SQL for Flat-File Drivers

To enable your application to be portable across other databases, you can use SQL statements with the flat-file drivers (Btrieve, dBASE, Excel 4 and 5, Paradox 4, Paradox 5, and Text).   The database drivers parse SQL statements and translate them into a form that the database can understand.

The following SQL statements let you read, insert, update, and delete records from a database, create new tables, and drop existing tables:

Select Statement

Create Table Statement

Drop Table Statement

Insert Statement

Update Statement

Delete Statement

The following topics describe specific elements of the SQL statements supported:

Aggregate Functions

SQL Expressions

Reserved Keywords

Select Statement

The form of the Select statement supported by the flat-file drivers is

SELECT [DISTINCT] {* | *column_expression, ...*}

FROM *table_names* [*table_alias*] ...

[WHERE *expr1 rel_operator expr2*]

[GROUP BY {*column_expression, ...*}]
[HAVING *expr1 rel_operator expr2*]

[UNION [ALL] (SELECT...)]

[ORDER BY {*sort_expression* [DESC | ASC]}, ...]

[FOR UPDATE [OF {*column_expression, ...*}]]

Click any keyword to see a description of its statement clause.

Select Clause

Follow Select with a list of column expressions you want to retrieve or an asterisk (*) to retrieve all fields.

SELECT [DISTINCT] {* | column_expression, ...}

column_expression can be simply a field name (for example, LAST_NAME).   More complex expressions may include mathematical operations or string manipulation (for example, SALARY * 1.05).   See SQL Expressions for more information.

Separate multiple column expressions with commas (for example, LAST_NAME, FIRST_NAME, HIRE_DATE).

Field names can be prefixed with the table name or alias.   For example, EMP.LAST_NAME or E.LAST_NAME, where E is the alias for the table EMP.

The Distinct operator can precede the first column expression.   This operator eliminates duplicate rows from the result of a query.   For example,

SELECT DISTINCT dep FROM emp

Select statements can also include aggregate functions.

Aggregate Functions

Aggregate functions can also be a part of a <u>Select clause</u>.   Aggregate functions return a single value from a set of records.   An aggregate can be used with a field name (for example, AVG(SALARY)) or in combination with a more complex column expression (for example, AVG(SALARY * 1.07)).   The column expression can be preceded by the Distinct operator.   The Distinct operator eliminates duplicate values from an aggregate expression.   For example,

COUNT (DISTINCT last_name)

In this example, only distinct last name values are counted.

The following are valid aggregates:

Aggregate	Returns
SUM	The total of the values in a numeric field expression. For example, SUM(SALARY) returns the sum of all salary field values.
AVG	The average of the values in a numeric field expression. For example, AVG(SALARY) returns the average of all salary field values.
COUNT	The number of values in any field expression. For example, COUNT(NAME) returns the number of name values. When using COUNT with a field name, COUNT returns the number of non-null field values. A special example is COUNT(*), which returns the number of records in the set, including records with null values.
MAX	The maximum value in any field expression. For example, MAX(SALARY) returns the maximum salary field value.
MIN	The minimum value in any field expression. For example, MIN(SALARY) returns the minimum salary field value.

From Clause

The From clause indicates the tables that will be used in the Select statement.   The format of the From clause is

FROM *table_names* [*table_alias*]

table_names can be one or more simple table names in the current working directory or complete pathnames.

table_alias is a name used to refer to this table in the rest of the Select statement.   Database field names may be prefixed by the table alias.   Given the table specification

FROM emp E

you may refer to the LAST_NAME field as E.LAST_NAME.   Table aliases must be used if the Select statement joins a table to itself.   For example,

SELECT * FROM emp E, emp F WHERE E.mgr_id = F.emp_id

The equal sign (=) includes only matching rows in the results.

If you are joining more than one table, you may use LEFT OUTER JOIN, which includes nonmatching rows in the first table you name. For example,

SELECT * FROM T1 LEFT OUTER JOIN T2 on T1.key = T2.key

Where Clause

The Where clause specifies the conditions that records must meet to be retrieved.   The Where clause contains conditions in the form:

WHERE *expr1 rel_operator expr2*

expr1 and *expr2* may be field names, constant values, or expressions.

rel_operator is the relational operator that links the two expressions.   See SQL Expressions for more information.

For example, the following Select statement retrieves the names of employees that make at least $20,000.

SELECT last_name,first_name FROM emp WHERE salary >= 20000

Group By Clause

The Group By clause specifies the names of one or more fields by which the returned values should be grouped.   This clause is used to return a set of aggregate values.

It has the following form:

GROUP BY *column_expressions*

column_expressions must match the column expression used in the Select clause.   A column expression can be one or more field names of the database table, separated by a comma (,) or one or more expressions, separated by a comma (,).   See SQL Expressions for more information.

The following example sums the salaries in each department.

SELECT dept_id, sum(salary) FROM emp GROUP BY dept_id

This statement returns one row for each distinct department ID.   Each row contains the department ID and the sum of the salaries of the employees in the department.

Having Clause

The Having clause enables you to specify conditions for groups of records (for example, display only the departments that have salaries totaling more than $200,000).   This clause is valid only if you have already defined a Group By clause.

It has the following form:

HAVING *expr1 rel_operator expr2*

expr1 and *expr2* can be field names, constant values, or expressions.   These expressions do not have to match a column expression in the Select clause.

rel_operator   is the relational operator that links the two expressions.   See SQL Expressions for more information.

The following example returns only the departments whose sums of salaries are greater than $200,000:

SELECT dept_id, sum(salary) FROM emp
GROUP BY dept_id HAVING sum(salary) > 200000

Union Operator

The Union operator combines the results of two Select statements into a single result.   The single result is all of the returned records from both Select statements.   By default, duplicate records are not returned. To return duplicate records, use the All keyword (UNION ALL).   The form is

SELECT *statement*

UNION [ALL]

SELECT *statement*

When using the Union operator, the select lists for each Select statement must have the same number of column expressions with the same data types and must be specified in the same order.   For example,

SELECT last_name, salary, hire_date FROM emp

UNION

SELECT name, pay, birth_date FROM person

This example has the same number of column expressions, and each column expression, in order, has the same data type.

The following example is *not* valid because the data types of the column expressions are different (SALARY from EMP has a different data type than LAST_NAME from RAISES).   This example does have the same number of column expressions in each Select statement but the expressions are not in the same order by data type.

SELECT last_name, salary FROM emp

UNION

SELECT salary, last_name FROM raises

Order By Clause

The Order By clause indicates how the records are to be sorted.   The form is

ORDER BY {*sort_expression* [DESC | ASC]}, ...

sort_expression can be field names, expressions, or the positional number of the column expression to use.

The default is to perform an ascending (ASC) sort.

For example, to sort by LAST_NAME you could use either of the following Select statements:

SELECT emp_id, last_name, first_name FROM emp
ORDER BY last_name

or

SELECT emp_id, last_name, first_name FROM emp
ORDER BY 2

In the second example, LAST_NAME is the second column expression following Select, so Order By 2 sorts by LAST_NAME.

For Update Of Clause

The For Update Of clause locks the records of the database table selected by the Select statement.   The form is

FOR UPDATE [OF *column_expressions*]

column_expressions is a list of field names in the database table that you intend to update, separated by a comma (,). Note that *column_expressions* is optional.

The following example returns all records in the employee database that have a SALARY field value of more than $20,000.   When each record is fetched, it is locked.   If the record is updated or deleted, the lock is held until you commit the change.   Otherwise, the lock is released when you fetch the next record.

SELECT * FROM emp WHERE salary > 20000
              FOR UPDATE OF last_name, first_name, salary

SQL Expressions

Expressions are used in the Where clauses, Having clauses, and Order By clauses of SQL Select statements.

Expressions enable you to use mathematical operations as well as character string and date manipulation operators to form complex database queries.

The most common expression is a simple field name.   You can combine a field name with other expression elements.

Other valid expression elements are as follows:
- Constants
- Numeric operators
- Exponential notation
- Character operators
- Date operators
- Relational operators
- Logical operators
- Functions

The order in which these elements are evaluated is described in Operator Precedence.

Constants

Constants are values that do not change.   For example, in the expression PRICE * 1.05, the value 1.05 is a constant.

You must enclose character constants in pairs of single (') or double quotation marks (").   To include a single quotation mark in a character constant enclosed by single quotation marks, use two single quotation marks together (for example, 'Don''t').   Similarly, if the constant is enclosed by double quotation marks, use two double quotation marks to include one.

You must enclose date and time constants in braces ({}), for example, {01/30/89} and {12:35:10}.   The form for date constants is MM/DD/YY or MM/DD/YYYY.   The form for time constants is HH:MM:SS.

The logical constants are .T. and 1 for True and .F. and 0 for False.   For portability, use 1 and 0.

Exponential Notation

You may include exponential notation. For example.

SELECT col1, 3.4E+& FROM table1 WHERE calc < 3.4E-6 * col2

Numeric Operators

You may include the following operators in numeric expressions:

Operator	Meaning
+	Addition
-	Subtraction
*	Multiplication
/	Division
**	Exponentiation
^	Exponentiation

The following table shows examples of numeric expressions.   For these examples, assume SALARY is 20000.

Example	Resulting value
salary + 10000	30000
salary * 1.1	22000
2 ** 3	8

You can precede numeric expressions with a unary plus (+) or minus (-).   For example, -(salary * 1.1) is -22000.

Character Operators

Character expressions may include the following operators:

Operator	Meaning
+	Concatenation keeping trailing blanks.
-	Concatenation moving trailing blanks to the end.

The following chart shows examples of character expressions.   In the examples, LAST_NAME is 'JONES ' and FIRST_NAME is 'ROBERT   '.

Example	Resulting value
first_name + last_name	'ROBERT JONES '
first_name - last_name	'ROBERTJONES '

Note: Some flat-file drivers return character data with trailing blanks as shown in the table.   However, you cannot rely on the driver to return blanks.   Therefore, if you want an expression that works with drivers that do and do not return trailing blanks, use the TRIM function before concatenating strings to make the expression portable.   For example:

TRIM(first_name) + " + TRIM(last_name)

Date Operators

You may include the following operators in date expressions:

Operator	Meaning
+	Add a number of days to a date to produce a new date.
-	The number of days between two dates, or subtract a number of days from a date to produce a new date.

The following chart shows examples of date expressions.   In these examples, hire_date is {01/30/90}.

Example	Resulting value
hire_date + 5	{02/04/90}
hire_date - {01/01/90}	29
hire_date - 10	{01/20/90}

Relational Operators

The relational operators separating the two expressions may be any one of the following:

Operator	Meaning
=	Equal
<>	Not Equal
>	Greater Than
>=	Greater Than or Equal
<	Less Than
<=	Less Than or Equal
Like	Matching a pattern
Not Like	Not matching a pattern
Is Null	Equal to Null
Is Not Null	Not Equal to Null
Between	Range of values between a lower and upper bound.
In	A member of a set of specified values or a member of a subquery.
Exists	True if a subquery returned at least one record.
Any	Compares a value to each value returned by a subquery. Any must be prefaced by =, <>, >, >=, <, or <=. =Any is equivalent to In.
All	Compares a value to each value returned by a subquery. All must be prefaced by =, <>, >, >=, <, or <=.

The following list shows some examples of relational operators:

```
salary <= 40000

dept = 'D101'

hire_date > {01/30/89}

salary + commission >= 50000

last_name LIKE 'Jo%'

salary IS NULL

salary BETWEEN 10000 AND 20000

WHERE salary = ANY (SELECT salary FROM emp WHERE dept = 'D101')

WHERE salary > ALL (SELECT salary FROM emp WHERE dept = 'D101')
```

Logical Operators

Two or more conditions may be combined to form more complex criteria. When two or more conditions are present, they must be related by AND or OR. For example,

salary = 40000 AND exempt = 1

The logical NOT operator is used to reverse the meaning. For example,

NOT (salary = 40000 AND exempt = 1)

Operator Precedence

As expressions become more complex, the order in which the expressions are evaluated becomes important.   The following table shows the order in which the operators are evaluated.   The operators in the first line are evaluated first, then those in the second line, and so on.   Operators in the same line are evaluated left to right in the expression.

Precedence	Operator
1	Unary -, Unary +
2	**
3	*, /
4	+, -
5	=, <>, <, <=, >, >=, Like, Not Like, Is Null, Is Not Null, Between, In, Exists, Any, All
6	Not
7	AND
8	OR

The following example shows the importance of precedence:

WHERE salary > 40000 OR

hire_date > {01/30/89} AND

dept = 'D101'

Because AND is evaluated first, this query retrieves employees in department D101 hired after January 30, 1989, as well as every employee making more than $40,000, no matter what department or hire date.

To force the clause to be evaluated in a different order, use parentheses to enclose the conditions to be evaluated first.   For example,

WHERE (salary > 40000 OR hire_date > {01/30/89})

AND dept = 'D101'

retrieves employees in department D101 that either make more than $40,000 or were hired after January 30, 1989.

Functions

The flat-file drivers support a number of functions that you may use in expressions. In the following topics, the functions are grouped according to the type of result they return.

[Functions that Return Character Strings](#)

[Functions that Return Numbers](#)

[Functions that Return Dates](#)

Functions that Return Character Strings

The following functions return character string values:

Function	Description
CHR	Converts an ASCII code into a one-character string.
	CHR(67) returns C.
RTRIM	Removes trailing blanks from a string.
	RTRIM('ABC ') returns ABC.
TRIM	Removes trailing blanks from a string.
	TRIM('ABC ') returns ABC.
LTRIM	Removes leading blanks from a string.
	LTRIM(' ABC') returns ABC.
UPPER	Changes each letter of a string to uppercase.
	UPPER('Allen') returns ALLEN.
LOWER	Changes each letter of a string to lowercase.
	LOWER('Allen') returns allen.
LEFT	Returns leftmost characters of a string.
	LEFT('Mattson',3) returns Mat.
RIGHT	Returns rightmost characters of a string.
	RIGHT('Mattson',4) returns tson.
SUBSTR	Returns a substring of a string. Parameters are the string, the first character to extract, and the number of characters to extract (optional).
	SUBSTR('Conrad',2,3) returns onr.
	SUBSTR('Conrad',2) returns onrad.
SPACE	Generates a string of blanks.
	SPACE(5) returns ' '.
DTOC	Converts a date to a character string. An optional second parameter determines the format of the result:
	0 (the default) returns MM/DD/YY
	1 returns DD/MM/YY
	2 returns YY/MM/DD
	10 returns MM/DD/YYYY
	11 returns DD/MM/YYYY
	12 returns YYYY/MM/DD
	An optional third parameter specifies the date separator character. If not specified, a slash (/) is used.
	DTOC({01/30/89}) returns 01/30/89
	DTOC({01/30/89}, 0) returns 01/30/89

DTOC({01/30/89}, 1) returns 30/01/89

DTOC({01/30/89}, 2,'-') returns 89-01-30

DTOS	Converts a date to a character string using the format YYYYMMDD.

DTOS({01/23/90}) returns 19900123.

IIF	Returns one of two values. Parameters are a logical expression, the true value, and the false value. If the logical expression evaluates to True, the function returns the true value. Otherwise, it returns the false value.

IIF(salary>20000,'BIG','SMALL') returns BIG if   SALARY is greater than 20000.   If not, it returns SMALL.

STR	Converts a number to a character string. Parameters are the number, the total number of output characters (including the decimal point), and optionally the number of digits to the right of the decimal point.

STR(12.34567,4) returns 12

STR(12.34567,4,1) returns 12.3

STR(12.34567,6,3) returns 12.346

STRVAL	Converts a value of any type to a character string.

STRVAL('Woltman') returns Woltman

STRVAL({12/25/53}) returns 12/25/53

STRVAL (5 * 3) returns 15

STRVAL (4 = 5) returns 'False'

TIME	Returns the time of day as a string.

At 9:49 PM, TIME() returns 21:49:00

USERNAME	Returns the name in the UID attribute as a character string. If UID=Allen, USERNAME() returns Allen.

For Btrieve, the logon ID specified at connect time is returned. For Paradox and Paradox 5 drivers, the user name specified during configuration is returned. For all other flat file drivers, an empty string is returned.

