das mit *zAusdruck* übereinstimmt. Sie können dabei die Platzhalter Prozentzeichen (%) und Unterstrich (_) als Teil von *zAusdruck* angeben. Der Unterstrich entspricht einem unbekannten Zeichen der Zeichenfolge.

GROUP BY *Gruppenspalte* [, *Gruppenspalte* ...] Verwendet die Werte einer oder mehrerer Spalten, um in der Abfrage Gruppen zu bilden. *Gruppenspalte* kann folgendes sein: der Name eines normalen Tabellenfelds, ein Feld, das eine SQL-Feldfunktion enthält, oder ein numerischer Ausdruck, der die Position der Spalte in der Ergebnistabelle angibt. (Die Spalte ganz links hat die Nummer 1.)

HAVING *Filterbedingung* Gibt eine Filterbedingung an, der die Gruppen entsprechen müssen, um zu den Abfrageergebnissen hinzugefügt zu werden. HAVING sollte mit GROUP BY verwendet werden. HAVING kann beliebig viele Filterbedingungen enthalten, die mit den Operatoren AND oder OR verbunden werden. Mit NOT können Sie den Wert eines logischen Ausdrucks umkehren.

Filterbedingung kann keine Unterabfrage enthalten.

HAVING ohne GROUP BY hat dieselbe Auswirkung wie eine WHERE-Klausel. Sie können lokale Aliasnamen und Feldfunktionen in der HAVING-Klausel verwenden. Wenn Ihre HAVING-Klausel keine Feldfunktionen enthält, können Sie eine WHERE-Klausel zur Leistungsverbesserung angeben.

[UNION [ALL] *SELECTBefehl*] Verbindet die Endergebnisse eines SELECT-Befehls mit den Endergebnissen eines anderen SELECT-Befehls. Standardmäßig überprüft UNION die kombinierten Ergebnisse und entfernt doppelte Zeilen. Verwenden Sie Klammern zum Verbinden mehrerer UNION-Klauseln.

ALL verhindert, daß UNION doppelte Zeilen aus kombinierten Ergebnissen entfernt.

Folgende Regeln gelten für UNION-Klauseln:

- Sie können UNION nicht zum Verbinden von Unterabfragen verwenden.
- Beide SELECT-Befehle müssen die gleiche Anzahl von Spalten ausgeben.
- Jede Spalte des Abfrageergebnisses eines SELECT-Befehls muß den gleichen Datentyp und die gleiche Breite wie die entsprechende Spalte des anderen SELECT-Befehls besitzen.
- • Nur der letzte SELECT-Befehl kann eine ORDER BY-Klausel enthalten, die sich auf die Ausgabespalten über Nummern beziehen muß. Ist eine ORDER BY-Klausel angegeben, beeinflußt sie das gesamte Ergebnis.

Sie können die UNION-Klausel auch verwenden, um eine Inklusionsverknüpfung zu simulieren.

Wenn Sie zwei Tabellen in einer Abfrage miteinander verknüpfen, werden nur die Felder mit

übereinstimmenden Werten im Verknüpfungsfeld ausgegeben. Wenn es für einen Datensatz der Master-Tabelle keinen entsprechenden Datensatz in der Detailtabelle gibt, wird der Datensatz der Master-Tabelle nicht der Ausgabe hinzugefügt. Eine Inklusionsverknüpfung ermöglicht Ihnen, alle Datensätze der Master-Tabelle zusammen mit den übereinstimmenden Datensätzen der Detailtabelle in die Ausgabe zu übernehmen. Um eine Inklusionsverknüpfung in Visual FoxPro zu erstellen, benötigen Sie einen verschachtelten SELECT-Befehl wie im folgenden Beispiel:

```
SELECT customer.company, orders.order_id, orders.emp_id ;
   FROM customer, orders ;
   WHERE customer.cust_id = orders.cust_id ;
UNION ;
   SELECT customer.company, 0, 0 ;
   FROM customer ;
   WHERE customer.cust_id NOT IN ;
   (SELECT orders.cust_id FROM orders)
```

Anmerkung Achten Sie darauf, daß Sie das Leerzeichen direkt vor dem jeweiligen Semikolon (;) einfügen. Sonst kommt es zu einem Fehler.

Der Befehlsabschnitt vor der UNION-Klausel wählt Datensätze mit übereinstimmenden Werten aus beiden Tabellen aus. Die Kunden, für die keine Bestellungen (Rechnungen) vorhanden sind, werden nicht berücksichtigt. Der Befehlsabschnitt nach der UNION-Klausel wählt Datensätze in der Tabelle **customer** aus, die keine übereinstimmenden Datensätze in der Tabelle **orders** haben.

