

## **ODBC dBASE Driver**

### **Users**

The following topics discuss the ODBC dBASE driver and how to install it.

[Overview](#)

[Hardware and Software Requirements](#)

[Setting Up the ODBC dBASE Driver](#)

[Adding, Modifying, and Deleting a dBASE Data Source](#)

[Connecting to a dBASE Data Source](#)

[Using the ODBC dBASE Driver](#)

### **For Advanced Users**

The following topics discuss how to use the ODBC dBASE driver directly.

[Connection Strings \(Advanced\)](#)

[SQL Statements \(Advanced\)](#)

[Data Types \(Advanced\)](#)

[Error Messages \(Advanced\)](#)

### **For Programmers**

The following topics provide programming information on the ODBC dBASE driver. They are intended for application programmers and require knowledge of the Open Database Connectivity ([ODBC](#)) application programming interface ([API](#)).

[SQLGetInfo Return Values \(Programming\)](#)

[ODBC API Functions \(Programming\)](#)

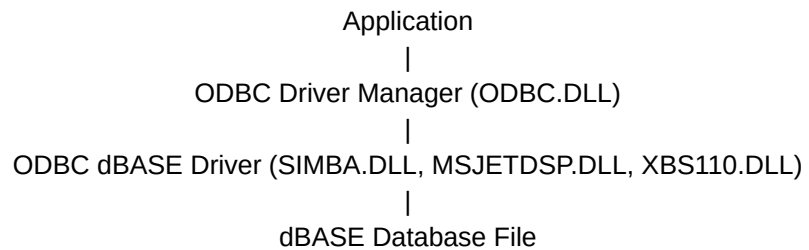
[Implementation Issues \(Programming\)](#)

## Overview

[See Also](#)

The ODBC dBASE driver allows you to open and query a dBASE database through the Open Database Connectivity ([ODBC](#)) interface.

The application/driver architecture is:



**See Also**

For All Users

[Adding, Modifying, and Deleting a dBASE Data Source](#)

[Connecting to a dBASE Data Source](#)

[Hardware and Software Requirements](#)

[Setting Up the ODBC dBASE Driver](#)

[Using the ODBC dBASE Driver](#)

## Hardware and Software Requirements

See Also

To access dBASE data, you must have:

- The ODBC dBASE driver
- dBASE III or dBASE IV database files
- A computer running MS-DOS 3.3 or later.
- Microsoft Windows 3.0a or later.
- The ODBC Driver Manager 1.0 (ODBC.DLL).

To add, modify, or delete drivers or data sources, you should have the ODBC Control Panel option (or the ODBC Administrator program if you're running Windows 3.0a) installed on your computer.

For more information about dBASE databases, see your dBASE documentation.

**See Also**

For All Users

[Setting Up the ODBC dBASE Driver](#)

## Setting Up the ODBC dBASE Driver

See Also

### To set up the ODBC dBASE driver

- 1 In the Main group in the Program Manager window, double-click the Control Panel icon. In the Control Panel window, double-click the ODBC icon.

---

**Note** For Microsoft Windows 3.0a, start the ODBC Administrator by double-clicking the ODBC Administrator icon in the Microsoft ODBC group.

---

- 2 In the Data Sources dialog box, choose the Drivers button.
- 3 In the Drivers dialog box, choose the Add button.
- 4 In the Add Driver dialog box, enter the name of the drive and directory containing the ODBC dBASE driver in the text box. Or choose the Browse button to select a drive and directory name.
- 5 Choose the OK button.
- 6 In the Install Drivers dialog box, choose dBASE from the Available ODBC Drivers list.
- 7 Choose the OK button to install the driver.

---

**Note** The ODBC dBASE driver may share some of the same dynamic link libraries (DLLs) with other drivers installed on your computer. If so, you will be asked to overwrite the ODBC dBASE driver, regardless of whether it has been installed. Choose the Yes button to install the driver.

---

Before using the driver, you must add a data source for it. In addition, you need to place the following line in your AUTOEXEC.BAT file:

```
SHARE /L:200
```

The SHARE command allows file sharing and locking in a multitasking environment. The /L option specifies the number of files that can be locked at one time.

### To delete the ODBC dBASE driver

- 1 In the Main group in the Program Manager window, double-click the Control Panel icon. In the Control Panel window, double-click the ODBC icon.

---

**Note** For Windows 3.0a, start the ODBC Administrator by double-clicking the Microsoft ODBC Administrator icon in the Microsoft ODBC group.

---

- 2 In the Data Sources dialog box, choose the Drivers button.
- 3 In the Drivers dialog box, select the ODBC dBASE driver from the Installed ODBC Drivers list.
- 4 Choose the Delete button.

The ODBC dBASE setup program asks if you want to remove the driver and all the data sources that use the driver.

- 5 Choose the Yes button.

**See Also**

For All Users

[Adding, Modifying, and Deleting a dBASE Data Source](#)

[Hardware and Software Requirements](#)

## Adding, Modifying, and Deleting a dBASE Data Source

See Also

Before you can access data with the ODBC dBASE driver, you must add a data source for it. The ODBC dBASE driver uses the information you enter to access the data. You can change or delete a data source at any time.

### To add a dBASE data source

- 1 In the Main group in the Program Manager window, double-click the Control Panel icon. In the Control Panel window, double-click the ODBC icon.