Functions that Return Numbers

The following functions return number values:

Function	Description
MOD	Divides two numbers and returns the remainder of the division. MOD(10,3) returns 1
LEN	Returns the length of a string. LEN('ABC') returns 3
MONTH	Returns the month part of a date. MONTH({01/30/89}) returns 1
DAY	Returns the day part of a date. DAY({01/30/89}) returns 30
YEAR	Returns the year part of a date. YEAR({01/30/89}) returns 1989
MAX	Returns the larger of two numbers. MAX(66,89) returns 89
MIN	Returns the smaller of two numbers. MIN(66,89) returns 66
POW	Raises a number to a power. POW(7,2) returns 49
INT	Returns the integer part of a number. INT(6.4321) returns 6
ROUND	Rounds a number. ROUND(123.456, 0) returns 123 ROUND(123.456, 2) returns 123.46 ROUND(123.456, -2) returns 100
NUMVAL	Converts a character string to a number. If the character string is not a valid number, a zero is returned. NUMVAL('123') returns the number 123
VAL	Converts a character string to a number. If the character string is not a valid number, a zero is returned. VAL('123') returns the number 123

Functions that Return Dates

The following functions return date values:

Function	Description
DATE	Returns today's date.
	If today is 12/25/79, DATE() returns {12/25/79}
TODAY	Returns today's date.
	If today is 12/25/79, TODAY() returns {12/25/79}
DATEVAL	Converts a character string to a date.
	DATEVAL('01/30/89') returns {01/30/89}
CTOD	Converts a character string to a date. An optional second parameter specifies the format of the character string: 0 (the default) returns MM/DD/YY, 1 returns DD/MM/YY, and 2 returns YY/MM/DD.
	CTOD('01/30/89') returns {01/30/89}
	CTOD('01/30/89',1) returns {30/01/89}

The following examples use some of the number and date functions.

Retrieve all employees that have been with the company at least 90 days:

```
SELECT first_name, last_name FROM emp
WHERE DATE() - hire_date >= 90
```

Retrieve all employees hired in January of this year or last year:

```
SELECT first_name, last_name FROM emp
WHERE MONTH(hire_date) = 1 AND (YEAR(hire_date) = YEAR(DATE())
OR YEAR(hire_date) = YEAR(DATE()) - 1)
```

Create Table Statement

The Create Table statement is used to create database files.   The form of the Create Table statement is

CREATE TABLE *filename* (*col_definition*[,*col_definition, ...*])

filename can be a simple filename or a full pathname.   A simple filename is preferred for portability to other SQL data sources.   If it is a simple filename, the file is created in the directory you specified as the database directory in the connection string.   If you did not specify a database directory in the connection string, the file is created in the directory you specified as the database directory in ODBC.INI.   If you did not specify a database directory in either place, the file is created in the current working directory at the time you connected to the driver.

col_definition is the column name, followed by the data type, followed by an optional column constraint definition.   Values for column names are database specific.   The data type specifies a column's data type.

The only column constraint definition currently supported by some flat-file drivers is "not null."   Not all flat-file tables support "not null" columns.   In the cases where not null is not supported, this restriction is ignored and the driver returns a warning if "not null" is specified for a column.   The "not null" column constraint definition is allowed in the driver so that you can write a database-independent application (and not be concerned about the driver raising an error on a Create Table statement with a "not null" restriction).

A sample Create Table statement to create an employee database table is

CREATE TABLE emp (last_name CHAR(20) NOT NULL,
            first_name CHAR(12) NOT NULL,
            salary NUMERIC (10,2) NOT NULL,
            hire_date DATE NOT NULL)

Drop Table Statement

The Drop Table statement is used to delete database files.   The form of the Drop Table statement is

DROP TABLE *filename*

filename may be a simple filename (EMP) or a full pathname.   A simple filename is preferred for portability to other SQL data sources.   If it is a simple filename, the file is dropped from the directory you specified as the database directory in the connection string.   If you did not specify a database directory in the connection string, the file is deleted from the directory you specified as the database directory in ODBC.INI.   If you did not specify a database directory in either of these places, the file is dropped from the current working directory at the time you connected to the driver.

A sample Drop Table statement to delete the employee database table is

DROP TABLE emp

Insert Statement

The SQL Insert statement is used to add new records to a database table. With it, you can specify either of the following:
- A list of values to be inserted as a new record
- A Select statement that copies data from another table to be inserted as a set of new records

The form of the Insert statement is

INSERT INTO *table_name* [(*col_name, ...*)]
VALUES (*expr, ...*)

table_name may be a simple filename or a full pathname.   A simple filename is preferred for portability to other SQL data sources.

col_name is an optional list of column names giving the name and order of the columns whose values are specified in the Values clause.   If you omit *col_name*, the value expressions (*expr*) must provide values for all columns defined in the file and must be in the same order that the columns are defined for the file.

expr is the list of expressions giving the values for the columns of the new record.   Usually, the expressions are constant values for the columns.   Character string values must be enclosed in single or double quotation marks, date values must be enclosed in braces {}, and logical values that are characters must be enclosed in periods (for example, .T. or .F.).

An example of an Insert statement that uses a list of expressions is

INSERT INTO emp (last_name, first_name, emp_id, salary, hire_date)
VALUES ('Smith', 'John', 'E22345', 27500, {4/6/91})

Each Insert statement adds one record to the database table.   In this case a record has been added to the employee database table, EMP.   Values are specified for five columns.   The remaining columns in the table are assigned a blank value, meaning Null.

select_statement is a query that returns values for each *col_name* value specified in the column name list. Using a Select statement instead of a list of value expressions lets you select a set of rows from one table and insert it into another table using a single Insert statement.

An example of an Insert statement that uses a Select statement is:

INSERT INTO emp1 (first_name, last_name, emp_id, dept, salary)
SELECT first_name, last_name, emp_id, dept, salary from emp
   WHERE dept = D050

In this type of Insert statement, the number of columns to be inserted must match the number of columns in the Select statement. The list of columns to be inserted must corresond to the columns in the Select statement just as it would to a list of value expressions in the other type of Insert statement. That is, the first column inserted corresponds to the first column selected; the second inserted to the second, etc.

The size and data type of these corresponding columns must be compatible. Each column in the Select list should have a data type that the ODBC driver accepts on a regular Insert/Update of the corresponding column in the Insert list. Values are truncated when the size of the value in the Select list column is greater than the size of the corresponding Insert list column.

The *select_statement* is evaluated before any values are inserted. This query cannot be made on the table into which values are inserted.

Update Statement

The SQL Update statement is used to change records in a database file.   The form of the Update statement supported for flat-file drivers is

UPDATE *filename* SET *col_name = expr, ...*
[WHERE { *conditions* | CURRENT OF *cursor_name* }]

filename may be a simple filename or a full pathname.   A simple filename is preferred for portability to other SQL data sources.

col_name is the name of a column whose value is to be changed.   Several columns can be changed in one statement.

expr is the new value for the column.   The expression can be a constant value or a subquery.   Character string values must be enclosed with single or double quotation marks, date values must be enclosed by braces {}, and logical values that are letters must be enclosed by periods (for example, .T.   or .F.).   Subqueries must be enclosed in parentheses.

The Where clause is any valid clause.   It determines which records are to be updated.

The Where Current Of *cursor_name* clause can be used only by developers coding directly to the ODBC API.   It causes the row at which *cursor_name* is positioned to be updated.   This is called a "positioned update."   You must first execute a Select...For Update statement with a named cursor and fetch the row to be updated.

An example of an Update statement on the employee table is

UPDATE emp SET salary=32000, exempt=1
WHERE emp_id = 'E10001'

The Update statement changes every record that meets the conditions in the Where clause.   In this case the salary and exempt status is changed for all employees having the employee ID E10001.   Since employee IDs are unique in the employee table, only one record is updated.

An example using a subquery is

UPDATE emp SET salary = (SELECT avg(salary) from emp)
WHERE emp_id = 'E10001'

In this case, the salary is changed to the average salary in the company for the employee having employee ID E10001.

Delete Statement

The SQL Delete statement is used to delete records from a database table.   The form of the Delete statement supported for flat-file drivers is

DELETE FROM *filename*

[WHERE { *conditions* | CURRENT OF *cursor_name* }]

filename may be a simple filename or a full pathname.   A simple filename is preferred for portability to other SQL data sources.

The Where clause is any valid clause.   It determines which records are to be deleted.   If you include only the keyword Where, all records in the table are deleted but the file is left intact.

The Where Current Of *cursor_name* clause can be used only by developers coding directly to the ODBC API.   It causes the row at which *cursor_name* is positioned to be deleted.   This is called a "positioned delete."   You must first execute a Select...For Update statement with a named cursor and fetch the row to be deleted.

An example of a Delete statement on the employee table is

DELETE FROM emp WHERE emp_id = 'E10001'

Each Delete statement removes every record that meets the conditions in the Where clause.   In this case every record having the employee ID E10001 is deleted.   Since employee IDs are unique in the employee table, at most one record is deleted.

Reserved Keywords

The following words are reserved for use in SQL statements.   If they are used for file or column names in a database that you use, you must enclose them in quotation marks in any SQL statement where they appear as file or column names.

ALL	FULL	NOT
AND	GROUP	NULL
BETWEEN	HAVING	ON
COMPUTE	INNER	OPTIONS
CROSS	INTO	OR
DISTINCT	LEFT	ORDER
FOR	LIKE	RIGHT
FROM	NATURAL	WHERE

DB2/2 Driver

Two DB2/2 drivers are available. The new DB2/2 driver supports the use of the Client Application Enabler (CAE) 1.2 interfaces and the previous DB2/2 driver supports the use of CAE 1.0 interfaces and pcdrdll.dll. Note that the previous DB2/2 driver, QEDBM*nn*.DLL, is supported in this release but will not be supported in the next release of INTERSOLV ODBC drivers.

The DB2/2 driver supports the following database systems:
- DB2/2 for OS/2 (formerly called Database Manager for OS/2)
- DB2/6000
- SQL/400 (through the DDCS/2 or DDCS/6000 gateway)
- MVS DB2 (through the DDCS/2 or DDCS/6000 gateway)
- SQL/DS (through the DDCS/2 or DDCS/6000 gateway)

Note   The filename for the DB2/2 driver has been changed from QEDBM*nn*.DLL to QEDB2*nn*.DLL.

System Requirements

System Requirements for Users of QEDBM*nn*.DLL Only

Configuring Data Sources

DB2/2 and DDCS/2 Gateway

DB2/6000 and DDCS/6000 Gateway

Connecting to DB2/2 Using a Logon Dialog Box

Connecting to DB2/2 Using a Connection String

Data Types

Isolation and Lock Levels Supported

ODBC Conformance Level

Number of Connections and Statements Supported

Note   The previous version of this driver (using CAE 1.0 interfaces and pcdrdll.dll), QEDBM*nn*.DLL, is supported in this release, version 2.10 of the drivers, but will not be supported in the next release of INTERSOLV ODBC drivers.

System Requirements

You must have a DB2 Client Application Enabler (CAE) or DOS/Windows Requester to access the DB2 family of databases.

There are three distinct types of DB2/2 clients for Windows:
- The DOS/Windows Requester (formerly Database Manager Client) and CAE versions prior to 1.2
- DB2/6000 or DDCS/6000 using CAE
- CAE version 1.2

System requirements are as follows:
- The DB2/2 Server must be installed as the Server Version (*not* the Local Version).
- To use the DDCS/2 gateway, you must install the DDCS/2 Multi-User Edition (*not* the Single-User Edition).
- DB2/2 requires network protocols from products such as Communications Manager from Extended Services version 1.0, Communications Manager/2 version 1.0, Network Transport Services/2 version 1.0, LAN Services,   Novell NetWare, or numerous TCP/IP vendors.
- To connect to DB2/6000, use the CAE for DOS and Windows that comes with DB2/6000 or the CAE 1.2.   You cannot use any version of the CAE prior to 1.2.   DB2/6000 does not require a gateway. The DB2/6000 CAE for DOS/Windows and the CAE 1.2 connect directly to DB2/6000 via TCP/IP.
- To connect to DDCS/6000, use the CAE for DOS and Windows that comes with DB2/6000 or CAE 1.2.   You cannot use any version of the DOS/Windows Requester prior to 1.2.   The DB2/6000 CAE for DOS/Windows and the CAE 1.2 connect directly to DB2/6000 via TCP/IP.

System Requirements for Users of QEDBM*nn*.DLL Only

To use versions of the DB2/2 CAE prior to 1.2, you must first modify the Windows Client Enabler configuration file, dbdrqlib.cfg, located in the dbdrqlib directory.   Three parameters must be changed from their default values for the Database Requester to work properly, as follows:

SQLSIZE=4096/512000
SQLCODEPAGE=*code_page_number*
SQLNAME=*YOUR_NAME*

Change SQLSIZE exactly as shown.

The SQLCODEPAGE setting must match the code page number in which the DB2/2 database was created.

SQLNAME must be set to the SQL name assigned to you by your system administrator and must be entered in uppercase letters.

Note: When using DB2/6000 or DDCS/6000 with CAE, or when using CAE version 1.2 for DB2/2, you must catalog the protocol node (usually TCP/IP, NetBios, or IPX/SPX) before you catalog the database. You need to use the DB2 command line interface that comes from the CAE.   For details, see the *DATABASE 2 Client Application Enabler/DOS User's Guide.*

CAE version 1.2 and the DB2/6000 CAE do not have a DBDRQLIB.CFG file.

pcdrdll.dll must be either in your Windows SYSTEM directory or the dbdrqlib directory must be on your search path.

If you attempt to configure a data source and you do not have PCDRDLL.DLL on your path or in your Windows SYSTEM directory, the following message appears:

The setup routines for the INTERSOLV DB2/2 ODBC driver could not be loaded. You may be low on memory and need to quit a few applications.

Configuring Data Sources

To configure a DB2/2 data source, do the following:

1 Start the ODBC Administrator by double-clicking the ODBC icon in the the Windows Control Panel. A list of data sources appears.

2 If you are configuring a new data source, click **Add**. A list of installed drivers appears. Select INTERSOLV DB2/2 and click **OK**.

If you are configuring an existing data source, select the data source name and click **Setup**.

The setup dialog box appears.

3 Specify values as follows:

Data Source Name: A string that identifies this DB2/2 data source configuration in ODBC.INI. Examples include "Accounting" or "DB2/2-Serv1."

Description: An optional long description of a data source name. For example, "My Accounting Database" or "DB2/2 on Server #1."

Database Name: The name of the database to which you want to connect by default.

The following values are optional:

Owner of Catalog Tables: On most DB2/2 systems, SYSIBM is the owner of the catalog system tables. If you have read access to the system tables, you do not need to change this option.

If you do not have read access, the system administrator must create a view of the system tables in another account and give you permission to use that view. In this case, specify the Authorization ID for the account that owns the views of the system tables.

If you are using DDCS/2 to connect to a SQL/400 database, specify a comma-separated list of the Collection IDs to which you want to have access in your ODBC application.

If you are using DDCS/2 to connect to a SQL/DS database, specify SYSTEM as the owner.

Groups: A value to determine which tables you can access. Your system administrator may have placed you in a "group" of users and granted table access to the entire group. If this is the case, specify the names of any groups to which you belong; separate each name with a comma. Alternatively, specify the word ALL so that you see all table names even if you cannot access the table.

Default Authorization ID: The default Logon ID used to connect to your DB2/2 database. A Logon ID is required only if security is enabled on your database. Your ODBC application may override this value or you may change this value in the logon dialog box.

Cursor Behavior: Select Preserve if you want cursors to be held at the current position when the transaction ends. Otherwise, leave this set to Close. Selecting Preserve may impact the performance of your database operations. This setting does not apply to SQL/DS.

When you select Preserve, the driver returns SQL_CB_PRESERVE from SQLGetInfo (SQL_CURSOR_COMMIT_BEHAVIOR). But only Select statements and prepared Update or Delete . . . Where Current of Cursor statements are preserved when the transaction ends. All other prepared statements are closed and deleted.