Bezüglich des zweiten Befehlsabschnitts ist folgendes anzumerken:

- Die SELECT-Anweisung in Klammern wird als erste ausgeführt. Diese Anweisung hat als Ergebnis eine Auswahl aller Kundennummern in der Tabelle **orders**.
- Die WHERE-Klausel sucht alle Kundennummern in der Tabelle **customer**, die sich nicht in der Tabelle **orders** befinden. Da der erste Befehlsabschnitt alle Kunden mit Kundennummern in der Tabelle **orders** zur Verfügung gestellt hat, sind nun alle in der Tabelle **customer** befindlichen Kunden in den Abfrageergebnissen enthalten.
- Da die Strukturen von Tabellen, die in einer UNION-Klausel angegeben sind, identisch sein müssen, gibt es in der zweiten SELECT-Anweisung zwei Platzhalter, um **orders.order_id** und **orders.emp_id** aus der ersten SELECT-Anweisung darzustellen.

  Anmerkung Die Platzhalter müssen dieselben Datentypen haben wie die Felder, denen sie entsprechen. Wenn das Feld vom Typ Datum ist, sollte der Platzhalter wie folgt aussehen: { / / } oder { . . }. Handelt es sich bei dem Feld um ein Zeichen-Feld, sollte der Platzhalter eine leere Zeichenfolge ("") sein.

ORDER BY *Sortierobjekt* [ASC | DESC] [, *Sortierobjekt* [ASC | DESC] ...] Sortiert die Abfrageergebnisse, basierend auf den Daten einer oder mehrerer Spalten. Jedes *Sortierobjekt* muß einer Spalte in den Abfrageergebnissen entsprechen und kann folgendes sein:

- Ein Feld einer FROM-Tabelle, das auch ein *Select-Element* in der SELECT-Hauptklausel (nicht in einer Unterabfrage) ist.
- Ein numerischer Ausdruck, der die Position der Spalte in der Ergebnistabelle angibt. (Die ganz linke Spalte ist die Spalte 1.)

ASC ist die Standardeinstellung für ORDER BY und gibt an, daß die Abfrageergebnisse entsprechend den Sortierobjekten aufsteigend sortiert werden.

DESC gibt eine absteigende Reihenfolge für Abfrageergebnisse an.

Wenn Sie keine Reihenfolge mit Hilfe von ORDER BY angeben, werden die Abfrageergebnisse unsortiert angezeigt.

Bemerkungen

SELECT ist ein SQL-Befehl, der wie jeder andere Visual FoxPro-Befehl Bestandteil von Visual FoxPro ist. Wenn Sie mit dem Befehl SELECT eine Abfrage formulieren, interpretiert Visual FoxPro die Abfrage und ruft die entsprechenden Daten aus den Tabellen ab. Eine SELECT-Abfrage können

Sie auf folgende Weise erstellen:

- Im Befehlsfenster.
- In einem Visual FoxPro-Programm (wie jeden anderen Visual FoxPro-Befehl).

Zu beachten ist, daß SELECT die aktuelle mit SET FILTER angegebene Filterbedingung nicht berücksichtigt.

Anmerkung Eine *Unterabfrage*, wie sie bei einigen der oben erläuterten Argumente angegeben ist, ist ein SELECT-Befehl innerhalb eines anderen SELECT-Befehls und muß in Klammern stehen. Sie können in einer WHERE-Klausel bis zu zwei Unterabfragen auf einer Ebene (nicht verschachtelt) erstellen (siehe im entsprechenden Abschnitt der Argumente). Unterabfragen können mehrere Verknüpfungsbedingungen enthalten.

Treiberhinweise

Wenn Ihre Anwendung einen in ODBC SQL formulierten SELECT-Befehl an eine Visual FoxPro-Datenquelle sendet, wandelt der Visual FoxPro-ODBC-Treiber den Befehl in einen Visual FoxPro-SELECT-Befehl um, ohne sonstige Änderungen vorzunehmen. Dies gilt nur dann nicht, wenn der Befehl ODBC-Escape-Sequenzen enthält. Elemente, die in einer Escape-Sequenz stehen, werden in die Visual FoxPro-Syntax umgewandelt. Weitere Informationen, wie ODBC-Escape-Sequenzen verwendet werden, finden Sie unter Zeit- und Datumsfunktionen sowie im Handbuch *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

Siehe auch

CREATE TABLE - SQL

INSERT - SQL

SET ANSI

SET EXACT

SET ANSI-Befehl

Siehe auch

Gibt an, wie Vergleiche zwischen Zeichenfolgen verschiedener Länge mit dem Operator "=" in Visual FoxPro-SQL-Befehlen durchgeführt werden.