---

**Note** For Windows 3.0a, start the ODBC Administrator by double-clicking the Microsoft ODBC Administrator icon in the Microsoft ODBC group.

---

- 2 In the Data Sources dialog box, choose the Add button.
- 3 In the Add Data Source dialog box, select dBASE from the Installed ODBC Drivers list and choose OK.
- 4 In the ODBC dBASE Setup dialog box, enter information to set up the data source.

### To modify a dBASE data source

- 1 In the Main group in the Program Manager window, double-click the Control Panel icon. In the Control Panel window, double-click the ODBC icon.

---

**Note** For Windows 3.0a, start the ODBC Administrator by double-clicking the Microsoft ODBC Administrator icon in the Microsoft ODBC group.

---

- 2 In the Data Sources dialog box, select the data source from the Data Sources list.
- 3 Choose the Setup button.
- 4 In the ODBC dBASE Setup dialog box, enter information to set up the data source.

### To delete a dBASE data source

- 1 In the Main group in the Program Manager window, double-click the Control Panel icon. In the Control Panel window, double-click the ODBC icon.

---

**Note** For Windows 3.0a, start the ODBC Administrator by double-clicking the Microsoft ODBC Administrator icon in the Microsoft ODBC group.

---

- 2 In the Data Sources dialog box, select the data source from the Data Sources list.
- 3 Choose the Delete button, and then choose the Yes button to confirm the deletion.



**See Also**

For All Users

[Connecting to a dBASE Data Source](#)

[Setting Up the ODBC dBASE Driver](#)

**dBASE data source**

A dBASE data source specifies the default data directory in which the ODBC dBASE driver searches for dBASE files you want to access, as well as other database and driver information.

## ODBC dBASE Setup Dialog Box

[See Also](#)

The ODBC dBASE Setup dialog box contains the following fields:

### Data Source Name

A name that identifies the [data source](#), such as Payroll or Personnel.

### Description

An optional description of the data in the data source; for example, "Hire date, salary history, and current review of all employees."

### Version

Enables you to select the version of dBASE files used: dBASE III or dBASE IV.

### Directory

Displays the currently selected directory. Before you add the data source, you must either use the Select Directory button to select a directory, or select the Use Current Directory check box to use the application's current working directory.

When defining a dBASE data source directory, specify the directory where your most commonly used dBASE files are located. The ODBC dBASE driver uses this directory as the default directory. Copy other dBASE files into this directory if they are used frequently. Alternatively, you can qualify file names in a SELECT statement with the directory name:

```
SELECT * FROM C:\MYDIR\EMP
```

Or, you can use the USE statement to specify a new default directory:

```
USE C:\MYDIR
```

### Select Directory

Displays a dialog box where you can select a directory containing the dBASE files you want to access.

### Select Indexes

Displays the [Select Indexes dialog box](#) where you can associate dBASE files with index files.

### Use Current Directory

When selected, makes the application's current working directory the data source directory and disables the Select Directory option. When cleared, enables you to select the data source directory using the Select Directory option.

### Options

Displays the following options:

---

**Caution** Except for the Exclusive option, these options apply to all data sources that use the ODBC dBASE driver and ODBC Microsoft FoxPro driver.

---

### Collating Sequence

The sequence in which the fields are sorted: ASCII or International.

### Page Timeout

Specifies the period of time, in tenths of a second, that a page (if not used) remains in the buffer before being removed. The default is 600 tenths of a second (60 seconds).

### Exclusive

If the Exclusive box is selected, dBASE files will be opened in Exclusive mode and can be accessed by only one user at a time. If the Exclusive box is cleared, dBASE files will be opened in Shared mode and can be accessed by more than one user at a time. Performance is enhanced when running in Exclusive mode.

### Show Deleted Rows

Specifies whether or not rows that have been marked as deleted should be displayed.

**Approximate Row Count**

Determines whether table size statistics are approximated.

**See Also**

For All Users

[Adding, Modifying, and Deleting a dBASE Data Source](#)

[Select Indexes Dialog Box](#)

## Select Indexes Dialog Box

[See Also](#)

The Select Indexes dialog box contains the following fields:

### Tables

Displays a list of the files in the currently selected data source.

### Indexes

Displays the indexes assigned to the currently selected file in the Tables list.

### List Files of Type

Displays and allows you to choose the types of files to display in the Indexes list.

### To associate an index with a table

- 1 From the Tables list, select a file.
- 2 From the Indexes list, select an index.
- 3 Choose the OK button to save the table/index associations.

---

**Note** dBASE III indexes must be assigned using this dialog box for the driver to recognize them.

---

**See Also**

For All Users

[ODBC dBASE Setup Dialog Box](#)

## Connecting to a dBASE Data Source

See Also

When you connect to a dBASE data source, an application may prompt you to enter the name of a directory. If you are prompted, enter or select the directory containing the dBASE database files you want to access.



**See Also**

For All Users

[Adding, Modifying, and Deleting a dBASE Data Source](#)

[Using the ODBC dBASE Driver](#)

For Advanced Users

[Connection Strings \(Advanced\)](#)

## Using the ODBC dBASE Driver

The following information may be helpful when using the ODBC dBASE driver:

### Indexes

You must use the Select Indexes dialog box in the ODBC Control Panel option (or ODBC Administrator if you're running Windows 3.0a) to assign indexes to dBASE III files. If you do not do this, the driver cannot use or maintain the indexes.