4 You must catalog any database you want to access. To do so, click the **Catalog Db** button (QEDBMnn.DLL users click the **Catalog Remote Database** button). The Catalog Db dialog box appears.

In this dialog box, type an alias name, the name of the database, and the name of the node (like an alias for the remote node where the server is installed and the protocol used to access it). Click **Add**. The alias name you added will be listed in the Database Name drop-down list in the logon dialog box when you exit this dialog box.

This dialog box also lets you uncatalog remote databases.   To do this, select the alias you want to delete from the Alias Name drop-down list and click **Delete**.

5   To provide information about the remote site for the DB2/2 databases, click **Catalog Node**. (QEDBMnn.DLL users do not have this option). In the Catalog Node dialog box, select a node name (only names of previously cataloged nodes are available, the name of the server or number of the adapter (for NetBIOS nodes), the name of the remote node where the DB2 server is installed, and the type of communication protocol to use to access the node. Optionally, enter a comment. Click **Add**.

This dialog box also lets you uncatalog remote databases.   To do this, select the alias you want to delete from the Alias Name drop-down list and click **Delete**.

This dialog box also lets you uncatalog nodes.   To do this, select the alias you want to delete from the Node Name drop-down list and click **Delete**.

6   Click **Translate** to perform a translation of your data from one character set to another. The Select Translator dialog box appears in which you can select a translator to peform the translation. INTERSOLV provides a translator named INTERSOLV OEM ANSI that translates your data from the IBM PC character set to the ANSI character set. Click **OK** to leave this dialog box and perform the translation.

The translators that are listed in this dialog box are determined by the values listed in the ODBC Translators section of your ODBCINST.INI file.

7   Click **OK** to write these values to ODBC.INI. These values are now the defaults when you connect to the data source.   You can change the defaults by configuring your data source again.   You can override the defaults by connecting to the data source using a connection string with alternate values.

DB2/2 and DDCS/2 Gateway

Access to DB2/2 databases, or other databases via the DDCS/2 gateway, requires that you bind to tables and grant privileges to all users.   These processes are described in the following topics.

Binding

Before you can access a table, you must bind to the database. You must bind to every database you intend to access.   o do this, enter the following commands on the server:

SQLBIND QECSV1.BND *database_name*
SQLBIND QERRV1.BND *database_name*
SQLBIND QEURV1.BND *database_name*
SQLBIND QECSWHV1.BND *database_name*
SQLBIND QERRWHV1.BND *database_name*
SQLBIND QEURWHV1.BND *database_name*

You may need to copy the bind files from the installation directory to the server.

database_name is the name of the database you want to access.

Note for SQL/DS Users: For binding to SQL/DS databases, enter only the first three of these commands.   The files QECSWHV1.BND, QERRWHV1.BND, and QEURWHV1.BND will not work with SQL/DS.

For SQL/400 Users

If you receive the error

'QSYS.QE' is an undefined name

you must create a collection named QE.   To do this, enter the following command on the host AS/400:

CREATE COLLECTION QE

Then issue the SQLBIND commands again.

Granting Privileges

Users must be granted the necessary privileges before they can access a database or table through a gateway.

DB2/2 users are assigned an Authorization ID.   There are three categories of Authorization IDs: administrator, local administrator, and user.   The third category, user, must be granted the privilege to access DB2/2 tables.

To grant the necessary privileges, an administrator must run Query Manager or the DBM Command Line Interface and execute the appropriate GRANT statement.

For DB2/2 databases and databases that reside on DDCS/2 connecting to DB2:

GRANT EXECUTE, BIND ON PROGRAM QE.QECSV1 TO *authorization_list*

GRANT EXECUTE, BIND ON PROGRAM QE.QERRV1 TO *authorization_list*

GRANT EXECUTE, BIND ON PROGRAM QE.QEURV1 TO *authorization_list*

GRANT EXECUTE, BIND ON PROGRAM QE.QECSWHV1 TO *authorization_list*

GRANT EXECUTE, BIND ON PROGRAM QE.QERRWHV1 TO *authorization_list*

GRANT EXECUTE, BIND ON PROGRAM QE.QEURWHV1 TO *authorization_list*

For databases that reside on the DDCS/2 gateway connecting to SQL/DS:

GRANT RUN ON QE.QECSV1 TO *authorization_list*

GRANT RUN ON QE.QERRV1 TO *authorization_list*

GRANT RUN ON QE.QEURV1 TO *authorization_list*

GRANT RUN ON QE.QECSWHV1 TO *authorization_list*

GRANT RUN ON QE.QERRWHV1 TO *authorization_list*

GRANT RUN ON QE.QEURWHV1 TO *authorization_list*

For databases that reside on the DDCS/2 gateway connecting to SQL/400:

GRANT RUN ON PROGRAM QE.QECSV1 TO *authorization_list*

GRANT RUN ON PROGRAM QE.QERRV1 TO *authorization_list*

GRANT RUN ON PROGRAM QE.QEURV1 TO *authorization_list*

GRANT RUN ON PROGRAM QE.QECSWHV1 TO *authorization_list*

GRANT RUN ON PROGRAM QE.QERRWHV1 TO *authorization_list*

GRANT RUN ON PROGRAM QE.QEURWHV1 TO *authorization_list*

authorization_list is a list of comma-separated Authorization IDs or Group IDs, or PUBLIC if you want to grant access to all users.

DB2/6000 and DDCS/6000 Gateway

Before you can access a table via DB2/6000 or the DDCS/6000 gateway, you must bind to the database and grant each user access to it.   To do this, enter the following commands on the client computer from the DBM Command Line Interface:

BIND QECSV1.BND BLOCKING ALL GRANT *authorization_list*
BIND QERRV1.BND BLOCKING ALL GRANT *authorization_list*
BIND QEURV1.BND BLOCKING ALL GRANT *authorization_list*
BIND QECSWHV1.BND BLOCKING ALL GRANT *authorization_list*
BIND QERRWHV1.BND BLOCKING ALL GRANT *authorization_list*
BIND QEURWHV1.BND BLOCKING ALL GRANT *authorization_list*

authorization_list is a list of comma-separated Authorization IDs or Group IDs, or PUBLIC if you want to grant access to all users.

Note for SQL/DS Users: For binding to SQL/DS databases, enter only the first three of these commands.   The files QECSWHV1.BND, QERRWHV1.BND, and QEURWHV1.BND will not work with SQL/DS.

For SQL/400 Users

If you receive the error

'QSYS.QE' is an undefined name

you must create a collection named QE.   To do this, enter the following command on the host AS/400:

CREATE COLLECTION QE

Then issue the BIND commands again.

Connecting to DB2/2 Using a Logon Dialog Box

Some ODBC applications display a logon dialog box when you are connecting to a data source.   In these cases, the data source name has already been specified.

In the logon dialog box, do the following:

1   Type the name of the remote database or select the name of the remote database from the Database Name drop-down list.

You must have cataloged any database you want to access from the client.   (See the topic Configuring Data Sources for information on how to do this.)

2   If required, type your user name (authorization ID).

3   If required, type your password.

If you leave your user name and password blank, the ODBC application assumes you have already logged on using SQLLOGN2 (under DOS) or using User Profile Management (under OS/2).   If you have not, the application returns an error.   You must either type your user name and password in the dialog box or log on using SQLLOGN2 and STARTDRQ (under DOS) or using User Profile Management (under OS/2).   See the *IBM Extended Services for OS/2 Guide to DB2/2 Clients* for more information.

4   Click **OK** to complete the logon and to update the values in ODBC.INI.

Connecting to DB2/2 Using a Connection String

If your application requires a connection string to connect to a data source, you must specify the data source name that tells the driver which ODBC.INI section to use for the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in ODBC.INI.   These values are not written to ODBC.INI.

You can specify either long or short names in the connection string.   The connection string has the form:

DSN=*data_source_name*[;*attribute=value*[;*attribute=value*]...]

An example of a connection string for DB2/2 is

DSN=DB22 TABLES;DB=PAYROLL;UID=JOHN;PWD=XYZZY;GRP=ACCTNG

The following table gives the long and short names for each attribute, as well as a description.

The defaults listed in the table are initial defaults that apply when no value is specified in either the connection string or in the data source definition in ODBC.INI.   If you specified a value for the attribute when configuring the data source, that value is your default.

Attribute	Description
DataSourceName (DSN)	Identifies a DB2/2 data source configuration in ODBC.INI. Examples include "Accounting" or "DB2/2-Serv1."
Database (DB)	Name of the database to which you want to connect.
Sysibm (SI)	On most DB2/2 systems, SYSIBM is the owner of the catalog system tables. If you have read access to the system tables, you do not need to change this option. If you do not have read access, the database administrator must create a view of the system tables in another account and give you permission to use that view. In this case, specify the Authorization ID for the account that owns the views of the system tables. If you are using DDCS/2 to connect to a SQL/400 database, specify a comma-separated list of the Collection IDs to which you want to have access in your ODBC application. If you are using DDCS/2 to connect to a SQL/DS database, specify SYSTEM as the owner.
Groups (GRP)	Determines which tables you can access. Your system administrator may have placed you in a "group" of users and granted table access to the entire group. If this is the case, set Groups to the names of any groups to which you belong; separate each name with a comma. Alternatively, Groups=ALL lets your application see all table names even if you cannot access the table.
LogonID (UID)	Default logon ID used to connect to your DB2/2 database. A logon ID is required only if security is enabled on your database. If so, contact your system administrator to get your logon ID.
Password (PWD)	Password.
CursorBehavior	CursorBehavior={0	1}. Determines whether cursors

(CB)   are preserved or closed at the end of each transaction.   Valid values are 1 (preserve) and 0 (close, the initial default).   Set this attribute to 1 if you want cursors to be held at the current position when the transaction ends.   Doing so may impact the performance of your database operations.

This attribute is not valid for SQL/DS.

When you select Preserve, the driver returns SQL_CB_PRESERVE from SQLGetInfo (SQL_CURSOR_COMMIT_BEHAVIOR). But only Select statements and prepared Update or Delete . . . Where Current of Cursor statements are preserved when the transaction ends. All other prepared statements are closed and deleted.

Data Types

The DB2/2 data types map to the standard ODBC data types as follows.

DB2/2	ODBC
Char	SQL_CHAR
Char() for Bit Data	SQL_BINARY
Date	SQL_DATE
Decimal	SQL_DECIMAL
Float	SQL_DOUBLE
Integer	SQL_INTEGER
Long Varchar[1]	SQL_LONGVARCHAR
Long Varchar for Bit Data[1]	SQL_LONGVARBINARY
Smallint	SQL_SMALLINT
Time	SQL_TIME
Timestamp	SQL_TIMESTAMP
Varchar[2]	SQL_VARCHAR
Varchar[3]	SQL_LONGVARCHAR
Varchar() for Bit Data[2]	SQL_VARBINARY
Varchar() for Bit Data[3]	SQL_LONGVARBINARY

[1] Not supported for SQL/400

[2] For DB2/2, DB2/6000, and SQL/400; SQL/DS and MVS DB2 up to 254

[3] For SQL/DS and MVS DB2 over 254

Note   The Graphic, Vargraphic, and Long Vargraphic data types are not supported.

Isolation and Lock Levels Supported

DB2/2 supports isolation levels 0 (read uncommitted), 1 (read committed), and 2 (repeatable read).   It supports record-level locking.

ODBC Conformance Level

The DB2/2 driver supports the Core, Level 1, and Level 2 API functions listed in [Supported ODBC Functions](#). The driver supports the minimum SQL grammar.

Number of Connections and Statements Supported

The DB2/2 family of database systems supports a single connection and multiple statements per connection.

ALLBASE Driver

The ALLBASE driver supports the HP ALLBASE/SQL and HP IMAGE/SQL database systems.

The driver filename is QEHP*nn*.DLL.

[System Requirements](#)

[Configuring Data Sources](#)

[Connecting to ALLBASE Using a Logon Dialog Box](#)

[Connecting to ALLBASE Using a Connection String](#)

[Data Types](#)

[Isolation and Lock Levels Supported](#)

[ODBC Conformance Level](#)

[Number of Connections and Statements Supported](#)

System Requirements

The ALLBASE driver requires that you use the HP ALLBASE PC API product.   Consult the *HP ALLBASE/SQL PC API User's Guide* (HP part number B2463-90001) for proper installation instructions.

If you attempt to configure a data source and you do not have SQLAPIW.DLL on your path or in your Windows SYSTEM directory, the following message appears:

The setup routines for the INTERSOLV ALLBASE ODBC driver could not be loaded. You may be low on memory and need to quit a few applications.

Configuring Data Sources

To configure an ALLBASE data source, do the following:

1 Start the ODBC Administrator by double-clicking the ODBC icon in the the Windows Control Panel. A list of data sources appears.

2 If you are configuring a new data source, click **Add**. A list of installed drivers appears. Select INTERSOLV AllBase and click **OK**.

If you are configuring an existing data source, select the data source name and click **Setup**.

The setup dialog box appears.

3 Specify values as follows:

Data Source Name: A string that identifies this ALLBASE data source configuration in ODBC.INI. Examples include "Accounting" or "ALLBASE-Serv1."

Description: An optional long description of a data source name. For example, "My Accounting Database" or "ALLBASE Database on Server #1."

Database Name: The name of the database to which you want to connect by default.

The following settings are optional:

Default User Name: The default user name used to connect to your ALLBASE database. This user name is case-sensitive. A user name is required only if security is enabled on your database. Your ODBC application may override this value or you may override this value in the Logon dialog box or connection string.

Yield Proc: A value of 1, 2, or 3 that determines whether you can work in other applications when ALLBASE is busy. This attribute is useful to users of ODBC applications. Valid values are:

- 0 (peek and dispatch), which causes the driver to check the Windows message queue and send any messages to the appropriate Windows application.
- 1 (no yielding, the default), which does not let you work in other applications.
- 2 (ALLBASE's yield procedure), which uses ALLBASE's default yield procedure.
- 3 (dispatch via Windows Yield function), which turns control over to the Windows kernel. The Windows kernel checks the message queue and sends any messages to the appropriate application window.

It is recommended that you use the value 1.

4 Click **OK** to write these values to ODBC.INI. These values are now the defaults when you connect to the data source. You can change the defaults by configuring your data source again. You can override the defaults by connecting to the data source using a connection string with alternate values.

5 Click **Translate** to perform a translation of your data from one character set to another. The Select Translator dialog box appears in which you can select a translator to peform the translation. INTERSOLV provides a translator named INTERSOLV OEM ANSI that translates your data from the IBM PC character set to the ANSI character set. Click **OK** to leave this dialog box and perform the translation.

The translators that are listed in this dialog box are determined by the values listed in the ODBC Translators section of your ODBCINST.INI file.

Connecting to ALLBASE Using a Logon Dialog Box

Some ODBC applications display a logon dialog box when you are connecting to a data source.   In these cases, the data source name has already been specified.

In the logon dialog box, do the following:

1   If you want to access a database other than the one specified in ODBC.INI or if no default database is specified, type the name of the database you want to access, or select the name from the Database Name drop-down list.

2   If required, type your case-sensitive user name.

3   If required, type your case-sensitive password.

4   Click **OK** to complete the logon.   The name of the database and your user name are saved in ODBC.INI and displayed the next time you log on.

Connecting to ALLBASE Using a Connection String

If your application requires a connection string to connect to a data source, you must specify the data source name that tells the driver which ODBC.INI section to use for the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in ODBC.INI.   These values are not written to ODBC.INI.

You can specify either long or short names in the connection string.   The connection string has the form:

DSN=*data_source_name*[;*attribute=value*[;*attribute=value*]...]

An example of a connection string for ALLBASE is

DSN=ACCOUNTING;DB=PAYROLL;UID=JIM;PWD=XYZZY

The following table gives the long and short names for each attribute, as well as a description.

The defaults listed in the table are initial defaults that apply when no value is specified in either the connection string or in the data source definition in ODBC.INI.   If you specified a value for the attribute when configuring the data source, that value is your default.