Syntax

SET ANSI ON | OFF

Argumente

ON (Standardeinstellung für den Treiber; die Standardeinstellung für Visual FoxPro lautet OFF) Füllt die kürzere Zeichenfolge mit Leerzeichen auf, bis sie mit der r anderen Zeichenfolge übereinstimmt. Die beiden Zeichenfolgen werden dann Zeichen für Zeichen miteinander verglichen. Angenommen, Sie haben folgenden Vergleich:

```
'Tommy' = 'Tom'
```

Das Ergebnis lautet .F. (falsch), wenn SET ANSI auf ON eingestellt ist, da "Tom" zu "Tom " aufgefüllt wird und dann im direkten Vergleich nicht mit "Tommy" übereinstimmt.

Der Operator == verwendet diese Methode für Vergleiche in Visual FoxPro-SQL-Befehlen.

OFF Gibt an, daß die kürzere Zeichenfolge nicht mit Leerzeichen aufgefüllt wird. Die beiden Zeichenfolgen werden Zeichen für Zeichen bis zum Ende der kürzeren Zeichenfolge miteinander verglichen. Betrachten Sie erneut den folgenden Vergleich:

```
'Tommy' = 'Tom'
```

Das Ergebnis lautet .T. (wahr), wenn SET ANSI auf OFF eingestellt ist, da der Vergleich nach "Tom" endet.

Bemerkungen

SET ANSI gibt an, ob bei einem SQL-Zeichenfolgenvergleich die kürzere der beiden Zeichenfolgen mit Leerzeichen aufgefüllt wird. SET ANSI hat keinen Einfluß auf den Operator ==; bei Verwendung dieses Operators wird die kürzere Zeichenfolge vor einem Vergleich immer mit Leerzeichen aufgefüllt.

Reihenfolge der Zeichenfolgen

In SQL-Befehlen spielt es keine Rolle, welche der beiden Zeichenfolgen auf der linken oder rechten Seite steht. Das Umstellen einer Zeichenfolge von einer Seite des Operators = bzw. == auf die andere Seite wirkt sich nicht auf das Ergebnis des Vergleichs aus.

Siehe auch

SELECT - SQL

SET EXACT

SET BLOCKSIZE-Befehl

Gibt an, wie Visual FoxPro Speicherplatz zum Speichern von Memo-Feldern zuordnet.

Syntax

SET BLOCKSIZE TO *nBytes*

Argumente

nBytes Gibt die Blockgröße an, mit der Speicherplatz für Memo-Felder reserviert wird. Wenn *nBytes* gleich 0 ist, wird der Speicherplatz in einzelnen Bytes (1-Byte-Blöcken) reserviert. Ist *nBytes* eine ganze Zahl von 1 bis 32, wird der Speicherplatz in Blöcken reserviert, die *nBytes* multipliziert mit 512 groß sind. Ist *nBytes* größer als 32, wird der Speicherplatz in Blöcken reserviert, die *nBytes* groß sind. Wenn Sie als Blockgröße einen Wert größer als 32 angeben, können Sie viel Speicherplatz sparen.

Bemerkungen

Der Standardwert für SET BLOCKSIZE beträgt 64. Um die Blockgröße nach dem Erstellen der Datei auf einen anderen Wert einzustellen, setzen Sie die Blockgröße auf den gewünschten Wert, und erstellen Sie dann mit Hilfe des Befehls COPY eine neue Tabelle. Diese neue Tabelle hat dann die angegebene Blockgröße.

SET COLLATE-Befehl

Siehe auch

Gibt eine Sortierreihenfolge für Zeichen-Felder an. Diese Sortierreihenfolge wird dann in späteren Indizierungs- und Sortiervorgängen verwendet.

Syntax

SET COLLATE TO *zReihenfolge*

Argumente

zReihenfolge Gibt eine Sortierreihenfolge an, wobei die folgenden Optionen verfügbar sind:

Optionen	Sprache
DUTCH	Niederländisch
GENERAL	Englisch, Französisch, Deutsch, modernes Spanisch, Portugiesisch und andere westeuropäische Sprachen
GERMAN	Reihenfolge deutscher Telefonbücher (DIN)
ICELAND	Isländisch
MACHINE	Maschine (die standardmäßige Sortierreihenfolge früherer FoxPro-Versionen)
NORDAN	Norwegisch, Dänisch
SPANISH	Traditionelles Spanisch
SWEFIN	Schwedisch, Finnisch
UNIQWT	Unique Weight

Anmerkung Wenn Sie die Option SPANISH verwenden, wird "ch" als ein einziger Buchstabe betrachtet, der in der Sortierreihenfolge zwischen "c" und "d" liegt; "ll" liegt zwischen "l" und "m".