### Using Reserved Words

Do not use reserved words listed in the SQL grammar in Appendix C of the *Microsoft ODBC Programmers Reference* as identifiers (that is, table or column names), unless you surround the word in double quotation marks (for example, "DATE").

### Column and Table Names

If column or table names contain any characters except letters, numbers, and underscores, they must be delimited. To delimit a column or table name, enclose the name in double quotes(""). Characters with an ASCII value greater than 127 are converted to underscores.

### .NTX Index Files

The ODBC dBASE driver does not support Clipper .NTX index files.

### Improving Response Time

Response time on large files can be improved by building an .MDX (or .NDX) index on the column (field) specified in the WHERE clause of a SELECT statement.

Existing .MDX indexes will automatically be applied for =, >, <, >=, <=, and BETWEEN operators in a WHERE clause, and LIKE predicates as well as in satisfying join predicates.

## Connection Strings (Advanced)

See Also

The connection string for the ODBC dBASE driver uses the following keywords:

<b>Keyword</b>	<b>Description</b>
<b>DSN</b>	Name of the dBASE data source.
<b>DBQ</b>	The dBASE database directory.
<b>FIL</b>	File type (DBASE3 or DBASE4).

For example, to connect to the Accounting data source in the directory C:\DBASE, use the following connection string:

```
DSN=Accounting;DBQ=C:\DBASE;FIL=DBASE4
```

**See Also**

For All Users

[Connecting to a dBASE Data Source](#)

## SQL Statements (Advanced)

[See Also](#)

The ODBC dBASE driver supports all [SQL statements](#) and clauses in the ODBC minimum grammar. For information about ODBC SQL grammar limitations, and additional and driver-specific grammar supported, see the following topics:

[Additional Supported ODBC SQL Grammar \(Advanced\)](#)

[Driver-specific ODBC SQL Grammar \(Advanced\)](#)

[Limitations to ODBC SQL Grammar \(Advanced\)](#)

**See Also**

For Advanced Users

[Data Types \(Advanced\)](#)

## Additional Supported ODBC SQL Grammar (Advanced)

See Also

The ODBC dBASE driver completely supports the following SQL statements and clauses in the Core and Extended ODBC grammar:

Core and Extended Grammar	Comments
Approximate numeric literal	Supported
AVG( <i>expression</i> ), COUNT(*), MAX( <i>expression</i> ), MIN( <i>expression</i> ), and SUM( <i>expression</i> )	Supported. See also the description of COUNT( <i>expression</i> ) in <u>Driver-specific ODBC SQL Grammar</u> .
BETWEEN predicate	Supported.
Correlation names are fully supported, including within the table list.	For example, in the following string, E1 is the correlation name for the table named Emp: <pre>SELECT * FROM Emp E1 WHERE E1.LastName = 'Smith'</pre>
DELETE, INSERT, and UPDATE support pathnames with table names	For example: <pre>INSERT INTO R:\MYDIR\EMP\MYTABLE</pre>
Exact numeric literal	Supported.
[HAVING <i>search-condition</i> ]	Supported.
IN ( <i>valuelist</i> )	Implemented as specified in the ODBC core grammar. For example: <pre>SELECT * FROM EMP WHERE Dept IN ('Sales','Marketing')</pre>

**See Also**

For Advanced Users

[Driver-specific ODBC SQL Grammar \(Advanced\)](#)

[Limitations to ODBC SQL Grammar \(Advanced\)](#)



## Driver-specific ODBC SQL Grammar (Advanced)

[See Also](#)

The ODBC dBASE driver supports the following driver-specific ODBC SQL grammar:

Driver-specific ODBC SQL Grammar	Comments
BETWEEN predicate	The syntax: <i>expression1</i> BETWEEN <i>expression2</i> AND <i>expression3</i> returns True only if <i>expression1</i> is greater than or equal to <i>expression2</i> and <i>expression1</i> is less than or equal to <i>expression3</i> .
COUNT( <i>expression</i> )	Counts all non-NULL values for an expression across a predicate. This function behaves like other set functions, such as SUM, AVG, MIN, and MAX. For example: <pre>SELECT COUNT (A+B) FROM Q</pre> counts all the rows in Q where A+B does not equal NULL.
Date arithmetic	The driver supports adding and subtracting an integer from a date column. The integer specifies the number of days to add or subtract. The driver also supports subtracting one date column from another to return a number of days.
Date literals	The YYYY-MM-DD format is supported.
GROUP BY <i>expression-list</i>	GROUP BY supports an expression list as well as a column name.
ORDER BY <i>expression-list</i>	If the expression is a single integer literal, it is interpreted as the number of the column in the result set. Ordering is done on one of the result table columns. No ordering is allowed on Set functions or an expression that contains a Set function.  For example, in the following clauses the table is ordered by three key expressions: a+b, c+d, and e. <pre>SELECT * FROM emp ORDER BY a+b, c+d, e</pre>
ORDER BY with GROUP BY	ORDER BY can be performed on any expression in the GROUP BY <i>expression-list</i> or

### Outer Joins

any column in the result set.

A SELECT statement can contain a list of OUTER JOIN clauses.

### Scalar Functions

Supported.