Attribute	Description
DataSourceName (DSN)	Identifies an ALLBASE data source configuration in ODBC.INI. Examples include "Accounting" or "ALLBASE-Serv1."
LogonID (UID)	Default Logon ID (user name) used to connect to your ALLBASE database. This ID is case-sensitive. A Logon ID is required only if security is enabled on your database. If so, contact your system administrator to get your Logon ID.
Password (PWD)	Case-sensitive password.
Database (DB)	Name of the database to which you want to connect.
YieldProc (YLD)	YieldProc={0 | 1 | 2 | 3} Determines whether you can work in other applications when ALLBASE is busy. This attribute is useful to users of ODBC applications. ▪ YieldProc=0 (peek and dispatch) causes the driver to check the Windows message queue and send any messages to the appropriate Windows application. ▪ YieldProc=1 (no yielding) does not let you work in other applications. ▪ YieldProc=2 (default yield procedure) uses ALLBASE's default yield procedure. ▪ YieldProc=3 (dispatch via Windows Yield function) turns control over to the Windows kernel. The Windows kernel checks the message queue and sends any messages to the appropriate application window. It is recommended that you use YieldProc=1, the initial default. YieldProc=0, YieldProc=2, and YieldProc=3 do not work with all Windows applications.

Data Types

ALLBASE data types map to the standard ODBC data types as follows:

ALLBASE	ODBC
Char	SQL_VARCHAR
Varchar	SQL_VARCHAR
Date	SQL_DATE
Datetime	SQL_TIMESTAMP
Float(24)	SQL_REAL
Double Precision	SQL_DOUBLE
Decimal	SQL_DECIMAL
Integer	SQL_INTEGER
Smallint	SQL_SMALLINT
Time	SQL_TIME

Isolation and Lock Levels Supported

ALLBASE supports isolation levels 1 (read committed) and 3 (serializable).   The default is 1.   ALLBASE supports page-level locking.

ODBC Conformance Level

The ALLBASE driver supports the Core, Level 1, and Level 2 API functions listed in Supported ODBC Functions.   Also, the ALLBASE driver supports the SQLSetPos function, as well as backward and random fetching in SQLExtendedFetch.

The driver supports the minimum SQL grammar.

Number of Connections and Statements Supported

The HP/ALLBASE and HP/IMAGE database systems support a single connection and multiple statements per connection.

Btrieve Driver

The Btrieve driver supports Btrieve version 5.0 and higher.   The driver executes SQL statements directly on Btrieve databases.

The driver filename is QEBTR*nn*.DLL.

[System Requirements](#)

[Managing Databases](#)

[Installing Btrieve to Process Transactions](#)

[Configuring Data Sources](#)

[Defining Table Structure](#)

[Connecting to Btrieve Using a Connection String](#)

[Data Types](#)

[Indexes](#)

[Select Statement](#)

[Create and Drop Index Statements](#)

[Isolation and Lock Levels Supported](#)

[ODBC Conformance Level](#)

[Number of Connections and Statements Supported](#)

System Requirements

To access a Btrieve database, you must be using the following software:
- WDDLSVCS.DLL dynamic link library.
- WBTRCALL.DLL dynamic link library.   There are two versions of WBTRCALL.DLL: networked and non-networked.   INTERSOLV supplies the non-networked version when you install the Btrieve driver. If you want network access, you must get the networked version from Btrieve Technologies.

Before you attempt to access Btrieve files, you must follow these steps:

1   Incorporate existing Btrieve files into a Scalable SQL database.   To do this, see the topic Defining Table Structure.

2   Make sure the WBTRCALL.DLL and WDDLSVCS.DLL dynamic link libraries are in a directory on your DOS path or in the Windows SYSTEM directory.

Managing Databases

If you already use Scalable SQL, the Btrieve driver can access your Scalable SQL databases directly.   If not, your Btrieve files must be incorporated into a Scalable SQL database.

A Scalable SQL database is composed of data files that contain your records and data dictionary files that describe the database.   The data files are Btrieve files.   The data dictionary files are special Btrieve files that contain descriptions of the data files, views, fields, and indexes in your database.

All Btrieve files in a Scalable SQL database must reside in the same directory.   In addition to the Btrieve data files, the three data dictionary files, FILE.DDF, FIELD.DDF, and INDEX.DDF, also must be in the directory.

Incorporating a Btrieve file into a Scalable SQL database does not change the Btrieve file in any way. You can continue to access the file directly with any existing Btrieve application.

Installing Btrieve to Process Transactions

The Btrieve driver supports transactions.   A transaction is a series of database changes that is treated as a single unit.   In applications that don't use transactions, the Btrieve driver immediately executes Insert, Update, and Delete statements on the database files and the changes are automatically committed when the SQL statement is executed.   You cannot undo these changes.   In applications that use transactions, the Btrieve driver holds inserts, updates, and deletes until you issue a COMMIT or ROLLBACK.   A COMMIT saves the changes to the database file; a ROLLBACK undoes the changes.

Transactions affect the removal of record locking.   All locks are removed when SQLTransact is called with the Commit or Rollback option to end the active transaction.

To use the Btrieve driver's transaction processing capabilities, you must change the required settings for Btrieve in the WIN.INI file. if you are using Btrieve 6.0 or higher, the NOVDB.INI file.   Edit your WIN.INI or NOVDB.INI file to contain the following options setting in the [btrieve] section:

options=/m:64 /p:4096 /t:c:\windows_directory\bbt.trn /n:12

This setting specifies the following options:

/m   The total size, in kilobytes, of Btrieve's data area.   64 is the
        maximum value that can be specified.

/p   The maximum page size of any Btrieve file you want to access.
        4096 is the maximum value that can be specified.

/t   The pathname to the transaction control file used to process
        transactions.   In a network environment, the transaction control
        file should be on a network disk drive, and all users who start
        Btrieve should use the same file.

/n   The maximum number of files involved per transaction.   This
        number must not exceed 18.

The complete set of options are documented in the *Btrieve for Windows Installation and Operations Manual*, in the section "Starting the Record Manager."

For example, to have a maximum of 16 files per transaction and to place the transaction control file in F:\BTRANS\BTRIEVE.TRN, use these settings in the options specification:

/n:16 /t:f:\btrans\btrieve.trn

Add a [brequestDPMI] section to the appropriate INI file to initialize the requester DLL.   For example,

[brequestDPMI]
datalength=4096
chkparms=no
local={yes | no}
tasks=10

The datalength setting must be at least as large as the largest record to be accessed.

Set local=no if you are accessing network files and WBTRCALL.DLL is the networked version.   Set local=yes if you are accessing local files and the non-networked version of WBTRCALL.DLL is on your disk.   If you want to access both network and local files, you must have the networked version of WBTRCALL.DLL, rename the local version of WBTRCALL.DLL to WBTRLOCL.DLL, and set local=yes. If you are using a requester with a version earlier than 6.0A, the local=yes line must not have anything after the yes (no spaces, no comment).

Configuring Data Sources

To configure a Btrieve data source, do the following:

1   Start the ODBC Administrator by double-clicking the ODBC icon in the the Windows Control Panel.
A list of data sources appears.

2   If you are configuring a new data source, click **Add**.   A list of installed drivers appears.   Select
INTERSOLV Btrieve and click **OK**.

If you are configuring an existing data source, select the data source name and click **Setup**.

The setup dialog box appears.

3   Specify values as follows:

Data Source Name: A string that identifies this Btrieve data source configuration in ODBC.INI.
Examples include "Accounting" or "Btrieve Files."

Description: An optional long description of a data source name.   For example, "My Accounting
Database" or "Btrieve files in C:\ACCOUNTS."

Database Directory: The full pathname of the directory that contains the Btrieve files and the data
dictionary files (.DDF).   Data dictionary files describe the structure of Btrieve data.   If no directory is
specified, the current working directory is used.

The following settings are optional:

Default Logon ID: The default logon ID used to connect to your Btrieve database.   A logon ID is
required only if security is enabled on your database.   Your ODBC application may override this
value or you may override this value in a connection string.

File Open Cache: A numeric value to specify the maximum number of unused file opens to cache.
For example, the value 4 specifies that when a user opens and closes four tables, the tables are not
actually closed.   The driver keeps them open so that if another query uses one of these tables, the
driver does not have to perform another open, which is expensive.   The advantage of file open
caching is increased performance.   The disadvantage is that a user who specifies file locking on
open may get a locking conflict even though no one appears to have the file open.   The default is 0,
which means no file open caching.

Action for Undefined Tables: A setting to indicate whether the driver should prompt the user when
it encounters a table for which it has no structure information.   Set the Prompt for Definition radio
button to prompt the user; set the Return Error Condition radio button (the default) to return an error.

Version: The version of Btrieve you want to access.   Choose either the Btrieve 5.*x* radio button (the
default) or the Btrieve 6.*x* radio button.

International Sort: A setting to indicate the order in which records are retrieved when you issue a
Select statement with an Order By clause.   Select this check box to use the international sort order
as defined by your operating system.   International sort order is case-insensitive (*a* precedes *B*); the
sorting of accented characters is also affected (see your operating system documentation).   Leave
this box blank to use ASCII sort order.   ASCII sort order is case-sensitive, where uppercase letters
precede lowercase letters (*B* precedes *a*).

Array Size: A value that enables the driver to retrieve an array of records from the Btrieve engine
and in most cases results in better performance for the application. The Array Size value is the
number of bytes in the array. The default value is 4096 bytes and the maximum is 65535 bytes.

4   Click **Define** to define table structure.   See <u>Defining Table Structure</u> for step-by-step instructions for
how to define table structure.

5   Click **Translate** to perform a translation of your data from one character set to another. The Select
Translator dialog box appears in which you can select a translator to peform the translation.
INTERSOLV provides a translator named INTERSOLV OEM ANSI that translates your data from the

IBM PC character set to the ANSI character set. Click **OK** to leave this dialog box and perform the translation.

The translators that are listed in this dialog box are determined by the values listed in the ODBC Translators section of your ODBCINST.INI file.

6   Click **OK** to write these values to ODBC.INI.   These values are now the defaults when you connect to the data source.   You can change the defaults by configuring your data source again.   You can override the defaults by connecting to the data source using a connection string with alternate values.

Defining Table Structure

Since Btrieve does not store any column information in the data file, you must define its structure through the Btrieve setup dialog box, which you can access through the ODBC Administrator.

To define the structure of a file, take the following steps:

1   Click **Define** in the setup dialog box.   The Define File dialog box appears.

2   Select the file you want to define and click **OK.**   The Define Table dialog box appears.

In this dialog box, Database Name and File display the name of the directory containing the Scalable SQL data dictionary files and the name of the file, respectively.

3   In the Table Information section of this dialog box, type the name of the table to be returned by SQLTables.   The name can be up to 20 characters and may not be the same as another defined table in the database.   This field is required.

4   The Column Information section of this dialog box lets you add, modify, and delete columns.   The box at the top of this section displays the defined columns.   To add a column definition, do the following:

a   In the Name box, type the name of the column.

b   In the Type box, open the drop-down list and select the column's data type.

c   In the Length box, type the column's length, if applicable.

d   In the Scale box, type the column's scale, if applicable.

e   Click **Add**.

To modify a column's definition, select the column you want to modify, type the new definition in the appropriate boxes (refer to steps 4a-d), and click **Modify**.

To remove a column's definition, select the column you want to remove and click **Remove**.

5   Click **OK** to define the table.

Connecting to Btrieve Using a Connection String

If your application requires a connection string to connect to a data source, you must specify the data source name that tells the driver which ODBC.INI section to use for the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in ODBC.INI.   These values are not written to ODBC.INI.

You can specify either long or short names in the connection string.   The connection string has the form:

DSN=*data_source_name*[;*attribute=value*[;*attribute=value*]...]

An example of a connection string for Btrieve is

DSN=BTRIEVE FILES;DB=EMP;UID=JOHN;PWD=XYZZY

The following table lists the long and short names for each attribute, as well as a description.

The defaults listed in the table are initial defaults that apply when no value is specified in either the connection string or in the data source definition in ODBC.INI.   If you specified a value for the attribute when configuring the data source, that value is your default.

Attribute	Description
DataSourceName (DSN)	String that identifies a Btrieve data source configuration in ODBC.INI. Examples include "Accounting" or "Btrieve Files."
Database (DB)	Directory in which the Btrieve files and the data dictionary files (.DDF) are stored. Data dictionary files describe the structure of Btrieve data. These files must exist to connect with the driver.
UndefinedTable (UT)	UndefinedTable={PROMPT | ERROR} Determines whether the driver should prompt the user when it encounters a table for which it has no structure information. Set this option to PROMPT to prompt the user; set it to ERROR to return an error. The initial default is to return an error.
FileOpenCache (FOC)	Determines the maximum number of unused file opens to cache. For example, when FileOpenCache=4, and a user opens and closes 4 tables, the tables are not actually closed. The driver keeps them open so that if another query uses one of these tables, the driver does not have to perform another open, which is expensive. The advantage of using file open caching is increased performance. The disadvantage is that a user who specifies file locking on open may get a locking conflict even though no one appears to have the file open. The initial default is 0, which means no file open caching.
LogonID (UID)	Default logon ID used to connect to your Btrieve database. A logon ID is required only if security is enabled on your database. If so, contact your system administrator to get your logon ID.
Password (PWD)	Password that you must enter if your Scalable

	SQL data dictionary files have security restrictions imposed.
Version (VER)	Version={5 | 6} The version of Btrieve you want to access. Valid values are 5 and 6. The initial default is 5.
IntlSort (IS)	IntlSort={0 | 1} Determines the order that records are retrieved when you issue a Select statement with an Order By clause. If IntlSort=1, the driver uses the international sort order as defined by your operating system. The sort is case-insensitive (*a* precedes *B*); the sorting of accented characters is also affected (see your operating system documentation). If IntlSort=0 (the initial default), the driver uses the ASCII sort order, where uppercase letters precede lowercase letters (*B* precedes *a*).
ModifySQL (MS)	ModifySQL={0 | 1}. Provided for backward compatibility. Specify ModifySQL=0 to have the driver understand SQL dialects found in earlier drivers. Specify ModifySQL=1 (the default) to have the driver modify the SQL statement to conform to ODBC specifications.
DeferQueryEvaluation (DQ)	DeferQueryEvaluation={0 | 1}. Determines when a query is evaluated--after all records are read or each time a record is fetched.
	If DeferQueryEvaluation=0, the driver generates a result set when the first record is fetched. The driver reads all records, evaluates each one against the Where clause, and compiles a result set containing the records that satisfy the search criteria. This process slows performance when the first record is fetched, but activity performed on the result set after this point is much faster because the result set has already been created. You do not see any additions, deletions, or changes in the database that occur while working with this result set.
	If DeferQueryEvaluation=1 (the default), the driver evaluates the query each time another record is fetched and stops reading through the records when it finds one that matches the search criteria. This setting avoids the slowdown while fetching the first record, but each fetch takes longer because of the evaluation taking place. The data you retrieve reflects the latest changes to the database. However, a result set is still generated if the query is a Union of multiple Select statements, if it contains the Distinct keyword, or if it has an Order By or Group By clause.
ArraySize (AS)	Enables the driver to retrieve an array of

records from the Btrieve engine and in most cases results in better performance for the application. The value of ArraySize is the number of bytes in the array. The default ArraySize is 4,096 bytes and the maximum is 65,535 bytes.

Data Types

The following table shows how the Btrieve data types map to the standard ODBC data types.   The Btrieve data types are used when you incorporate Btrieve files into a Scalable SQL database.

Btrieve	ODBC
Autoincrement(2)	SQL_SMALLINT
Autoincrement(4)	SQL_INTEGER
Bfloat(4)	SQL_REAL
Bfloat(8)	SQL_DOUBLE
Char	SQL_CHAR
Date	SQL_DATE
Decimal	SQL_DECIMAL
Float(4)	SQL_REAL
Float(8)	SQL_DOUBLE
Integer(1)	SQL_TINYINT
Integer(2)	SQL_SMALLINT
Integer(4)	SQL_INTEGER
Logical(1)	SQL_BIT
Logical(2)	SQL_BIT
Lstring	SQL_VARCHAR
Money	SQL_DECIMAL
Note	SQL_LONGVARCHAR
Numeric	SQL_NUMERIC
Numericsts	SQL_NUMERIC
Time	SQL_TIME
Zstring	SQL_VARCHAR

Indexes

Note If you define an index using the INTERSOLV Btrieve driver, the index will not have the restrictions discussed here.For query optimization, the Btrieve driver does not use:

- indexes containing all-segment-null keys or any-segment-null keys.
- any index key that is marked case-insensitive.
- any index keys where the data type of the index key does not match the data type of the field.