Wenn Sie eine Sortierreihenfolgenoption als Literalzeichenfolge angeben, müssen Sie diese in Anführungszeichen setzen:

```
SET COLLATE TO "SWEFIN"
```

MACHINE ist die Standardoption der Sortierreihenfolge und stellt die Sortierreihenfolge dar, mit der Xbase-Benutzer vertraut sind. Dabei sind die Zeichen entsprechend der aktuellen Codeseite sortiert.

GENERAL ist sicherlich für westeuropäische und US-Benutzer von Vorteil. Die Zeichen sind entsprechend der aktuellen Codeseite sortiert. In FoxPro, Version 2.5 oder früher, haben Sie beim Erstellen von Indizes möglicherweise UPPER( )- oder LOWER( )-Funktionen verwendet, um für Zeichen-Felder eine durchgängige Schreibweise zu erreichen. In FoxPro-Versionen höher als Version 2.5 können Sie statt dessen die Sortierreihenfolgenoption GENERAL angeben und auf die Konvertierung mit UPPER( ) verzichten.

Beachten Sie, wenn Sie eine andere Sortierreihenfolgenoption als MACHINE angeben und eine .IDX-Datei erstellen, daß immer eine komprimierte .IDX-Datei erstellt wird.

Mit SET("COLLATE") können Sie die aktuelle Sortierreihenfolge ausgeben.

Die für eine Datenquelle zu verwendende Sortierreihenfolge können Sie auf zwei Arten angeben: im Dialogfeld **ODBC Visual FoxPro-Setup** oder mit dem Schlüsselwort **Collate** in der Verbindungszeichenfolge (connection string), die Sie an **SQLDriverConnect** übergeben. Mit beiden Vorgehensweisen erreichen Sie dasselbe wie mit folgendem Befehl:

```
SET COLLATE TO zReihenfolge
```

Bemerkungen

SET COLLATE ermöglicht Ihnen das Sortieren von Tabellen, die Sonderzeichen einer der

unterstützten Sprachen enthalten. Das Ändern der Einstellung von SET COLLATE wirkt sich nicht auf die Sortierreihenfolge zuvor geöffneter Indizes aus. Visual FoxPro verwaltet automatisch bestehende Indizes und bietet somit die Flexibilität, mehrere verschiedene Indexarten, sogar für ein und dasselbe Feld, zu erstellen.

Wenn Sie beispielsweise einen Index mit dem Befehl SET COLLATE und der Option GENERAL erstellen und die SET COLLATE-Einstellung später in SPANISH ändern, behält der Index die Sortierreihenfolge GENERAL.

Siehe auch

Dialogfeld **ODBC Visual FoxPro-Setup**

SET DELETED-Befehl

Siehe auch

Gibt an, ob Datensätze, die zum Löschen markiert sind, verarbeitet werden und für andere Befehle zur Verfügung stehen.

Syntax

SET DELETED ON | OFF

Argumente

ON (Standardeinstellung für den Treiber; die Standardeinstellung für Visual FoxPro ist OFF) Gibt an, daß Befehle, die für die Verarbeitung von Datensätzen (einschließlich von Datensätzen eingebundener Tabellen) einen Geltungsbereich verwenden, alle Datensätze ignorieren, die zum Löschen markiert sind.

OFF Gibt an, daß Befehle, die für die Verarbeitung von Datensätzen (einschließlich von Datensätzen eingebundener Tabellen) einen Geltungsbereich verwenden, auf alle Datensätze zugreifen können, die zum Löschen markiert sind.

Bemerkungen

Abfragen, die den Befehl DELETED( ) verwenden, um den Status von Datensätzen zu testen, können mit der Rushmore-Technologie optimiert werden, wenn die Tabelle auf DELETED( ) indiziert ist.

Wichtig SET DELETED wird ignoriert, wenn der Geltungsbereich des Befehls standardmäßig gleich dem aktuellen Datensatz ist oder wenn Sie einen Geltungsbereich mit nur einem Datensatz angeben. INDEX ignoriert stets SET DELETED und indiziert alle Datensätze der jeweiligen Tabelle.

Siehe auch

DELETE - SQL

SET EXACT-Befehl

Siehe auch

Gibt die Regeln an, nach denen zwei Zeichenfolgen unterschiedlicher Länge verglichen werden.