Table names that occur in the FROM clause of SELECT, after the INTO clause in INSERT, and after CREATE and DROP TABLE can contain a valid pathname, primary name, and filename extension.

Use of a table name elsewhere in a SQL statement does not support the use of pathnames or extensions but will accept only the primary name (for example, EMP FROM C:\ABC\EMP).

If a filename extension is not specified, .DBF is assumed.

Correlation names (aliases) can be used. For example:

```
SELECT * FROM  
C:\ABC\EMP T1  
WHERE T1.COL1 = 'aaa'
```

### USE [drive:]\dir

Sets the current database directory. *drive* is a valid drive name and *dir* is any valid MS-DOS directory name.

For example, the following changes the current directory to C:\DBDIR:

```
USE C:\DBDIR
```

**See Also**

For Advanced Users

[Additional Supported ODBC SQL Grammar \(Advanced\)](#)

[Limitations to ODBC SQL Grammar \(Advanced\)](#)

## Outer Joins (Advanced)

See Also

The ODBC dBASE driver extends the OUTER JOIN syntax to support nested outer joins. The OUTER JOIN syntax is:

*left-outer-join* ::=

*table-reference* **LEFT OUTER JOIN** *table-reference*  
**ON** *search-condition*

*table-reference* ::=

*table-name* | [(*left-outer-join*)]

where *table-name* can be a table name or a table name followed by a correlation name. For example, the following statement uses a three-way outer join to create a list of sales orders. For each sales order, all line numbers (if any) are listed, and for each line number, the part and description (if any) are listed.

```
SELECT Order.SONum,  
       Line.LineNum,  
       Part.PartNum,  
       Part.Description  
FROM Order LEFT OUTER JOIN  
  (Line LEFT OUTER JOIN Part  
   ON Line.PartNum=Part.PartNum)  
  ON Order.SONum=Line.SONum
```

---

**Note** The rightmost ON corresponds to the leftmost LEFT OUTER JOIN.

---

**See Also**

For Advanced Users

[Additional Supported ODBC SQL Grammar \(Advanced\)](#)

[Limitations to ODBC SQL Grammar \(Advanced\)](#)

## Limitations to ODBC SQL Grammar (Advanced)

See Also

The ODBC dBASE driver imposes the following limitations on the ODBC SQL grammar:

<b>Grammar</b>	<b>Limitation</b>
Character string literals	Any ANSI character (1 - 255 decimal). One single quotation mark (') is represented using two consecutive single quotation marks (").
Columns	Maximum number is 255 when a table is created.
Comparison predicate	For SQL_BIT data, comparisons can be made for equality and inequality only (= and <> operators).
LIKE Predicate	If data in a column is longer than 255 characters, the LIKE comparison will be based only on the first 255 characters.
Literals	Maximum length is 1000 characters.
NOT NULL	Not supported in CREATE TABLE statement.
Predicates	Maximum number is 300.
Set Functions	AVG, MAX, MIN, and SUM do not support the DISTINCT keyword.
Sort Keys	The maximum length of a sort key in a GROUP BY clause, ORDER BY clause, SELECT DISTINCT statement, or outer join is 255 bytes; the maximum length of all sort keys in a sort row is 65,500 bytes.  If the length of the data in a column is greater than 255 characters, sorting will be based on the

first 255 characters.

**See Also**

For Advanced Users

[Additional Supported ODBC SQL Grammar \(Advanced\)](#)

[Driver-specific ODBC SQL Grammar \(Advanced\)](#)



## **dBASE Indexes (Advanced)**

The ODBC dBASE driver automatically opens and updates dBASE IV index files. You must use the ODBC Control Panel option (or ODBC Administrator program if you're running Windows 3.0a) to associate dBASE III .NDX files with dBASE files.

The following limitations apply to the creation of dBASE indexes:

- All column names must be valid.
- All columns must be in the same ascending or descending order.
- If there is a single text column, its length must be less than 100 bytes.
- If there is more than one column, all of them must be text columns and the sum of the column sizes must be less than 100 bytes.
- Logical or memo fields cannot be indexed.
- An index must not be specified for the current set of fields (that is, duplicate indexes are not allowed).
- The index name must match the dBASE index naming convention. dBASE III requires that each index be in a separate file, each having an .NDX extension. In dBASE IV, indexes are created as tag names that are stored in a single .MDX file. The .MDX file has the same base name as the database file (for example, EMP.MDX is the index file for the EMP.DBF database).

## Data Types (Advanced)

See Also

The following table shows how dBASE data types are mapped to ODBC SQL data types. Note that not all ODBC SQL data types are supported.

<b>dBASE Data Type</b>	<b>ODBC Data Type</b>
Character	SQL_CHAR
Date	SQL_DATE
Logical	SQL_BIT
Memo	SQL_LONGVARCHAR
Numeric (BCD)	SQL_DOUBLE

Precision in dBASE III allows numbers with up to two-digit exponents and in dBASE IV numbers with up to three-digit exponents. Since numbers are stored as text, they are converted to numbers. If the number to convert does not fit in a field, unexplained results may occur.

While dBASE allows a precision and a scale to be specified with a Numeric data type, it is not supported by the ODBC dBASE driver. The ODBC dBASE driver always returns a precision of 15 and a scale of 0 for Numeric data type.