The one exception is if the index key is declared as an unsigned integer and the field in the file is declared as signed integer, or vice versa, then the driver assumes the field contains only unsigned quantities and uses the index. Note that this can lead to incorrect results if the field in fact does contain signed quantities.

The Btrieve driver only uses an alternate-collating-sequence (ASC) index key for equality lookups. Additionally, if an ASC key is part of a segmented index, the other index segments are not used for query optimization unless the WHERE clause contains an equality condition for the ASC key.

Column Names

Column names in SQL statements can be up to 20 characters long.   If column names are in all lowercase, a combination of upper and lowercase, contain blank spaces, or are reserved words, they must be surrounded by the grave character (`) (ASCII 96).   For example,

SELECT `name` FROM emp

Select Statement

You use the SQL Select statement to specify the columns and records to be read.   Btrieve Select statements support all the [Select statement](#) clauses.   This topic describes the information that is specific to Btrieve, which is ROWID.

ROWID Psuedo-Column

Each Btrieve record contains a special column named ROWID. This field contains a unique number that indicates the record's sequence in the database. You can use the ROWID in Where and Select clauses.

ROWID is particularly useful when you are updating records. You can retreive the ROWID of the records in the database along with the other field values. For example,

SELECT last_name, first_name, salary, rowid FROM emp

Then you can use the ROWID of the record that tyou want to update to ensure that you are updating the correct record and no other. For example

UPDATE emp set salary = 40000 FROM emp WHERE rowid=21

The fastest way of updating a single row is to use a Where clause with the ROWID. You cannot update the ROWID column.

Select statements that use the ROWID pseudo-column in the Where clause achieve maximum performance only for exact equality matches. If you use range scans instead of exact equality matches, a full table scan is performed. For example,

SELECT * FROM emp WHERE rowid=21 //fast search
SELECT * FROM emp WHERE rowid <=25 /full table search

Create and Drop Index Statements

The Btrieve driver supports SQL statements to create and delete indexes. The Create Index statement is used to create indexes and the Drop Index is used to delete indexes.

Create Index

The Create Index statement for Btrieve files has the form:

CREATE [UNIQUE] INDEX *index_name* ON *table_name*
([*field_name* [ASC | DESC] [,*field_name* [ASC | DESC]]...)

UNIQUE means that Btrieve does not let you insert two records with the same index values.

index_name is the name of the index.

table_name is the name of the table on which the index is to be created.

ASC tells Btrieve to create the index in ascending order.   DESC tells Btrieve to create the index in descending order.   By default, indexes are created in ascending order.

For example,

CREATE INDEX lname ON emp (last_name)

Drop Index

The form of the Drop Index statement is

DROP INDEX *table_name.index_name*

table_name is the name of the table from which the index is to be dropped.

index_name is the name of the index.

For example,

DROP INDEX emp.lname

Isolation and Lock Levels Supported

Btrieve supports isolation level 1 (read committed) only.   Btrieve supports page-level locking.

ODBC Conformance Level

The Btrieve driver supports the Core, Level 1, and Level 2 API functions listed in Supported ODBC Functions.   The Btrieve driver also supports backward and random fetching in SQLExtendedFetch.

The driver supports the minimum SQL grammar.

Number of Connections and Statements Supported

Btrieve files support a single connection and multiple statements per connection.

dBASE Driver

The dBASE driver supports
- dBASE II, dBASE III, dBASE IV and dBASE V files
- Clipper files
- FoxPro and FoxBASE tables and indexes

The dBASE driver executes the SQL statements directly on dBASE-compatible files.   You do not need to own the dBASE product to access these files.

The driver filename is QEDBF*nn*.DLL.

Configuring Data Sources

Defining Index Attributes

Connecting to dBASE Using a Connection String

Data Types

Column Names

Select Statement

Create Index Statement

Drop Index Statement

Pack Statement

Locking

Isolation and Lock Levels Supported

ODBC Conformance Level

Number of Connections and Statements Supported

Configuring Data Sources

To configure a dBASE data source, do the following:

1 Start the ODBC Administrator.   A list of data sources appears.

2 If you are configuring a new data source, click **Add**.   A list of installed drivers appears.   Select INTERSOLV dBASEFile and click **OK**.

If you are configuring an existing data source, select the data source name and click **Setup**.

The setup dialog box appears.

3 Specify values as follows:

Data Source Name: A string that identifies this dBASE data source configuration in ODBC.INI. Examples include "Accounting" or "dBASE Files."

Description: An optional long description of a data source name.   For example, "My Accounting Database" or "dBASE files in C:\ACCOUNTS."

Database Directory: A path specification to the directory that contains the database files.   If none is specified, the current working directory is used.

The following values are optional:

Create Type: The type of table or index to be created on a Create Table or Create Index statement. Select dBASE II, dBASE III, dBASE IV, dBASE V, Clipper, FoxBASE, FoxPro1, or FoxPro25.   The default is dBASE V.

Locking: The level of locking for the database file (FILE, RECORD, or NONE).   FILE locks all of the records in the table.   RECORD (the default) locks only the records affected by the statement. NONE offers the best performance but is intended only for single-user environments.

Lock Compatibility: The locking scheme the driver uses when locking records.   Select dBASE, Q+E, Q+EVirtual, Clipper, or Fox.   The default is dBASE.   These values determine locking support as follows:

- dBASE specifies Borland-compatible locking.
- Q+E specifies that locks be placed on the actual bytes occupied by the record.   Only applications that use the dBASE driver can read and write to the database.   Other applications are locked out of the table completely (they cannot even read other records).   This locking is compatible with earlier versions of Q+E products.
- Q+EVirtual specifies that locks be placed on bytes beyond the physical end-of-file.   Q+EVirtual is the same as Q+E except that other applications can open the table and read the data.

    The advantage of using a Q+E locking scheme over dBASE locking is that, on Inserts and Updates, Q+E locks only individual index tags, while dBASE locks the entire index.
- Clipper specifies Clipper-compatible locking.
- Fox specifies FoxPro- and FoxBASE-compatible locking.

If you are accessing a table with an application that uses the dBASE driver, your locking scheme does not have to match the Create Type.   However, if you are accessing a table with two applications, and only one uses the dBASE driver, set your locking scheme to match the other application.   For example, you do not have to set this value to Fox to work with a FoxPro table.   But if you are using a FoxPro application simultaneously with an application using the dBASE driver on the same set of tables, set this value to Fox to ensure that your data does not get corrupted.

File Open Cache: An integer value to specify the maximum number of unused file opens to cache. For example, the value 4 specifies that when a user opens and closes four tables, the tables are not actually closed.   The driver keeps them open so that if another query uses one of these tables, the driver does not have to perform another open, which is expensive.   The advantage of file open caching is increased performance.   The disadvantage is that a user who specifies file locking on open may get a locking conflict even though no one appears to have the file open.   The default is 0,

which means no file open caching.

Cache Size: The number of 64K blocks the driver uses to cache database records.   The higher the number, the better the performance.   The maximum number of blocks you can set depends on the system memory available.   If the cache size is greater than 0, when browsing backwards, you will not be able to see updates made by other users until you reexecute the Select statement. The default is 4.

International Sort: A setting to indicate the order in which records are retrieved when you issue a Select statement with an Order By clause.   Select this check box to use the international sort order as defined by your operating system.   International sort order is case-insensitive (*a* precedes *B*); the sorting of accented characters is also affected (see your operating system documentation).   Leave this box blank to use the ASCII sort order.   ASCII sort order is case-sensitive, where uppercase letters precede lowercase letters (*B* precedes *a*).

4   Click **Define** to define the index attributes for your data files.   See Defining Index Attributes for step-by-step instructions.

5   Click **Translate** to perform a translation of your data from one character set to another. The Select Translator dialog box appears in which you can select a translator to peform the translation. INTERSOLV provides a translator named INTERSOLV OEM ANSI that translates your data from the IBM PC character set to the ANSI character set. Click **OK** to leave this dialog box and perform the translation.

The translators that are listed in this dialog box are determined by the values listed in the ODBC Translators section of your ODBCINST.INI file.

6   Click **OK** to write these values to ODBC.INI.   These values are now the defaults when you connect to the data source.   You can change the defaults by configuring your data source again.   You can override the defaults by connecting to the data source using a connection string with alternate values.

Defining Index Attributes

Since dBASE lets you create index files that have different names than their corresponding data files, you must tell the driver what index files are associated with the dBASE file.   The driver updates the indexes for you, which ensures that they match the records in the dBASE file.   The driver also makes indexes available for optimizing queries.   It is not necessary to mark production .MDX files or structured .CDX files as maintained, as the driver maintains them for you.   However, you may wish to use this method to mark a tag as unique.

To define the index files that are associated with a dBASE file, take the following steps:

1   Click **Define** in the dBASE setup dialog box, which you can access through the ODBC Administrator. The Define File dialog box appears.

2   Select a dBASE file and click **OK** to define the special indexes using the Define Table dialog box.

The upper section of the dialog box displays the directory name and filename that contains the data file.

The lower section of the dialog box displays the index information for the data file.   The Index File drop-down list lets you select any index file in the database directory.   If the index file is in a different directory, you must provide the full pathname.

3   Select the Maintain check box to associate this index file with your dBASE file.

4   To specify that an index file is unique, select the Unique check box that appears at the right of the index filename.

5   If the selected index has an .MDX or .CDX extension, you cannot mark the index file as unique. Instead, you may mark the tags within the index as unique.   To do so, select the tag name in the Tag drop-down list and select the Unique check box that appears at the right of the tag name.

6   Click **OK** to save this information, or press Cancel.

Connecting to dBASE Using a Connection String

If your application requires a connection string to connect to a data source, you must specify the data source name that tells the driver which ODBC.INI section to use for the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in ODBC.INI.   These values are not written to the ODBC.INI file.

You can specify either long or short names in the connection string.   The connection string has the form:

DSN=*data_source_name*[;*attribute=value*[;*attribute=value*]...]

An example of a connection string for dBASE is

DSN=DBASE FILES;LCK=NONE;IS=0

The following table gives the long and short names for each attribute, as well as a description.

The defaults listed in the table are initial defaults that apply when no value is specified in either the connection string or in the data source definition in ODBC.INI.   If you specified a value for the attribute when configuring the data source, that value is your default.

Attribute	Description
DataSourceName (DSN)	String that identifies a dBASE data source configuration in ODBC.INI. Examples include "Accounting" or "dBASE Files."
Database (DB)	Directory in which the dBASE files are stored.
CreateType (CT)	CreateType={dBASE2 | dBASE3 | dBASE4 | dBASE5 | Clipper | FoxBASE | FoxPro1 | FoxPro25}. Type of table or index to be created on a Create Table or Create Index statement. dBASE5 is the initial default.
LockCompatibility (LCOMP)	LockCompatibility={Q+E | Q+EVirtual | dBASE | Clipper | Fox}. Locking schemes to be used in your dBASE tables.
	▪ LockCompatibility=Q+E specifies that locks be placed on the actual bytes occupied by the record. Only applications that use the dBASE driver can read and write to the database. Other applications are locked out of the table completely (they cannot even read other records). This locking is compatible with earlier versions of Q+E products.
	▪ LockCompatibility=Q+EVirtual specifies that locks be placed on bytes beyond the physical end-of-file. Q+EVirtual is the same as Q+E except that other applications can open the table and read the data.
	The advantage of using a Q+E locking scheme over dBASE locking is that, on Inserts and Updates, Q+E locks only individual index tags, while dBASE locks the entire index.
	▪ LockCompatibility=dBASE specifies Borland-compatible locking. This is the initial default.
	▪ LockCompatibility=Clipper specifies Clipper-compatible locking.
	▪ LockCompatibility=Fox specifies FoxPro- and FoxBASE-compatible locking.
	If you are accessing a table with an application that uses the dBASE driver, your locking scheme does

not have to match the Create Type.   However, if you are accessing a table with two applications, and only one uses the dBASE driver, set your locking scheme to match the other application.   For example, you don't have to set LCOMP=Fox to work with a FoxPro table.   But if you are using a FoxPro application simultaneously with an application using the dBASE driver on the same set of tables, set LCOMP=Fox to ensure that your data does not get corrupted.

Locking (LCK)	Locking={NONE | RECORD | FILE} Level of locking for the database tables.
	Locking=NONE offers the best performance but is intended only for single-user environments..
	Locking=RECORD locks only the records affected by the statement. This is the initial default.
	Locking=FILE locks all of the records in the table.
FileOpenCache (FOC)	Maximum number of unused file opens to cache. For example, when FileOpenCache=4, and a user opens and closes four files, the files are not actually closed. The driver keeps them open so that if another query uses one of these files, the driver does not have to perform another open, which is expensive. The advantage of using file open caching is increased performance. The disadvantage is that a user who tries to open the file exclusively may get a locking conflict even though no one appears to have the file open. The initial default is 0.
CacheSize (CSZ)	The number of 64K blocks the driver uses to cache database records. The higher the number, the better the performance. The maximum number of blocks you can set depends on the system memory available. If the cache size is greater than 0, when browsing backwards, you will not be able to see updates made by other users until you reexecute the Select statement. The default is 4.
IntlSort (IS)	IntlSort={0 | 1}. A number that specifies the order that records are retrieved when you issue a Select statement with an Order By clause. If IntlSort=1, the driver uses the international sort order as defined by your operating system. The sort is case-insensitive (*a* precedes *B*); the sorting of accented characters is also affected (see your operating system documentation). If IntlSort=0 (the initial default), the driver uses the ASCII sort order, where uppercase letters precede lowercase letters (*B* precedes *a*).
ModifySQL (MS)	ModifySQL={0 | 1}. This attribute is provided for backward compatibility with earlier versions of Q+E products. It determines whether the driver modifies SQL statements to conform to ODBC specifications or passes the SQL statement directly to dBASE.

Specify ModifySQL=1 to have the driver modify the SQL statement to conform to ODBC specifications. Specify ModifySQL=0 to have the driver understand SQL dialects found in earlier drivers.   The default is 1.

Compatibility (COMP)

Compatibility={0 | 1}. This attribute is provided for backward compatibility with earlier versions of Q+E products.   Use Compatibility=DBASE for backward compatibility; use Compatibility=ANSI for portability. The default is ANSI.

UltraSafeCommit (USF)

UltraSafeCommit={0 | 1}. A number that specifies when the driver flushes the file cache.   If UltraSafeCommit=1, the driver updates directory entries after each Commit.   This decreases performance.   If UltraSafeCommit=0 (the default) the driver updates the directory entry when the file is closed.   In this case, a machine "crash" before closing the file causes newly inserted records to be lost.

Data Types

The following table shows how dBASE data types map to the standard ODBC data types.   These dBASE data types can be used in a <u>Create Table statement</u>.

Note: A few products can create dBASE files with numbers that do not conform to the precision and scale of the Number column.   For example, these products can store 100000 in a column declared as NUMBER(5,2).   When this occurs, the dBASE driver displays error 1244, "Unsupported decimal format." To remedy this situation, multiply the nonconforming column by 1, which converts it to the Float data type. For example,

SELECT BADCOL * 1 FROM BADFILE

BADCOL * 1 is evaluated as an expression and is returned as a float value.

dBASE	ODBC
Binary[1]	SQL_LONGVARBINARY
Char[2]	SQL_CHAR
Date[3]	SQL_DATE
Float[4]	SQL_DECIMAL
General[5]	SQL_LONGVARBINARY
Logical	SQL_BIT
Memo[3]	SQL_LONGVARCHAR
Numeric	SQL_DECIMAL
Picture[6]	SQL_LONGVARBINARY

[1] dBASE V only

[2] 254 characters maximum (1024 for Clipper)

[3] dBASE III, IV, FoxPro, FoxBASE, and Clipper

[4] dBASE IV only

[5] FoxPro 2.5 and dBASE V only

[6] FoxPro, FoxBASE, and Clipper

Column Names

Column names in SQL statements (such as Select, Insert, and so on) can be up to 10 characters long.   A column name can contain alphanumeric characters and the hyphen character (-).   The first character must be a letter (a through z).