Syntax

SET EXACT ON | OFF

Argumente

ON Gibt an, daß zwei Ausdrücke Zeichen für Zeichen übereinstimmen müssen, um gleich zu sein. Angehängte Leerzeichen in den Ausdrücken werden beim Vergleich ignoriert. Der kürzere der beiden Ausdrücke wird für den Vergleich auf der rechten Seite mit Leerzeichen aufgefüllt, damit er mit der Länge des anderen Ausdrucks übereinstimmt.

OFF (Standardeinstellung) Gibt an, daß in den Ausdrücken alle Zeichen bis zum Ende auf der rechten Seite übereinstimmen müssen, um als gleich bezeichnet zu werden.

Bemerkungen

Der Befehl SET EXACT hat keine Auswirkungen, wenn die zu vergleichenden Zeichenfolgen dieselbe Länge haben.

Zeichenfolgenvergleiche

Visual FoxPro besitzt zwei Vergleichsoperatoren, die auf Gleichheit prüfen.

Der Operator = führt einen Vergleich zwischen zwei Werten desselben Datentyps durch. Dieser Operator eignet sich zum Vergleichen von Daten der Typen Zeichen, Numerisch, Datum und Logisch.

Wenn Sie Zeichenausdrücke mit dem Operator = vergleichen, kann es jedoch vorkommen, daß das Ergebnis nicht dem entspricht, was Sie erwartet haben. Zeichenausdrücke werden zeichenweise von links nach rechts verglichen, bis entweder die Ausdrücke nicht mehr übereinstimmen oder das Ende des Ausdrucks auf der rechten Seite des Operators = erreicht ist (SET EXACT OFF) oder das Ende beider Ausdrücke erreicht ist (SET EXACT ON).

Der Operator == kann verwendet werden, wenn ein exakter Vergleich der Zeichenausdrücke erfolgen soll. Wenn zwei Zeichenausdrücke mit dem Operator == verglichen werden, müssen beide Ausdrücke (unabhängig davon, auf welcher Seite des Operators sie sich befinden) einschließlich Leerzeichen genau dieselben Zeichen enthalten, um als gleich zu gelten. Die Einstellung von SET EXACT wird ignoriert, wenn Zeichenfolgen mit dem Operator == verglichen werden.

Die folgende Tabelle zeigt, wie sich die Wahl des Operators und die SET EXACT-Einstellung auf einen Vergleich auswirken. (Ein Unterstrich stellt ein Leerzeichen dar.)

Vergleich	= EXACT OFF	= EXACT ON	== EXACT ON oder OFF
"abc" = "abc"	Übereinstimmung	Übereinstimmung	Übereinstimmung
"ab" = "abc"	Keine Übereinstimmung	Keine Übereinstimmung	Keine Übereinstimmung
"abc" = "ab"	Übereinstimmung	Keine Übereinstimmung	Keine Übereinstimmung
"abc" = "ab_"	Keine Übereinstimmung	Keine Übereinstimmung	Keine Übereinstimmung

"ab" = "ab_"	Keine Übereinstimmung	Übereinstimmung	Keine Übereinstimmung
"ab_" = "ab"	Übereinstimmung	Übereinstimmung	Keine Übereinstimmung
"" = "ab"	Keine Übereinstimmung	Keine Übereinstimmung	Keine Übereinstimmung
"ab" = ""	Übereinstimmung	Keine Übereinstimmung	Keine Übereinstimmung
"__" = ""	Übereinstimmung	Übereinstimmung	Keine Übereinstimmung
"" = "___"	Keine Übereinstimmung	Übereinstimmung	Keine Übereinstimmung
TRIM("___") = ""	Übereinstimmung	Übereinstimmung	Übereinstimmung
"" = TRIM("___")	Übereinstimmung	Übereinstimmung	Übereinstimmung

Siehe auch

SET ANSI

SET EXCLUSIVE-Befehl

Siehe auch

Gibt an, ob Tabellendateien in einem Netzwerk zur exklusiven oder zur gemeinsamen Benutzung geöffnet werden.

Syntax

SET EXCLUSIVE ON | OFF

Argumente

ON Beschränkt die Zugriffsmöglichkeit auf eine im Netzwerk geöffnete Tabelle auf den Benutzer, der die Tabelle geöffnet hat. Auf die Tabelle kann nicht von anderen Benutzern des Netzwerks zugegriffen werden. SET EXCLUSIVE ON verhindert auch den schreibgeschützten Zugriff für alle anderen Benutzer.

OFF (Standardeinstellung für den Treiber; die Standardeinstellungen für Visual FoxPro lauten ON für eine globale Datensitzung und OFF für eine private Datensitzung) Ermöglicht, daß eine Tabelle, die in einem Netzwerk geöffnet ist, von jedem Benutzer des Netzwerks verwendet und geändert werden kann.