A column created with the Numeric data type using the ODBC dBASE driver maps to the SQL\_DOUBLE ODBC data type. Thus the data in this column is subject to rounding. This behavior is not the same as that of the Numeric data type in dBASE (type N), which is Binary Coded Decimal (BCD).

---

**Note** **SQLGetTypeInfo** returns ODBC SQL data types. All conversions in Appendix D of the *Microsoft ODBC SDK Programmer's Reference* are supported for the ODBC SQL data types listed earlier in this topic.

---

### Limitations

The ODBC dBASE driver and dBASE impose the following limitations on the data types:

- The maximum length of a LONGVARCHAR column is 65,500 bytes.
- When converting dBASE data to the C data type SQL\_C\_TINYINT, numbers from 0 to 127 are converted correctly. Numbers from 128 to 255 are converted to numbers from -128 to -1. Numbers less than 0 or greater than 255 cannot be converted.

When converting data from the C data type SQL\_C\_TINYINT to dBASE data, numbers from 0 to 127 are converted correctly. Numbers from -128 to -1 are converted to numbers from 128 to 255.

This occurs because SQL\_C\_TINYINT is signed, but the ODBC dBASE driver uses unsigned single-byte integers.

**See Also**

For Advanced Users

[SQL Statements \(Advanced\)](#)

## Error Messages (Advanced)

When an error occurs, the ODBC dBASE driver returns the native error number, the SQLSTATE (an ODBC error code), and an error message.

### Native Error

For errors that occur in the ISAM layer, the ODBC dBASE driver returns the native error returned to it by the ODBC File Library (that is, the dBASE ISAM). For errors that are detected by the Simba driver, the ODBC dBASE driver returns a native error of zero.

### SQLSTATE

For errors that occur in the data source, the ODBC dBASE driver maps the returned native error to the appropriate SQLSTATE. For errors that are detected by the driver or the Driver Manager, the ODBC dBASE driver or Driver Manager generates the appropriate SQLSTATE.

### Error Message

For errors that occur in the data source, the ODBC dBASE driver returns an error message returned to it by the ODBC File Library. For errors that occur in the ODBC dBASE driver or the Driver Manager, the ODBC dBASE driver returns an error message based on the text associated with the SQLSTATE.

Error messages have the following format:

*[vendor][ODBC-component][data-source]message-text*

where the prefixes in brackets ([ ]) identify the location of the error. When the error occurs in the Driver Manager or Simba driver, *data-source* is not given. When the error occurs in the data source, the *[vendor]* and *[ODBC-component]* prefixes identify the vendor and name of the ODBC component that received the error from the data source.

The following table shows the error messages returned by the Driver Manager, Simba driver and dBASE ISAM:

Error Message	Error location
<i>[Microsoft][ODBC DLL]message-text</i>	Driver Manager (ODBC.DLL)
<i>[Microsoft][ODBC Single-Tier Driver]message-text</i>	Simba Driver (SIMBA.DLL)
<i>[Microsoft][ODBC Single-Tier Driver][ODBC File Library]message-text</i>	dBASE ISAM (XBS110.DLL)

## SQLGetInfo Return Values (Programming)

[See Also](#)

The following table lists the C language #defines for the *flInfoType* argument and the corresponding values returned by **SQLGetInfo**. This information can be retrieved by passing the listed C language #defines to **SQLGetInfo** in the *flInfoType* argument. Where **SQLGetInfo** returns a 32-bit bitmask, a vertical bar (|) represents a bitwise OR. For more information about the values return by **SQLGetInfo**, see the *Microsoft ODBC SDK Programmer's Reference, Version 1.0*.

<i>flInfoType</i> Value (#define)	Returned value
SQL_ACCESSIBLE_PROCEDURES	"N"
SQL_ACCESSIBLE_TABLES	"N"
SQL_ACTIVE_CONNECTIONS	0
SQL_ACTIVE_STATEMENTS	0
SQL_CONCAT_NULL_BEHAVIOR	1
SQL_CONVERT_BIGINT	0
SQL_CONVERT_BINARY	0
SQL_CONVERT_BIT	SQL_CVT_BIT   SQL_CVT_CHAR   SQL_CVT_DOUBLE   SQL_CVT_LONGVARCHAR
SQL_CONVERT_CHAR	SQL_CVT_BIT   SQL_CVT_CHAR   SQL_CVT_DATE   SQL_CVT_DOUBLE   SQL_CVT_LONGVARCHAR
SQL_CONVERT_DATE	SQL_CVT_CHAR   SQL_CVT_DATE
SQL_CONVERT_DECIMAL	0
SQL_CONVERT_DOUBLE	SQL_CVT_BIT   SQL_CVT_CHAR   SQL_CVT_DOUBLE   SQL_CVT_LONGVARCHAR
SQL_CONVERT_FLOAT	SQL_CVT_BIT   SQL_CVT_CHAR   SQL_CVT_DOUBLE   SQL_CVT_LONGVARCHAR
SQL_CONVERT_FUNCTIONS	SQL_FN_CVT_CONVERT
SQL_CONVERT_INTEGER	SQL_CVT_BIT   SQL_CVT_CHAR   SQL_CVT_DOUBLE   SQL_CVT_LONGVARCHAR
SQL_CONVERT_LONGVARBINARY	0
SQL_CONVERT_LONGVARCHAR	SQL_CVT_CHAR   SQL_CVT_LONGVARCHAR
SQL_CONVERT_NUMERIC	SQL_CVT_BIT   SQL_CVT_CHAR   SQL_CVT_DOUBLE   SQL_CVT_LONGVARCHAR
SQL_CONVERT_REAL	SQL_CVT_BIT   SQL_CVT_CHAR   SQL_CVT_DOUBLE   SQL_CVT_LONGVARCHAR
SQL_CONVERT_SMALLINT	SQL_CVT_BIT   SQL_CVT_CHAR