Select Statement

You use a SQL Select statement to specify the columns and records to be read.   dBASE Select statements support all of the <u>Select statement</u> clauses.   This topic describes the information that is specific to dBASE, which is ROWID.

ROWID Pseudo-Column

Each dBASE record contains a special column named ROWID.   This field contains a unique number that indicates the record's sequence in the database.   For example, a table that contains 50 records has ROWID values from 1 to 50 (if no records are marked deleted).   You can use ROWID in Where and Select clauses.

ROWID is particularly useful when you are updating records.   You can retrieve the ROWID of the records in the database along with the other field values.   For example,

SELECT last_name, first_name, salary, rowid FROM emp

Then you can use the ROWID of the record that you want to update to ensure that you are updating the correct record and no other.   For example,

UPDATE emp set salary = 40000 FROM emp WHERE rowid=21

The fastest way of updating a single row is to use a Where clause with the ROWID.   You cannot update the ROWID column.

Create Index Statement

The type of index you create is determined by the value of the CreateType attribute, which you set in the setup dialog box or as a connection string option.   The index can be
- dBASE II or III (.NDX)
- dBASE IV (.MDX)
- Clipper (.NTX)
- FoxBASE (.IDX)
- FoxPro (.CDX)

The syntax for creating an index is

CREATE [UNIQUE] INDEX *index_name* ON *base_table_name*
(*field_name* [ASC | DESC] [,*field_name* [ASC | DESC]] ...)

index_name is the name of the index file.   For FoxPro 2.5 and dBASE IV, this is a tag, which is required to identify the indexes in an index file.   Each index for a table must have a unique name.

UNIQUE means that the driver creates an ANSI-style unique index over the column and ensures uniqueness of the keys.   Use of unique indexes improves performance.   ANSI-style unique indexes are different from dBASE-style unique indexes.   With ANSI-style unique indexes, you receive an error message when you try to insert a duplicate value into an indexed field.   With dBASE-style unique indexes, you do not see an error message when you insert a duplicate value into an indexed field.   This is because only one key is inserted in the index file.

base_table_name is the name of the database file whose index is to be created.   The .DBF extension is not required, the driver automatically adds it if it is not present.   By default, dBASE IV index files are named *base_table_name*.MDX and FoxPro 2.5 indexes are named *base_table_name*.CDX.

field_name is a name of a column in the dBASE table.   You can substitute a valid dBASE-style index expression for the list of field names.

ASC tells dBASE to create the index in ascending order.   DESC tells dBASE to create the index in descending order.   By default, indexes are created in ascending order.   You cannot specify both ASC and DESC orders within a single Create Index statement.   For example, the following statement is invalid:

CREATE INDEX emp_i ON emp (last_name ASC, emp_id DESC)

The following table shows the attributes of the different index files supported by the dBASE driver.   For each type supported, it provides the following details:
- Whether dBASE-style unique indexes are supported
- Whether descending order is supported
- The maximum size supported for key columns
- The maximum size supported for the column specification in the Create Index statement
- Whether production/structural indexes are supported

Create Type/Ext.	dBASE UNIQUE	DESC	Max Size of Key Column	Max Size of Column Spec.	Production/ Structural Indexes
dBASE II .NDX	No	No	100	99	No
dBASE III .NDX	Yes	No	100	219	No
Clipper .NTX	Yes	Yes	250	255	No
FoxBASE .IDX*	Yes	No	100	219	No
dBASE IV	Yes	Yes	100	220	Yes

and
V .MDX

FoxPro 2.5 .IDX**	Yes	Yes	240	255	No
FoxPro 2.5 .CDX	Yes	Yes	240	255	Yes

* These IDX indexes are also created as the default for FoxPro 1.0.

**Compact IDX indexes have the same internal structure as a tag in a CDX file.   These indexes can be
   created if the IDX extension is included with the index name in the Create Index statement.

Drop Index Statement

The syntax for dropping a dBASE index is as follows:

DROP INDEX table_name.index_name

table_name is the name of the dBASE file without the extension.

For FoxPro 2.5 and dBASE IV, *index_name* is the tag.   Otherwise, *index_name* is the name of the index file without the extension.

To drop the index EMPHIRE.NDX, issue the following statement:

DROP INDEX emp.emphire

Pack Statement

When records are deleted from a dBASE file, they are not removed from the file.   Instead, they are marked as having been deleted.   Also, when memo fields are updated, space may be wasted in the files. To remove the deleted records and free the unused space from updated memo fields, you must use the Pack statement.   It has the following form:

PACK filename

filename is the name of the dBASE file to be packed.   The .DBF extension is not required; the driver automatically adds the extension if it is not present.   For example,

PACK emp

You cannot pack a file that is opened by another user, and you cannot use the Pack statement in manual commit mode.

For the specified file, the Pack statement does the following:
- Removes all deleted records from the file
- Removes the entries for all deleted records from .CDX and .MDX files having the same name as the file
- Compresses unused space in the memo (.DBT or .FPT) file

Locking

With the dBASE driver, you can build and run applications that share dBASE database files on a network. Whenever more than one user is running an application that accesses a shared database file, the applications should lock the records that are being changed.   Locking a record prevents other users from locking, updating, or deleting the record.

Levels of Database Locking

The dBASE driver supports three levels of database locking: NONE, RECORD, and FILE.   You can set these levels in
- The connection string (LCK=)
- The setup dialog box

No locking offers the best performance but is intended only for single-user environments.

With record or file locking, the system locks the database tables during Insert, Update, Delete, or Select...For Update statements.   The locks are released when the user commits the transaction.   The locks prevent other users from modifying the locked objects, but they do not lock out readers.

With record locking, only records affected by the statement are locked.   Record locking provides better concurrency with other users who also want to modify the table.

With file locking, all the records in the table are locked.   File locking has lower overhead and may work better if records are modified infrequently, if records are modified primarily by one user, or if a large number of records are modified.

Using Locks on Local Files

If you use database locking and are accessing files locally (not on a network), run the DOS utility SHARE.EXE before running Windows.   If you add SHARE.EXE to your AUTOEXEC.BAT file, it runs automatically each time you boot your computer.

Limit on Number of Locks

There is a limit on the number of locks that can be placed on a file.   If you are accessing a dBASE file from a server, the limit depends on the server (see your server documentation).   If you are accessing a dBASE file locally, the limit depends on the buffer space allocated when SHARE.EXE was loaded (see your DOS documentation).   If you are exceeding the number of locks available, you may want to switch to file locking.

How Transactions Affect Record Locks

When an Update or Delete statement is executed, the driver locks the records affected by that statement. The locks are released after the driver commits the changes.   Under manual commit mode, the locks are held until the application commits the transaction.   Under autocommit mode, the locks are held until the statement is executed.

When a Select...For Update statement is executed, the driver locks a record only when the record is fetched.   If the record is updated, the driver holds the lock until the changes are committed.   Otherwise, the lock is released when the next record is fetched.

Isolation and Lock Levels Supported

dBASE supports isolation level 1.   It supports both file- and record-level locking.

ODBC Conformance Level

The dBASE driver supports the Core, Level 1, and Level 2 API functions listed in [Supported ODBC Functions](#).   The dBASE driver also supports backward and random fetching in SQLExtendedFetch.

The driver supports the minimum SQL grammar.

Number of Connections and Statements Supported

dBASE supports multiple connections and multiple statements per connection.

Excel Drivers

There are two drivers that support Excel .XLS files.   One supports Excel versions 2, 3, and 4 and is called the Excel 4 driver.   The other supports Excel version 5 and is called the Excel 5 driver.

The Excel 4 driver filename is QEXLS*nn*.DLL.   The Excel 5 driver filename is QEXLS5*nn*.DLL.

System Requirements

Configuring Data Sources

Using Excel Databases

Connecting to an Excel Database Using a Connection String

Select Statement

Create Table Statement

Data Types

ODBC Conformance Level

Number of Connections and Statements Supported

System Requirements

There are no software requirements for using the Excel 4 driver, because you do not need the Excel program to use it.   It works wherever your Excel version 2, 3, or 4 .XLS files are present.

To use the Excel 5 driver, you must have version 5 of the Excel program installed on the system where you intend to use an Excel 5 database.   There are two main reasons for this restriction:
▪   The Excel 5 driver requires two DLLs to operate.   These DLLs, STORAGE.DLL and COMPOBJ.DLL, are installed when you install Excel version 5.
▪   Because the Excel 5 driver is able only to read data from Excel 5 .XLS files, you must use the Excel 5 program for any manipulation of your Excel 5 databases.

Using Excel Databases

The way that you create and use Excel databases differs according to which version of Excel and corresponding Excel driver you use.   The different methods are described in the following sections.

Excel 4 Databases

The Excel 4 driver lets you create databases using Excel version 2, 3, or 4.   An Excel 4 database is a directory of component .XLS worksheet files, each of which contains a single table.

With the Excel 4 driver, you can open any Excel worksheet file that contains a database section.   To create a database section in a worksheet, you select a range to be used as a database and choose Excel's Set Database command from the Data menu.   The driver accesses the rows and columns from the database section.

The Excel 4 driver allows some database manipulation via SQL statements.   The Insert statement is limited to use with the Create Table statement only.   The Drop Table statement is also supported.   You may not execute Update or Delete statements.

Excel 5 Databases

An Excel 5 database is any Excel 5 .XLS workbook file that contains one or more named lists containing record data.   Each named list is treated as a table within the workbook database.   The lists must be set up as follows:
- The first row in each list must contain the column labels that identify the fields in each record.
- The name of each list must be a book-level name, not a sheet-level name.
- Although the Excel 5 driver recognizes one list named "Database" per .XLS file, it is recommended that you avoid using this name--both to ensure that the Excel 5 driver recognizes other lists in the file and to prevent future naming conflicts.

See your Excel 5 documentation for more information on creating and naming lists in Excel.

Configuring Data Sources

To configure an Excel data source, do the following:

1 Start the ODBC Administrator.   A list of data sources appears.

2 If you are configuring a new data source, click **Add**.   A list of installed drivers appears.   Select INTERSOLV ExcelFile (or INTERSOLV Excel5Workbook for the Excel 5 driver) and click **OK**.

If you are configuring an existing data source, select the data source name and click **Setup**.

The Setup dialog box appears.

3 Specify values as follows:

Data Source Name: A string that identifies this Excel data source configuration in ODBC.INI. Examples include "Accounting" or "Excel Database."

Description: An optional long description of a data source name.   For example, "My Accounting Database" or "Excel .XLS file data."

Database Directory *or* Database Workbook: For the Excel 4 driver, this identifies the directory in which the Excel files are stored.   If none is specified, the current working directory is used.   For the Excel 5 driver, this field identifies the workbook file containing the Excel database.

The following values are optional:

File Open Cache: A numeric value to specify the maximum number of unused file opens to cache. For example, the value 4 specifies that when a user opens and closes four tables, the tables are not actually closed.   The driver keeps them open so that if another query uses one of these tables, the driver does not have to perform another open, which is expensive.   The advantage of file open caching is increased performance.   The default is 0, which means no file open caching.

Cache Size: The number of 64K blocks the driver uses to cache database records.   The higher the number, the better the performance.   The maximum number of blocks you can set depends on the system memory available.   If the cache size is greater than 0, when browsing backwards, you will not be able to see updates made by other users until you reexecute the Select statement. The default is 4.

Rows to Scan: The number of rows to scan to determine the column data types.   The default is 25 rows.   A value of 0 means to scan the whole table.

International Sort: A setting to indicate the order in which records are retrieved when you issue a Select statement with an Order By clause.   Select this check box to use the international sort order as defined by your operating system.   International sort order is case-insensitive (*a* precedes *B*); the sorting of accented characters is also affected (see your operating system documentation).   Do not this box to use the ASCII sort order.   ASCII sort order is case-sensitive, where uppercase letters precede lowercase letters (*B* precedes *a*).

Ultra Safe Commit: Excel 4 attribute driver only. A number that specifies when the driver flushes the file cache. If UltraSafeCommit=1, the driver updates directory entries after each Commit. This decreases performance. If UltraSafeCommit=0 (the default) the driver updates the directory entry when the file is closed. In this case, a machine "crash" before closing the file causes newly inserted records to be lost.

Character Length Guess: A value of 0 or 1 that determines whether the driver tries to guess the length of character columns. If set to 0 (the default), the driver does not try to guess the length of character columns. Instead, it assumes that all character columns have a length of 255. If set to 1, the driver tries to guess the length.

4 Click **Translate** to perform a translation of your data from one character set to another. The Select Translator dialog box appears in which you can select a translator to peform the translation. INTERSOLV provides a translator named INTERSOLV OEM ANSI that translates your data from the

IBM PC character set to the ANSI character set. Click **OK** to leave this dialog box and perform the translation.

The translators that are listed in this dialog box are determined by the values listed in the ODBC Translators section of your ODBCINST.INI file.

5   Click **OK** to write these values to ODBC.INI.   These values are now the defaults when you connect to the data source.   You can change the default values by configuring your data source again.   You can override the defaults by connecting to the data source using a connection string with alternate values.

Connecting to an Excel Database Using a Connection String

If your application requires a connection string to connect to a data source, you must specify the data source name that tells the driver which ODBC.INI section to use for the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in ODBC.INI.   These values are not written to ODBC.INI.

You can specify either long or short names in the connection string.   The connection string has the form:

DSN=*data_source_name*[;*attribute=value*[;*attribute=value*]...]

An example of a connection string for Excel is

DSN=EXCEL FILES;FOC=4

The following table gives the long and short names for each attribute, as well as a description.

The defaults listed in the table are initial defaults that apply when no value is specified in either the connection string or in the data source definition in ODBC.INI.   If you specified a value for the attribute when configuring the data source, that value is your default.

Attribute	Description
DataSourceName (DSN)	String that identifies an Excel data source configuration in ODBC.INI. Examples include "Accounting" or "Excel Files."
Database (DB)	For the Excel 4 driver, the directory in which the Excel files are stored. For the Excel 5 driver, the name of the .XLS workbook file containing the Excel database.
ScanRows (SR)	Number of rows to scan to determine the column data types. A value of 0 means to scan the whole table. The initial default is 25.
FileOpenCache (FOC)	Maximum number of unused file opens to cache. For example, when FileOpenCache=4, and a user opens and closes four files, the files are not actually closed. The driver keeps them open so that if another query uses one of these files, the driver does not have to perform another open, which is expensive. The advantage of using file open caching is increased performance. The disadvantage is that a user who tries to open a file exclusively may get a locking conflict even though no one appears to have the file open. The initial default is 0.
CacheSize (CSZ)	Number of 64K blocks the driver uses to cache database records. The higher the number, the better the performance. The maximum number of blocks you can set depends on the system memory available. If the cache size is greater than 0, when browsing backwards, you will not be able to see updates made by other users until you reexecute the Select statement. The default is 4.
IntlSort (IS)	IntlSort={0 | 1}. Determines the order in which records are retrieved when you issue a Select statement with an Order By clause. If IntlSort=1, the driver uses the international sort order as defined by your operating system. The sort is

	case-insensitive (*a* precedes *B*); the sorting of accented characters is also affected (see your operating system documentation). If IntlSort=0 (the initial default), the driver uses the ASCII sort order, where uppercase letters precede lowercase letters (*B* precedes *a*).
FieldGuessing (FG)	FieldGuessing={0 | 1}. A value that determines whether the driver will try to guess column data types. If FieldGuessing=1 (the default), the driver attempts to guess the data types. If FieldGuessing=0, the driver assumes that all data types are SQL_CHAR.
ModifySQL (MS)	ModifySQL={0 | 1}. This attribute is provided for backward compatibility with earlier versions of Q+E products. It determines whether the driver modifies SQL statements to conform to ODBC specifications or passes the SQL statement directly to Excel. Specify ModifySQL=1 to have the driver modify the SQL statement to conform to ODBC specifications. Specify ModifySQL=0 to have the driver understand SQL dialects found in earlier drivers. The default is 1.
UltraSafeCommit (USF)	UltraSafeCommit={0 | 1}. Excel 4 driver attribute. A number that specifies when the driver flushes the file cache. If UltraSafeCommit=1, the driver updates directory entries after each Commit. This decreases performance. If UltraSafeCommit=0 (the default) the driver updates the directory entry when the file is closed. In this case, a machine "crash" before closing the file causes newly inserted records to be lost.
CharacterLength Guess (CLG)	CharacterLengthGuess={0 | 1}. Determines whether the driver tries to guess the length of character columns. If set to 0 (the default), the driver does not try to guess the length of character columns. Instead, it assumes that all character columns have a length of 255. If set to 1, the driver tries to guess the length.