Bemerkungen

Durch Ändern der SET EXCLUSIVE-Einstellung ändert sich nicht der Status zuvor geöffneter Tabellen. Wenn Sie beispielsweise eine Tabelle öffnen und dabei SET EXCLUSIVE auf ON setzen, aber diese Einstellung später in OFF ändern, behält die Tabelle ihren Status der exklusiven Benutzung.

Siehe auch

Dialogfeld **ODBC Visual FoxPro-Setup**

SET NULL-Befehl

Siehe auch

Legt fest, wie Nullwerte von den Befehlen ALTER TABLE - SQL, CREATE TABLE - SQL und INSERT - SQL unterstützt werden.

Syntax

SET NULL ON | OFF

Argumente

ON (Standardeinstellung für den Treiber; die Standardeinstellung für Visual FoxPro ist OFF) Gibt an, daß alle Spalten einer Tabelle, die mit ALTER TABLE oder CREATE TABLE erstellt wurde, Nullwerte enthalten können. Sie können die Unterstützung von Nullwerten für Spalten der Tabelle außer Kraft setzen, indem Sie in den Definitionen der Spalten die Klausel NOT NULL angeben.

ON gibt auch an, daß INSERT - SQL in alle Spalten, die nicht in der VALUE-Klausel von INSERT - SQL aufgeführt sind, Nullwerte einfügt. INSERT - SQL fügt nur in die Spalten Nullwerte ein, in denen Nullwerte zugelassen sind.

OFF Gibt an, daß keine der Spalten einer Tabelle, die mit ALTER TABLE oder CREATE TABLE erstellt wurde, Nullwerte enthalten kann. Sie können für die Spalten in ALTER TABLE und CREATE TABLE die Unterstützung von Nullwerten festlegen, indem Sie in den Definitionen der Spalten die NULL-Klausel angeben.

OFF gibt auch an, daß INSERT - SQL in alle Spalten, die nicht in der VALUE-Klausel von INSERT - SQL aufgeführt sind, leere Werte einfügt.

Bemerkungen

SET NULL hat nur Auswirkungen darauf, wie Nullwerte von den Befehlen ALTER TABLE, CREATE TABLE und INSERT - SQL unterstützt werden. Andere Befehle sind von SET NULL nicht betroffen.

Siehe auch

ALTER TABLE - SQL

CREATE TABLE - SQL

INSERT - SQL

SET PATH-Befehl

Siehe auch

Gibt einen Pfad für Dateisuchvorgänge an. Treiberspezifische Informationen finden Sie unter Treiberhinweise weiter unten.

Syntax

SET PATH TO [*Pfad*]

Argumente

TO [*Pfad*] Gibt die Verzeichnisse an, die Visual FoxPro durchsuchen soll. Trennen Sie die einzelnen Verzeichnisse durch Kommas oder Semikolons.

Bemerkungen

Mit SET PATH können Sie Suchpfade für andere Visual FoxPro-Programme angeben, die in gespeicherten Prozeduren aufgerufen werden können. SET PATH nimmt keine Änderungen am Pfad der Datenquelle vor, die Sie für die Verbindung angegeben haben.

Geben Sie SET PATH TO ohne *Pfad* aus, um den Pfad auf das Standardverzeichnis bzw. den Standardordner zurückzusetzen.

Treiberhinweise

Wenn Sie einen SET PATH-Befehl in einer gespeicherten Prozedur ausgeben, wird der Befehl von folgenden Funktionen und Befehlen ignoriert:

- Katalogfunktionen (catalog functions), wie z.B. **SQLTables** und **SQLColumns**, ignorieren den neuen Pfad und beziehen sich weiterhin auf den Pfad, der von der Datenquelle in **SQLPrepare** oder **SQLExecDirect** angegeben ist.
- Die Befehle SELECT, INSERT, UPDATE, DELETE und CREATE TABLE ignorieren den neuen Pfad und beziehen sich weiterhin auf den Pfad, der von der Datenquelle in **SQLPrepare** oder **SQLExecDirect** angegeben ist.

Wenn Sie SET PATH in einer gespeicherten Prozedur ausgeben und den Pfad später nicht wieder auf die alte Einstellung zurücksetzen, wird für weitere Verbindungen zu der Datenbank der neue Pfad verwendet (da SET PATH nicht auf Datensitzungen beschränkt ist).

Wenn Sie Tabellen erstellen, auswählen oder aktualisieren möchten, die sich in einem anderen Verzeichnis befinden als in der Datenquelle angegeben, sollten Sie in Ihrem Befehl den vollständigen Pfad der Dateien angeben.