	SQL_CVT_DOUBLE
	SQL_CVT_LONGVARCHAR
SQL_CONVERT_TIME	SQL_CVT_CHAR
SQL_CONVERT_TIMESTAMP	SQL_CVT_CHAR
	SQL_CVT_DATE
SQL_CONVERT_TINYINT	SQL_CVT_BIT
	SQL_CVT_CHAR
	SQL_CVT_DOUBLE
	SQL_CVT_LONGVARCHAR
SQL_CONVERT_VARBINARY	0
SQL_CONVERT_VARCHAR	0
SQL_CORRELATION_NAME	2
SQL_CURSOR_COMMIT_BEHAVIOR	2
SQL_CURSOR_ROLLBACK_BEHAVIOR	0
SQL_DATA_SOURCE_READ_ONLY	"N" (The driver does not check to see whether the disk drive is read-only.)
SQL_DBMS_NAME	"DBASE3" or "DBASE4"
SQL_DBMS_VER	"3.0" or "4.0"
SQL_DEFAULT_TXN_ISOLATION	0
SQL_DRIVER_NAME	"SIMBA.DLL"
SQL_DRIVER_VER	" 1.01.nnnn" (nnnn specifies the build date.)
SQL_EXPRESSIONS_IN_ORDERBY	"Y"
SQL_FETCH_DIRECTION	SQL_FD_FETCH_NEXT
SQL_IDENTIFIER_CASE	4
SQL_IDENTIFIER_QUOTE_CHAR	"" (double quotation mark).
SQL_MAX_COLUMN_NAME_LEN	10
SQL_MAX_CURSOR_NAME_LEN	18
SQL_MAX_OWNER_NAME_LEN	0
SQL_MAX_PROCEDURE_NAME_LEN	0
SQL_MAX_QUALIFIER_NAME_LEN	66
SQL_MAX_TABLE_NAME_LEN	12
SQL_MULT_RESULT_SETS	"N"
SQL_MULTIPLE_ACTIVE_TXN	"N"
SQL_NON_NULLABLE_COLUMNS	0
SQL_NUMERIC_FUNCTIONS	SQL_FN_NUM_MOD
SQL_ODBC_API_CONFORMANCE	1
SQL_ODBC_SAG_CLI_CONFORMANCE	1
SQL_ODBC_SQL_CONFORMANCE	0
SQL_ODBC_SQL_OPT_IEF	"N"
SQL_OUTER_JOINS	"Y"
SQL_OWNER_TERM	""
SQL_PROCEDURE_TERM	""
SQL_PROCEDURES	"N"
SQL_QUALIFIER_NAME_SEPARATOR	"\" (backslash)
SQL_QUALIFIER_TERM	"DIRECTORY"
SQL_ROW_UPDATES	"Y"
SQL_SCROLL_CONCURRENCY	SQL_SCCO_READ_ONLY

SQL_SCROLL_OPTIONS	SQL_SO_FORWARD_ONLY
SQL_SEARCH_PATTERN_ESCAPE	"\" (backslash)
SQL_SERVER_NAME	"DBASE3" or "DBASE4"
SQL_STRING_FUNCTIONS	SQL_FN_STR_CONCAT
	SQL_FN_STR_LCASE
	SQL_FN_STR_LEFT
	SQL_FN_STR_LENGTH
	SQL_FN_STR_LOCATE
	SQL_FN_STR_LTRIM
	SQL_FN_STR_RIGHT
	SQL_FN_STR_RTRIM
	SQL_FN_STR_SUBSTRING
	SQL_FN_STR_UCASE
SQL_SYSTEM_FUNCTIONS	SQL_FN_SYS_DBNAME
	SQL_FN_SYS_USERNAME
SQL_TABLE_TERM	"TABLE"
SQL_TIMEDATE_FUNCTIONS	SQL_FN_TD_CURDATE
	SQL_FN_TD_CURTIME
	SQL_FN_TD_DAYOFMONTH
	SQL_FN_TD_DAYOFWEEK
	SQL_FN_TD_MONTH
	SQL_FN_TD_YEAR
SQL_TXN_CAPABLE	0
SQL_TXN_ISOLATION_OPTIONS	0

**See Also**

For Advanced Users

[SQL Statements \(Advanced\)](#)

[Data Types \(Advanced\)](#)

For Programmers

[Scalar Functions \(Programming\)](#)



## Scalar Functions (Programming)

The ODBC dBASE driver supports the following scalar functions:

CONCAT	LCASE	RIGHT
CONVERT	LEFT	RTRIM
CURDATE	LENGTH	SUBSTRING
CURTIME	LOCATE	UCASE
DATABASE	LTRIM	USER
DAYOFMONTH	MOD	YEAR
DAYOFWEEK	MONTH	

For information about the arguments and return values of scalar functions, see Appendix G of the *Microsoft ODBC SDK Programmer's Reference*.