Data Types

The following table shows how Excel data types map to the standard ODBC data types.   These Excel data types can be used in a Create Table statement with the Excel 4 driver.

Excel	ODBC
Char	SQL_VARCHAR
Date	SQL_DATE
Logical	SQL_BIT
Number Float	SQL_DOUBLE
Time	SQL_TIME
Timestamp	SQL_TIMESTAMP

Table and Column Names

Table names are case-sensitive when using the Excel 5 driver.   They are not case-sensitive when using the Excel 4 driver.

Excel files contain column names in the first row of the database section or named list that is used as a table.   Use these names as the column names in a Select statement.   Column names are limited to 32 characters.   If column names are lowercase, contain upper- and lowercase, contain blank spaces, or are reserved words, surround them with the grave character (`) (ASCII 96).

Select Statement

You use a SQL Select statement to specify the columns and records to be read.   Excel Select statements support all of the Select statement clauses.   This topic describes the information that is specific to Excel, which is table and column names.

For example,

SELECT `name` FROM emp

Create Table Statement

The Excel 5 driver does not support the Create Table statement.

The Excel 4 driver supports the Create Table statement. The Insert statement can be used to load the newly created table, but this must be done during the same connection in which the Create Table statement was issued.   Once the connection used to create the table is closed, the Insert statement no longer operates on that table. Future rows need to be inserted directly through Excel. The Excel driver creates tables in Microsoft Excel version 4.0 (BIFF3) format.

ODBC Conformance Level

The Excel drivers support the Core, Level 1, and Level 2 API functions listed in [Supported ODBC Functions](#).   The Excel drivers support backward and random fetching in SQLExtendedFetch.

The drivers support the minimum SQL grammar.

Number of Connections and Statements Supported

The Excel drivers support multiple connections and multiple statements per connection.

INFORMIX 4 Driver

The INFORMIX 4 driver supports version 4.1 of the INFORMIX database system.

The driver filename is QEINF*nn*.DLL.

[System Requirements](#)

[Configuring Data Sources](#)

[Connecting to INFORMIX Using a Logon Dialog Box](#)

[Connecting to INFORMIX Using a Connection String](#)

[Data Types](#)

[Isolation and Lock Levels Supported](#)

[ODBC Conformance Level](#)

[Number of Connections and Statements Supported](#)

System Requirements

INFORMIX 4 system requirements are as follows:
- You must have the Informix-Net or Informix-Star product from INFORMIX.
- The message files RDS.IEM and SQL.IEM must be located in the INFORMIX subdirectory MSG.
- The environment variable INFORMIXDIR must be set to the directory where you have installed the INFORMIX client.   For example:

        SET INFORMIXDIR=C:\INFORMIX

    You can include this line in your AUTOEXEC.BAT file.
- The Informix-Net package includes LDLLSQLW.DLL.   The path to this DLL must be in your PATH environment variable.

Configuring Data Sources

To configure an INFORMIX data source, do the following:

1   Start the ODBC Administrator.   A list of data sources appears.

2   If you are configuring a new data source, click **Add**.   A list of installed drivers appears.   Select INTERSOLV INFORMIX and click **OK**.

If you are configuring an existing data source, select the data source name and click **Setup**.

The setup dialog box appears.

3   Specify values as follows:

Data Source Name: A string that identifies this INFORMIX data source configuration in ODBC.INI. Examples include "Accounting" or "INFORMIX on Serv #1."

Description: An optional long description of a data source name.   For example, "My Accounting Database" or "INFORMIX tables on server number 1."

Database Name: The name of the database to which you want to connect by default.

Database List: The list of databases that will be available in the logon dialog box.   Separate the names with commas.

The following value is optional:

Cursor Behavior: Select Preserve if you want cursors to be held at the current position when the transaction ends.   Otherwise, leave this set to Close.   Selecting Preserve may impact the performance of your database operations.

4   Click **Translate** to perform a translation of your data from one character set to another. The Select Translator dialog box appears in which you can select a translator to peform the translation. INTERSOLV provides a translator named INTERSOLV OEM ANSI that translates your data from the IBM PC character set to the ANSI character set. Click **OK** to leave this dialog box and perform the translation.

The translators that are listed in this dialog box are determined by the values listed in the ODBC Translators section of your ODBCINST.INI file.

5   Click **OK** to write these values to ODBC.INI.   These values are now the defaults when you connect to the data source.   You can change the defaults by configuring your data source again.   You can override the defaults by connecting to the data source using a connection string with alternate values.

Connecting to INFORMIX Using a Logon Dialog Box

Some ODBC applications display a logon dialog box when you are connecting to a data source.   In these cases, the data source name has already been specified.   For INFORMIX, the dialog box is as follows:

In the logon dialog box, do the following:

1   Type the name of the database you want to access or select the name from the Database Name drop-down list, which displays the names you specified in the setup dialog box.

2   Click **OK** to complete the logon and to update this value in ODBC.INI.

Connecting to INFORMIX Using a Connection String

If your application requires a connection string to connect to a data source, you must specify the data source name that tells the driver which ODBC.INI section to use for the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in ODBC.INI.   These values are not written to ODBC.INI.

You can specify either long or short names in the connection string.   The connection string has the form:

DSN=*data_source_name*[;*attribute=value*[;*attribute=value*]...]

An example of a connection string for INFORMIX 4 is

DSN=INFORMIX on Serv #1;DB=PAYROLL

The following table gives the long and short names for each attribute, as well as a description.

The defaults listed in the table are initial defaults that apply when no value is specified in either the connection string or in the data source definition in ODBC.INI.   If you specified a value for the attribute when configuring the data source, that value is your default.

Attribute	Description
DataSourceName (DSN)	Identifies an INFORMIX data source configuration in ODBC.INI. Examples include "Accounting" or "INFORMIX on Serv #1."
Database (DB)	Name of the database to which you want to connect.
CursorBehavior (CB)	CursorBehavior={0 | 1}. Determines whether cursors are preserved or closed at the end of each transaction. Valid values are 1 (preserve) and 0 (close, the initial default). Set this attribute to 1 if you want cursors to be held at the current position when the transaction ends. Doing so may impact the performance of your database operations.

Data Types

INFORMIX 4 data types map to the standard ODBC data types as follows:

INFORMIX 4	ODBC
Byte*	SQL_LONGVARBINARY
Char	SQL_CHAR
Date	SQL_DATE
Datetime year to fraction(5)	SQL_TIMESTAMP
Decimal	SQL_DECIMAL
Float	SQL_DOUBLE
Integer	SQL_INTEGER
Money	SQL_DECIMAL
Serial	SQL_INTEGER
Smallfloat	SQL_REAL
Smallint	SQL_SMALLINT
Text*	SQL_LONGVARCHAR
Varchar*	SQL_VARCHAR

* Not supported for Standard Engine Databases.

Note: The Interval data type is not supported. Existing columns of this type are mapped to the ODBC SQL_CHAR data type.

Isolation and Lock Levels Supported

If connected to an Online Server, INFORMIX 4 supports isolation levels 0 (read uncommitted), 1 (read committed), and 3 (serializable).   The default is 1.

INFORMIX 4 also supports an alternative isolation level 1, called cursor stability.   Your ODBC application can use this isolation level by calling SQLSetConnectOption (1040,1).

Additionally, if transaction logging has not been enabled for your database,then transactions are not supported by the driver (the driver is always in auto-commit mode).

INFORMIX 4 supports page-level locking.

ODBC Conformance Level

The INFORMIX 4 driver supports the Core, Level 1, and Level 2 API functions listed in Supported ODBC Functions.   In addition, the following Level 2 functions are supported:
- SQLBrowseConnect
- SQLProcedures

The driver supports the minimum SQL grammar.

Number of Connections and Statements Supported

The INFORMIX 4 database system supports a single connection and multiple statements per connection.

INFORMIX 5 Driver

The INFORMIX 5 driver supports multiple connections to the INFORMIX 5 database.

The driver filename is QEINF5*nn*.DLL.

[System Requirements](#)

[Configuring Data Sources](#)

[Connecting to INFORMIX Using a Logon Dialog Box](#)

[Connecting to INFORMIX Using a Connection String](#)

[Data Types](#)

[Isolation and Lock Levels Supported](#)

[ODBC Conformance Level](#)

[Number of Connections and Statements Supported](#)

System Requirements

To access remote INFORMIX databases, you need version 5.x of the INFORMIX-Net or INFORMIX-Star product from INFORMIX.

The message files RDS.IEM and SQL.IEM must be located in the INFORMIX subdirectory MSG.

The environment variable INFORMIXDIR must be set to the directory where you have installed the INFORMIX client.   For example:

SET INFORMIXDIR=C:\INFORMIX

This line can be included in your AUTOEXEC.BAT file.

The INFORMIX-Net 5.0.1 package includes ISQLI501.DLL.   The path to this DLL must be in your PATH environment variable.   If it is not, the following message appears:

The setup routines for the INTERSOLV INFORMIX5 ODBC driver could not be loaded. You may be low on memory and need to quit a few applications.

Configuring Data Sources

To configure an INFORMIX 5 data source, do the following:

1   Start the ODBC Administrator.   A list of data sources appears.

2   If you are configuring a new data source, click **Add**.   A list of installed drivers appears.   Select INTERSOLV INFORMIX5 and click **OK**.

If you are configuring an existing data source, select the data source name and click **Setup**.

The setup dialog box appears.

3   Specify values as follows:

Data Source Name: A string that identifies this INFORMIX data source configuration in ODBC.INI. Examples include "Accounting" or "INFORMIX-Serv1."

Description: An optional long description of a data source name.   For example, "My Accounting Database" or "INFORMIX 5 files on Server number 1."

Database Name: The name of the database to which you want to connect by default.

The following values are optional:

Database List: The list of databases that will be displayed in the logon dialog box. If Get DB List From Informix is set to 1, the list of databases that will be displayed in the logon dialog box is created from the database list returned for INFORMIX at logon time.

When you specify databases, separate the names with commas.

Host Name: The name of the machine on which the INFORMIX 5 server resides.

Default Logon ID: The name of the user as specified on the INFORMIX 5 server.

Service Name: The name of the service as it appears on the host machine.   This service is assigned by the system administrator.   The name you specify is displayed in the INFORMIX Server Options dialog box.

Protocol Type: The protocol used to communicate with the server.   Specify one or more values; separate the names with commas.   Values can be FTP, IPX, LANMAN, TCP-IP.   This list is displayed in the Logon Options dialog box.

Yield Proc: A numeric value that determines whether you can work in other Windows applications when INFORMIX 5 is busy.   This attribute is useful to users of ODBC applications.   Valid values are:

- 0 (peek and dispatch), which causes the driver to check the Windows message queue and send any messages to the appropriate Windows application.
- 1 (no yielding, the default), which does not let you work in other applications.
- 2 (INFORMIX's yield procedure), which uses INFORMIX's default yield procedure.
- 3 (dispatch via Windows Yield function), which turns control over to the Windows kernel.   The Windows kernel checks the message queue and sends any messages to the appropriate application window.

It is recommended that you use the value 1.

Cursor Behavior: Select Preserve if you want cursors to be held at the current position when the transaction ends.   Otherwise, leave this set to Close.   Selecting Preserve may impact the performance of your database operations.

Enable Scrollable Cursors: A value that determines whether the driver can use scrollable cursors. The initial default value is 0 (no use of scrollable cursors). The INFORMIX driver can use scollable cursors only if there are no long columns (SQL_LONGVARCHAR or SQL_LONGVARBINARY) in a Select list. If you set this option to use scrollable cursors (EnableScrollableCursors=1), you must not include long columns in the Select list.

Get DB List from Informix: A value that determines whether the driver requests the database list to be returned from the INFORMIX server or from the database list that the user entered at driver setup.

4   Click **Translate** to perform a translation of your data from one character set to another. The Select Translator dialog box appears in which you can select a translator to peform the translation. INTERSOLV provides a translator named INTERSOLV OEM ANSI that translates your data from the IBM PC character set to the ANSI character set. Click **OK** to leave this dialog box and perform the translation.

The translators that are listed in this dialog box are determined by the values listed in the ODBC Translators section of your ODBCINST.INI file.

5   Click **OK** to write these values to ODBC.INI.   These values are now the defaults when you connect to the data source.   You can change the defaults by configuring your data source again.   You can override the defaults by connecting to the data source using a connection string with alternate values.

Connecting to INFORMIX Using a Logon Dialog Box

Some ODBC applications display a logon dialog box when you are connecting to a data source. In these cases, the data source name has already been specified.

In the logon dialog box, do the following:

1 Type the name of the database you want to access or select the name from the Database Name drop-down list. This list displays names returned from the INFORMIX server. Or, if you set the Get DB List from Informix option to 0 in the Setup dialog box, this list displays the names you specified.

2 Type the name of the server (host name) on which INFORMIX 5 resides.

3 If required, type your user name as specified on the INFORMIX 5 server.

4 If required, type your password.

5 Optionally, click **Options** to display the INFORMIX Server Options dialog box. This dialog box enables you to change the Service Name and Protocol Type that you specified in the setup dialog box. Click **OK** to accept any changes you make.

6 Click **OK** to complete the logon and to update these values in ODBC.INI.

Connecting to INFORMIX Using a Connection String

If your application requires a connection string to connect to a data source, you must specify the data source name that tells the driver which ODBC.INI section to use for the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in ODBC.INI.   These values are not written to ODBC.INI.

You can specify either long or short names in the connection string.   The connection string has the form:

DSN=*data_source_name*[;*attribute=value*[;*attribute=value*]...]

An example of a connection string for INFORMIX 5 is

DSN=INFORMIX TABLES;DB=PAYROLL

The following table gives the long and short names for each attribute, as well as a description.

The defaults listed in the table are initial defaults that apply when no value is specified in either the connection string or in the data source definition in ODBC.INI.   If you specified a value for the attribute when configuring the data source, that value is your default.

Attribute	Description
DataSourceName (DSN)	Identifies an INFORMIX data source configuration in ODBC.INI. Examples include "Accounting" or "INFORMIX-Serv1."
Database (DB)	Name of the database to which you want to connect.
HostName (HOST)	Name of the machine on which the INFORMIX 5 server resides.
LogonID (UID)	Your user name as specified on the INFORMIX 5 server.
Password (PWD)	A password.
Service (SERV)	Name of the service as it appears on the host machine. This service is assigned by the system administrator.
Protocol (PRO)	Protocol={FTP | IPX | LANMAN | TCP-IP}. Protocol used to communicate with the server. You can specify one or more values; separate the names with commas.
YieldProc (YLD)	YieldProc={0 | 1 | 2 | 3}. Determines if you can work in other Windows applications when INFORMIX 5 is busy. This attribute is useful to users of ODBC applications. ▪ YieldProc=0 (peek and dispatch) causes the driver to check the Windows message queue and send any messages to the appropriate Windows application. ▪ YieldProc=1 (no yielding, the initial default) does not let you work in other applications. ▪ YieldProc=2 (INFORMIX's yield procedure) uses INFORMIX's default yield procedure. ▪ YieldProc=3 (dispatch via Windows Yield function) turns control over to the Windows kernel. The Windows kernel checks the message queue and sends any messages to the appropriate application window. It is recommended that you use YieldProc=1. YieldProc=0, YieldProc=2, and YieldProc=3 do not work with all Windows applications.
CursorBehavior	CursorBehavior={0 | 1}. Determines whether cursors

(CB)	will be preserved or closed at the end of each transaction. Valid values are 1 (preserve) and 0 (close, the initial default). Set this attribute to 1 if you want cursors to be held at the current position when the transaction ends. The value CursorBehavior=1 may impact the performance of your database operations.
EnableScrollable Cursors (ESC)	EnableScrollable Cursors={0	1}. Determines whether the driver can use scrollable cursors. The initial default value is 0 (no use of scrollable cursors). The INFORMIX driver can use scollable cursors only if there are no long columns (SQL_LONGVARCHAR or SQL_LONGVARBINARY) in a Select list. If you set this option to use scrollable cursors (EnableScrollableCursors=1), you must not include long columns in the Select list.
GetDBListFrom Informix (GDBLFI)	GetDBListFromInformix={0	1}. Determines whether the driver requests the database list to be returned from the INFORMIX server or from the database list that the user entered at driver setup.