Siehe auch

Dialogfeld **ODBC Visual FoxPro-Setup**

SQLColumns

SQLDriverConnect

SQLTables

SET REPROCESS-Befehl

Gibt an, wie oft oder wie lange nach einem erfolglosen Sperrversuch erneut versucht wird, eine Datei oder einen Datensatz zu sperren.

Syntax

SET REPROCESS TO *nVersuche* [SECONDS] | TO AUTOMATIC

Argumente

TO *nVersuche* [SECONDS] Gibt an, wie oft oder wie lange nach einem ersten erfolglosen Sperrversuch erneut versucht wird, eine Datei oder einen Datensatz zu sperren. Der Standardwert ist 0, der maximale Wert liegt bei 32.000.

SECONDS gibt an, daß Visual FoxPro versucht, *nVersuche* Sekunden lang eine Datei oder einen Datensatz zu sperren. SECONDS ist nur verfügbar, wenn *nVersuche* größer als 0 ist.

Ist *nVersuche* z.B. gleich 30, versucht Visual FoxPro bis zu 30 Mal, einen Datensatz oder eine Datei zu sperren. Wenn Sie zusätzlich SECONDS angeben (`SET REPROCESS TO 30 SECONDS`), wird der Versuch, einen Datensatz oder eine Datei zu sperren, bis zu 30 Sekunden lang ununterbrochen durchgeführt.

Ist eine ON ERROR-Routine aktiv und bleibt der Versuch eines Befehls, einen Datensatz oder eine Datei zu sperren, erfolglos, wird die ON ERROR-Routine ausgeführt. Wird der Sperrversuch jedoch von einer Funktion durchgeführt, wird die ON ERROR-Routine nicht ausgeführt, und die Funktion gibt .F. (falsch) zurück.

Wenn *nVersuche* gleich 0 (Standardwert) ist und Sie einen Befehl oder eine Funktion ausgeben, die einen Datensatz- oder Dateisperrversuch unternimmt, versucht Visual FoxPro, den Datensatz bzw. die Datei auf unbestimmte Zeit zu sperren. Wird der Datensatz bzw. die Datei zur Sperrung freigegeben, während Sie warten, wird die Sperrung durchgeführt und die Systemmeldung gelöscht. Führt eine Funktion den Sperrversuch aus, gibt sie den Wert .T. (wahr) zurück.

Wenn eine ON ERROR-Routine aktiv ist und ein Befehl versucht, den Datensatz oder die Datei zu sperren, hat die ON ERROR-Routine Vorrang vor weiteren Sperrversuchen. Die ON ERROR-Routine wird sofort ausgeführt. Visual FoxPro unternimmt keine weiteren Datensatz- oder Dateisperrversuche und zeigt nicht die Systemmeldung an.

Ist *nVersuche* gleich -1, versucht Visual FoxPro, die Datensatz- oder Dateisperrung auf unbestimmte Zeit fortzusetzen. Außerdem wird keine ON ERROR-Routine ausgeführt.

Ist der Datensatz bzw. die Datei, die Sie sperren möchten, von einem anderen Benutzer gesperrt, müssen Sie solange warten, bis dieser Benutzer die Sperrung aufgehoben hat.

TO AUTOMATIC Gibt an, daß Visual FoxPro versucht, den Datensatz oder die Datei auf unbestimmte Zeit zu sperren. (`SET REPROCESS TO -2` ist ein gleichwertiger Befehl.)

Bemerkungen

Eine Datensatz- oder Dateisperrung ist nicht immer beim ersten Versuch erfolgreich. Häufig wird ein Datensatz bzw. eine Datei von einem anderen Benutzer im Netzwerk gesperrt. SET REPROCESS gibt an, ob Visual FoxPro nach einem ersten erfolglosen Sperrversuch weitere Datensatz- bzw. Dateisperrversuche durchführen soll. Sie können entweder die Anzahl der weiteren Versuche oder die Dauer dieser Versuche angeben. Eine ON ERROR-Routine wirkt sich darauf aus, wie auf erfolglose Sperrversuche reagiert wird.

SET UNIQUE-Befehl

Siehe auch

Gibt an, ob Datensätze mit gleichem Indexschlüsselwert in einer Indexdatei verwaltet werden.

Syntax

SET UNIQUE ON | OFF

Argumente

ON Gibt an, daß zu der Indexdatei nicht mehrere Datensätze mit gleichem Indexschlüsselwert hinzugefügt werden. Lediglich der erste Datensatz mit diesem Indexschlüsselwert wird zu der Indexdatei hinzugefügt.