## ODBC API Functions (Programming)

[See Also](#)

The ODBC dBASE driver supports all Core and Level 1 functions and the following Level 2 functions:

- SQLDataSources
- SQLMoreResults

These ODBC API functions have the following implementations with the ODBC dBASE driver:

Function	Description
<b>SQLDriverConnect</b>	The following keywords are supported in the <u>connection string</u> : <b>DSN</b> , <b>DBQ</b> , and <b>FIL</b> .
<b>SQLGetConnectOption</b> and <b>SQLSetConnectOption</b>	These functions support the SQL_ACCESS_MODE, SQL_CURRENT_QUALIFIER, SQL_OPT_TRACE, and SQL_OPT_TRACEFILE connection options. <b>SQLGetConnectOption</b> also supports the SQL_AUTOCOMMIT option.
<b>SQLGetData</b>	This function can retrieve data from any column, whether or not there are bound columns after it and regardless of the order in which the columns are retrieved.
<b>SQLGetInfo</b>	<b>SQLGetInfo</b> supports a driver-specific information type, SQL_FILE_USAGE (65002). The returned value is a 16-bit integer that indicates how the driver directly treats files in a data source: 0 (SQL_FILE_NOT_SUPPORTED) = The driver is not a single-tier driver. 1 (SQL_FILE_TABLE) = A single-tier driver treats files in a data source as tables. 3 (SQL_FILE_QUALIFIER) = A single-tier driver treats files in a data source as a qualifier. The ODBC dBASE driver returns 1, since each dBASE file is a table.
<b>SQLGetStmtOption</b> and <b>SQLSetStmtOption</b>	These functions support the SQL_MAX_LENGTH, SQL_MAX_ROWS, and SQL_NOSCAN statement options. They also support a driver-specific statement option, <u>SQL_LOCK_TABLES</u> .
<b>SQLGetTypeInfo</b>	Only data type names returned by <b>SQLGetTypeInfo</b> can be used with CREATE statements.
<b>SQLMoreResults</b>	<b>SQLMoreResults</b> always returns SQL_NO_DATA_FOUND. It cannot return additional results.
<b>SQLSetCursorName</b> and	These functions are supported, but cannot be used for positioned

<b>SQLGetCursorName</b>	updates or deletes (for example, WHERE CURRENT OF <i>cursorname</i> ).
<b>SQLTables</b>	The table names returned by <b>SQLTables</b> have no filename extensions.
<b>SQLTransact</b>	The ODBC dBASE driver does not support transactions; all SQL statements that modify data are always committed. Therefore, <b>SQLTransact</b> always returns SQL_SUCCESS when <i>fType</i> is SQL_COMMIT and always returns SQL_ERROR when <i>fType</i> is SQL_ROLLBACK.

### See Also

For Advanced Users

[Error Messages \(Advanced\)](#)

For Programmers

[Implementation Issues \(Programming\)](#)

## SQL\_LOCK\_TABLES Statement Option (Programming)

The ODBC dBASE driver supports a driver-specific statement option, `SQL_LOCK_TABLES`, that supports the values `DEFLOCK` and `XLOCK`. When the value of the option is `DEFLOCK`, tables used by the *hstmt* are subject only to the default locking mechanisms of the ODBC dBASE driver. This is the default setting.

When the value of the option is `XLOCK`, all tables used by the *hstmt* are exclusively locked when a **SELECT**, **UPDATE**, **INSERT**, **DELETE**, or **CREATE INDEX** statement is executed on the *hstmt*. The tables remain locked until the *hstmt* is dropped (by calling **SQLFreeStmt** with the `SQL_DROP` option) or the option is set to `DEFLOCK` and the *hstmt* is reexecuted.

Locked tables can only be used by the locking *hstmt*; they cannot be used by any other *hstmts*. For example, the last function call in the following code generates an access violation because it uses a different *hstmt*:

```
SQLSetStmtOption(hstmt1, SQL_LOCK_TABLES, XLOCK);
SQLExecDirect(hstmt1, "SELECT * FROM EMP", SQL_NTS);
SQLExecDirect(hstmt1, "UPDATE EMP SET DEPT=12", SQL_NTS);
/* This call generates an access violation */
SQLExecDirect(hstmt2, "SELECT * FROM EMP", SQL_NTS);
```

Furthermore, **SQLColumns**, **SQLSpecialColumns**, and **SQLStatistics** cannot retrieve information about a table locked on another *hstmt*.

---

**Caution** Be careful to avoid locking more files than necessary. If the same *hstmt* is used to execute statements for many different tables, all the tables will remain locked until the *hstmt* is dropped or locking is turned off.

---

The `SQL_LOCK_TABLES` statement option uses the following `#defines`:

```
#define SQL_LOCK_TABLES 1153
#define DEFLOCK 0
#define XLOCK 1
```

The `SQL_LOCK_TABLES` statement option cannot be used with **SQLSetConnectOption**.

## Implementation Issues (Programming)

The following implementation-specific issues might affect the use of the ODBC dBASE driver.

### Arithmetic Errors

The ODBC dBASE driver evaluates the WHERE clause in a SELECT statement as it fetches each row. If a row contains a value that causes an arithmetic error, such as divide-by-zero or numeric overflow, the driver returns all rows, but returns errors for columns with arithmetic errors. When inserting or updating, however, the ODBC dBASE driver stops inserting or updating data when the first arithmetic error is encountered.