OFF (Standardeinstellung) Gibt an, daß zu der Indexdatei auch Datensätze mit gleichem Indexschlüsselwert hinzugefügt werden.

Bemerkungen

Bei der Ausführung von REINDEX behält die Indexdatei ihre SET UNIQUE-Einstellung. Weitere Informationen finden Sie unter INDEX.

Siehe auch

INDEX

UPDATE - SQL-Befehl

Siehe auch

Aktualisiert Datensätze einer Tabelle.

Der Visual FoxPro-ODBC-Treiber unterstützt die Visual FoxPro-spezifische Sprachsyntax dieses Befehls. Treiberspezifische Informationen finden Sie unter Treiberhinweise weiter unten.

Syntax

UPDATE [*Datenbankname1!*]*Tabellenname1*
SET *Spaltenname1 = aAusdruck1*
[, *Spaltenname2 = aAusdruck2* ...]
WHERE *Filterbedingung1* [AND | OR *Filterbedingung2* ...]]

Argumente

UPDATE [*Datenbankname1!*]*Tabellenname1* Gibt die Tabelle an, in der Datensätze aktualisiert werden sollen.

Datenbankname1! gibt den Namen einer Datenbank an, die die Tabelle enthält und nicht mit der Datenbank identisch ist, die für die Datenquelle angegeben ist. Wenn die Datenbank, die die Tabelle enthält, nicht die aktuelle Datenbank ist, müssen Sie den Namen der Datenbank angeben. Setzen Sie ein Ausrufezeichen (!) als Trennzeichen zwischen den Datenbank- und den Tabellennamen.

SET *Spaltenname1 = aAusdruck1*
[, *Spaltenname2 = aAusdruck2* Gibt die zu aktualisierenden Spalten sowie deren neue Werte an. Ist keine WHERE-Klausel angegeben, wird jeder Eintrag einer aufgeführten Spalte mit dem jeweils zugewiesenen Wert aktualisiert.

WHERE *Filterbedingung1* [AND | OR *Filterbedingung2* ...]] Gibt die Datensätze an, die aktualisiert werden sollen.

Filterbedingung gibt die Kriterien an, die die Datensätze erfüllen müssen, um aktualisiert zu werden. Sie können beliebig viele Filterbedingungen angeben und diese mit den Operatoren AND oder OR verbinden. Sie können auch den Operator NOT verwenden, um den Wert eines logischen Ausdrucks umzukehren, oder EMPTY( ), um auf ein leeres Feld zu prüfen.

Bemerkungen

UPDATE - SQL kann nur Datensätze einer einzigen Tabelle aktualisieren.

Im Gegensatz zu REPLACE verwendet UPDATE - SQL die Datensatzsperrung beim Aktualisieren mehrerer Datensätze in Tabellen, die zur gemeinsamen Nutzung geöffnet sind. Dadurch werden Datensatzkonflikte in Mehrbenutzerumgebungen verringert, aber u.U. auch das Leistungsverhalten. Um ein optimales Leistungsverhalten zu erzielen, öffnen Sie die Tabelle zur exklusiven Nutzung, oder verwenden Sie FLOCK( ) zum Sperren der Tabelle.

Treiberhinweise

Wenn Ihre Anwendung eine in ODBC SQL formulierte UPDATE-Anweisung an die Datenquelle sendet, wandelt der Visual FoxPro-ODBC-Treiber den Befehl in einen Visual FoxPro-UPDATE-Befehl um, ohne sonstige Änderungen vorzunehmen.

Siehe auch

DELETE - SQL

INSERT - SQL

Visual FoxPro-Felddatentypen

In der folgenden Tabelle sind die Werte aufgeführt, die in einem ALTER TABLE- oder CREATE TABLE-Befehl für das Argument *Feldtyp* möglich sind. Außerdem ist angegeben, ob die Argumente *nFeldbreite* und *nDezimalstellen* erforderlich sind.

Feldtyp	***nFeldbreite***	***nDezimalstellen***	**Erläuterung**
C	n	-	Zeichen-Feld (Character-Feld) der Breite *n*
D	-	-	Datum
T	-	-	DatumZeit (DateTime)
N	n	d	Numerisches Feld der Breite *n* mit *d* Dezimalstellen
F	n	d	Gleitkommazahl (Floating), numerisches Feld der Breite *n* mit *d* Dezimalstellen
I	-	-	Ganze Zahl (Integer)
B	-	d	Zahl (Double)
Y	-	-	Währung (Currency)
L	-	-	Logisch
M	-	-	Memo
G	-	-	Objekt (General)