### Closing Open Tables (Files)

Calling **SQLFreeStmt** with the SQL\_CLOSE option changes the statement state but does not close the tables used by the *hstmt*. To close the tables currently used by *hstmt*, you must call **SQLFreeStmt** with the SQL\_DROP option.

In the following example, when **SQLFreeStmt** is called, the emp and dept tables remain open:

```
SQLPrepare(hStmt, "SELECT * FROM emp, dept  
WHERE emp.dept = dept.dept_id", SQL_NTS);  
SQLExecute(hStmt);  
/* .SQLFetch until NO_DATA_FOUND  
SQLFreeStmt(hStmt, SQL_CLOSE);  
SQLPrepare(hStmt, "SELECT * FROM emp", SQL_NTS);
```

---

**Note** Each file used by the ODBC dBASE driver requires a file handle. Because tables (files) remain open until **SQLFreeStmt** is called with the SQL\_DROP option, reusing an *hstmt* for different tables without dropping it can result in an error caused by attempting to open too many files. Also note that dBASE indexes and Memo fields require separate files, so opening a single table can use several file handles.

---

### Single-user Mode vs. Multi-user Mode

The ODBC dBASE driver can use a dBASE data source in single- or multi-user mode. The user specifies which mode to use in the [ODBC dBASE Setup dialog box](#).

The ODBC dBASE driver allows only one user at a time to be connected to a single-user data source. When it uses a file in the data source, such as when it processes a SELECT or UPDATE statement, it exclusively locks the file.

The ODBC dBASE driver allows any number of users to be connected to a multi-user data source at the same time. It exclusively locks records before updating or deleting them. Other users cannot update or delete a locked record. Whether they can read the record depends on the version of the file. Other users cannot read locked dBASE III records; they can read locked dBASE IV records ("dirty reads" are permitted).

---

**Note** Although the ODBC dBASE driver locks records in multi-user mode, this does not imply that serializable transactions are available. The ODBC dBASE driver does not support transactions, nor does it support any transaction isolation levels.

---

### Sorting with DISTINCT, GROUP BY, JOIN, or ORDER BY

DISTINCT, GROUP BY, JOIN, and ORDER BY always result in a sort. If indexes are found, data is dynamically fetched and the sort is based using those indexes. If indexes are not found, a temporary table called a snapshot is taken of the data and the sort occurs on the temporary table. This type of sort is not based on dynamic data since the temporary table is built from data found in the original data file at **SQLExecute** time.

**API**

Application programming interface. A set of routines that an application, such as Microsoft Access, uses to request and carry out lower-level services.

**character set**

A character set is a set of 256 letters, numbers, and symbols specific to a country or language. Each character set is defined by a table called a code page. An OEM (Original Equipment Manufacturer) character set is any character set except the ANSI character set. The ANSI character set (code page 1007) is the character set used by Microsoft Windows.

**conformance level**

Some applications can use only drivers that support certain levels of functionality, or conformance levels. For example, an application might require that drivers be able to prompt the user for the password for a data source. This ability is part of the Level 1 conformance level for the application programming interface (API).

Every ODBC driver conforms to one of three API levels (Core, Level 1, or Level 2) and one of three SQL grammar levels (Minimum, Core, or Extended). Drivers may support some of the functionality in levels above their stated level.

For detailed information about conformance levels, programmers should see the *Microsoft ODBC SDK Programmer's Reference*.



**data source**

A data source includes the data a user wants to access and the information needed to get to that data.

Examples of data sources are:

- A SQL Server database, the server on which it resides, and the network used to access that server.
- A directory containing a set of dBASE files you want to access.

**DBMS**

Database management system. The software used to organize, analyze, search for, update, and retrieve data.

**DDL**

Data definition language. Any SQL statement that can be used to define data objects and their attributes. Examples include CREATE TABLE, DROP VIEW, and GRANT statements.

**DLL**

Dynamic-link library. A set of routines that one or more applications can use to perform common tasks. The ODBC drivers are DLLs.

**DML**

Data manipulation language. Any SQL statement that can be used to manipulate data. Examples include UPDATE, INSERT, and DELETE statements.

**ODBC**

Open Database Connectivity. A Driver Manager and a set of ODBC drivers that enable applications to access data using SQL as a standard language.

**ODBC Driver Manager**

A dynamic-link library (DLL) that provides access to ODBC drivers.

**ODBC driver**

A dynamic-link library (DLL) that an ODBC-enabled application, such as Microsoft Excel, can use to gain access to a particular data source. Each database management system (DBMS), such as Microsoft SQL Server, requires a different driver.



**SQL**

Structured Query Language. A language used for retrieving, updating, and managing data.

**SQL statement**

A command written in Structured Query Language (SQL); also known as a query. An SQL statement specifies an operation to perform, such as SELECT, DELETE, or CREATE TABLE; the tables and columns on which to perform that operation; and any constraints to that operation.

**translation option**

An option that specifies how a translator translates data. For example, a translation option might specify the character sets between which a translator translates character data. It might also provide a key for encryption and decryption.

**translator**

A dynamic-link library (DLL) that translates all data passing between an application, such as Microsoft Access, and a data source. The most common use of a translator is to translate character data between different character sets. A translator can also perform tasks such as encryption and decryption or compression and expansion.