When set to 1, the initial default, the driver requests the database list from the INFORMIX server. When set to 0, it uses the list that was entered by the user at driver setup. |

Data Types

The INFORMIX 5 data types map to the standard ODBC data types as follows:

INFORMIX 5	ODBC
Byte*	SQL_LONGVARBINARY
Char	SQL_CHAR
Date	SQL_DATE
Datetime year to fraction(5)	SQL_TIMESTAMP
Decimal	SQL_DECIMAL
Float	SQL_DOUBLE
Integer	SQL_INTEGER
Money	SQL_DECIMAL
Serial	SQL_INTEGER
Smallfloat	SQL_REAL
Smallint	SQL_SMALLINT
Text*	SQL_LONGVARCHAR
Varchar*	SQL_VARCHAR

Not supported for Standard Engine Databases

Note: The Interval data type is not supported. Existing columns of this type are mapped to the ODBC SQL_CHAR data type.

Isolation and Lock Levels Supported

INFORMIX 5 supports isolation levels 0 (read uncommitted), 1 (read committed), and 3 (serializable). The default is 1. The Standard Engine supports isolation level 0 (read uncommitted) only.

INFORMIX 5 also supports an alternative isolation level 1, called cursor stability. Your ODBC application can use this isolation level by calling SQLSetConnectOption (1040,1).

Additionally, if transaction logging has not been enabled for your database,then transactions are not supported by the driver (the driver is always in auto-commit mode).

INFORMIX 5 supports page-level locking.

ODBC Conformance Level

The INFORMIX 5 driver supports the Core, Level 1, and Level 2 API functions listed in <u>Supported ODBC Functions</u>.   In addition, the following Level 2 functions are supported:

- SQLBrowseConnect
- SQLProcedures

The driver supports the core SQL grammar.

Number of Connections and Statements Supported

The INFORMIX 5 database system supports multiple connections and multiple statements per connection.

INGRES Drivers

Two INGRES drivers are available.   One supports INGRES Release 6.4/04 and higher; the other supports Release 6.4/03 and lower.

The driver filenames are QEING*nn*.DLL for Release 6.4/03 and earlier and QEING4*nn*.DLL for Release 6.4/04.

[System Requirements](#)

[Configuring Data Sources](#)

[Connecting to INGRES Using a Logon Dialog Box](#)

[Connecting to INGRES Using a Connection String](#)

[Data Types](#)

[Isolation and Lock Levels Supported](#)

[ODBC Conformance Level](#)

[Number of Connections and Statements Supported](#)

System Requirements

To access INGRES databases from your client workstation you must have the INGRES/Net or INGRES/Star product installed on both your client and server nodes.

To access INGRES, you must have the following software installed:
- For INGRES Release 6.4/03 and earlier, you must have Networking for DOS Release 6.4/01 (net.msd/00) and Networking for Windows Release 6.4/01 (net.win/02)
- For INGRES Release 6.4/04 and later, you must have Networking for Windows Release 6.4/04 (net.win/00)

You must have the environment variable II_SYSTEM set to the directory where you installed the INGRES/Net or INGRES/Star product.   For example:

SET II_SYSTEM=C:

In the [386Enh] section of your SYSTEM.INI file, set the NetHeapSize attribute to a number from 16 to 64 .   When using the NetBIOS protocol, include the line   II_NETBIOS_NODE in your AUTOEXEC.BAT file for proper multiuser access to INGRES.

If you attempt to configure a data source and do not have the files CLW.DLL, ADFW.DLL, LIBQW.DLL, or GCCW.DLL either on your path or in your Windows SYSTEM directory, the following message appears:

The setup routines for the INTERSOLV INGRES ODBC driver could not be loaded. You may be low on memory and need to quit a few applications.

Configuring Data Sources

To configure an INGRES data source, do the following:

1   Start the ODBC Administrator.   A list of data sources appears.

2   If you are configuring a new data source, click **Add**.   A list of installed drivers appears.   Select INTERSOLV INGRES (INTERSOLV INGRES 6.4/04 if you are using INGRES Release 6.4/04) and click **OK**.

If you are configuring an existing data source, select the data source name and click **Setup**.

The setup dialog box appears.

3   Specify values as follows:

Data Source Name: A string that identifies this INGRES data source configuration in ODBC.INI. Examples include "Accounting" or "INGRES-Serv1."

Description: An optional long description of a data source name.   For example, "My Accounting Database" or "INGRES on Server number 1."

Server Name: The name of the virtual node that you defined using the INGRES NETU utility.   This virtual node tells INGRES which system to call, how to call it, and the user's name and password.

Database Name: The name of the database to which you want to connect by default.

The following values are optional:

Server List: The list of servers (virtual nodes) that will be available in the Logon dialog box. Separate the server names with commas.

Database List: The list of databases that will be available in the logon dialog box.   Separate the names with commas.

Default User Name: The default user name used to connect to your INGRES database.   A user name is required only if security is enabled on your database.   Your ODBC application may override this value or you may override this value in the logon dialog box.

Options: Any flag allowed on the INGRES SQL command line.   Some examples are
- -l (locks the database exclusively)
- -u (logs on as username)
- +w or -w (waits/doesn't wait for the database if a user has already opened it exclusively)
- +U or -U (enables/disables user updating of the system tables and locks the database exclusively)
- +Y or -Y (enables/disables user updating of the system tables but does not lock the database exclusively)

Yield Proc: A numeric value that determines whether you can work in other applications when INGRES is busy.   This attribute is useful to users of ODBC applications.   Valid values are:
- 0 (peek and dispatch), which causes the driver to check the Windows message queue and send any messages to the appropriate Windows application.
- 1 (no yielding), which does not let you work in other applications.   This is the default for the INGRES 6.4/03 driver, and it is recommended that you use this value with that driver.   This value is not valid for the INGRES 6.4/04 driver.
- 2 (INGRES' yield procedure), which uses INGRES' default yield procedure.   This is the default for the INGRES 6.4/04 driver, and it is recommended that you use this value with that driver.
- 3 (dispatch via Windows Yield function), which turns control over to the Windows kernel.   The Windows kernel checks the message queue and sends any messages to the appropriate application window.

Repeated Cache Size: An integer value that determines whether all Update and Insert statements are to be executed as repeated statements. This attribute improves the performance of applications

that repeat the same SQL statement. When set to 0, the initial default, no statements are repeated. The recommended setting for this attribute is 100 (RepeatedCacheSize=100).

To repeat a single statement rather than all statements, use the INGRES REPEATED syntax.

Repeated Selects: A value that determines whether the driver optimizes Select statements or executes them as repeated queries. When set to 0, the initial default, the driver executes all Select statements as it did in previous versions of the product.

When set to 1, the driver optimizes Select statements that return only one result row. When set to 2, the driver executes all Select statements as repeated queries. If this attribute is set to 1 or 2, the RepeatedCacheSize attribute must be set to greater than zero.

Setting this option to 1 or 2:
- has no effect on Select statements containing a For Update clause
- limits the driver to one active statement and one active connection

Value Replacement for Repeated Selects: A number that determines whether the driver substitutes parameters for hardcoded values in repeated statements. This option is convenient in applications that do not use dynamic parameters.

When set to 0, the initial default, the driver doesnot substitute parameters. When set to 1, the driver substitutes parameters for hardcoded values and the RepeatedCacheSize attribute must be greater than zero or the REPEATED INGRES keyword must be used.

This option has no effect upon Select statements that contain a For Update clause that requires a cursor or upon statements that already use parameter markers.

This atttribute supports only a subset of standard ODBC SQL grammar. It is intended for performance and does not utilize a full SQL parser. For example, subselects are not supported and use of "is NULL" for columns other than character columns is not supported.

4   Click **Translate** to perform a translation of your data from one character set to another. The Select Translator dialog box appears in which you can select a translator to peform the translation. INTERSOLV provides a translator named INTERSOLV OEM ANSI that translates your data from the IBM PC character set to the ANSI character set. Click **OK** to leave this dialog box and perform the translation.

The translators that are listed in this dialog box are determined by the values listed in the ODBC Translators section of your ODBCINST.INI file.

5   Click **OK** to write these values to ODBC.INI.   These values are now the defaults when you connect to the data source.   You can change the defaults by configuring your data source again.   You can override the defaults by connecting to the data source using a connection string with alternate values.

Connecting to INGRES Using a Logon Dialog Box

Some ODBC applications display a logon dialog box when you are connecting to a data source.   In these cases, the data source name has already been specified.

In the logon dialog box, do the following:

1 Type the server name of the computer containing the INGRES database you want to access or select the name from the Server Name drop-down list, which displays the server names you specified in the setup dialog box.   Server Name must be a valid virtual node that has been added using the INGRES NETU utility.

2 Type the name of the database to which you want to connect or select the name from the Database Name drop-down list, which displays the names you specified in the setup dialog box.

3 If required, type your user name.

4 Type any options for the connection.   The options can be any flags allowed on the INGRES SQL command line.

5 Click **OK** to log on to INGRES and to update the values in ODBC.INI.

Connecting to INGRES Using a Connection String

If your application requires a connection string to connect to a data source, you must specify the data source name that tells the driver which ODBC.INI section to use for the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in ODBC.INI.   These values are not written to ODBC.INI.

You can specify either long or short names in the connection string.   The connection string has the form:

DSN=*data_source_name*[;*attribute=value*[;*attribute=value*]...]

An example of a connection string for INGRES is

DSN=INGRES TABLES;SRVR=QESERV;DB=PAYROLL;UID=JOHN

The following table gives the long and short names for each attribute, as well as a description.

The defaults listed in the table are initial defaults that apply when no value is specified in either the connection string or in the data source definition in ODBC.INI.   If you specified a value for the attribute when configuring the data source, that value is your default.

Attribute	Description
DataSourceName (DSN)	A string that identifies an INGRES data source configuration in ODBC.INI. Examples include "Accounting" or "INGRES-Serv1."
ServerName (SRVR)	The name of the virtual node that you defined using the INGRES NETU utility. This virtual node tells INGRES which system to call, how to call it, and the user's name and password.
Database (DB)	The name of the database to which you want to connect.
LogonID (UID)	The default logon ID (user name) used to connect to your INGRES database. A logon ID is required only if security is enabled on your database. If so, contact your system administrator to get your logon ID.
Options (OPTS)	The flags allowed on the INGRES SQL command line. Some examples are ▪ -l (locks the database exclusively) ▪ -u (logs on as username) ▪ -w or -w (waits/doesn't wait for the database if someone has already opened it exclusively) ▪ +U or -U (enables/disables user updating system tables and locks the database exclusively) ▪ +Y or -Y (enables/disables user updating system tables but does not lock the database exclusively)
YieldProc (YLD)	YieldProc={0 | 1 | 2 | 3}. A value that determines if you can work in other applications when INGRES is busy. This attribute is useful to users of ODBC applications. ▪ YieldProc=0 (peek and dispatch) causes the driver to check the Windows message queue and send any messages to the appropriate Windows application. ▪ YieldProc=1 (no yielding), which does not let you work in other applications. This is the default for

the INGRES 6.4/03 driver, and it is recommended that you use this value with that driver. This value is not valid for the INGRES 6.4/04 driver.

- ▪ YieldProc=2 (INGRES' yield procedure), which uses INGRES' default yield procedure. This is the default for the INGRES 6.4/04 driver, and it is recommended that you use this value with that driver.
- ▪ YieldProc=3 (dispatch via Windows Yield function) turns control over to the Windows kernel. The Windows kernel checks the message queue and sends any messages to the appropriate application window.

RepeatedCache Size (RCS)	Determines whether all Update and Insert statements are to be executed as repeated statements. This attribute improves the performance of applications that repeat the same SQL statement. When set to 0, the initial default, no statements are repeated. The recommended setting for this attribute is 100 (RepeatedCacheSize=100). To repeat a single statement rather than all statements, use the INGRES REPEATED syntax.
RepeatedSelects (RS)	RepeatedSelects={0 | 1 | 2}. Determines whether the driver optimizes Select statements or executes them as repeated queries. When set to 0, the initial default, the driver executes all Select statements as it did in previous versions of the product. When set to 1, the driver optimizes Select statements that return only one result row. When set to 2, the driver executes all Select statements as repeated queries. If this attribute is set to 1 or 2, the RepeatedCacheSize attribute must be set to greater than zero. Setting this option to 1 or 2: ▪ has no effect on Select statements containing a For Update clause ▪ limits the driver to one active statement and one active connection
ValueReplacement (VR)	ValueReplacement={0 | 1}. A number that determines whether the driver substitutes parameters for hardcoded values in repeated statements. This option is convenient in applications that do not use dynamic parameters. When set to 0, the initial default, the driver does not substitute parameters. When set to 1, the driver substitutes parameters for hardcoded values and the RepeatedCacheSize attribute must be greater than zero or the REPEATED INGRES keyword must be used. This option has no effect upon Select statements that contain a For Update clause that requires a cursor or upon statements that already use parameter markers.

This attribute supports only a subset of standard ODBC SQL grammer. It is intended for performance and does not utilize a full SQL parser. For example, subselects are not supported and use of is NULL for columns other than character columns is not supported.

Data Types

INGRES data types map to the standard ODBC data types as follows:

INGRES	ODBC
Char	SQL_CHAR
Date	SQL_TIMESTAMP
Float	SQL_DOUBLE
Float4	SQL_REAL
Integer	SQL_INTEGER
Integer1	SQL_TINYINT
Money	SQL_DECIMAL
Smallint	SQL_SMALLINT
Varchar	SQL_VARCHAR

Note   INGRES date values do not directly map to the ODBC type SQL_TIMESTAMP. If interval data is placed in Date columns, then the driver raises an error when attempting to read the value.

Isolation and Lock Levels Supported

INGRES supports isolation levels 1 (read committed, the default) and 3 (serializable).   INGRES supports page-level locking.

ODBC Conformance Level

The INGRES driver supports the Core, Level 1, and Level 2 API functions listed in Supported ODBC Functions.   In addition, the SQLBrowseConnect Level 2 function is supported.

The driver supports the minimum SQL grammar.

Number of Connections and Statements Supported

The INGRES database system supports multiple connections and multiple statements per connection. Note that if you set the RepeatedSelects connection string attribute to 1 or 2, the driver is limited to one active statement and one active connection.

InterBase Driver

The InterBase driver supports version 4.0 of the InterBase database system.

The driver filename is QEINT*nn*.DLL.

[System Requirements](#)

[Configuring Data Sources](#)

[Connecting to InterBase Using a Logon Dialog Box](#)

[Connecting to InterBase Using a Connection String](#)

[Data Types](#)

[Isolation and Lock Levels Supported](#)

[ODBC Conformance Level](#)

[Number of Connections and Statements Supported](#)

System Requirements

System requirements are as follows:
- InterBase 4.0.
- The new Client-Library requires a PC user to purchase the Windows client piece for InterBase databases.   Also, the InterBase client uses TCP/IP as the communications protocol to connect to a UNIX server.

If you attempt to configure a data source and you do not have the InterBase directory on your path or the driver DLLs in your Windows SYSTEM directory, the following message appears:

The setup routines for the INTERSOLV InterBase ODBC driver could not be loaded. You may be low on memory and need to quit a few applications.

Configuring Data Sources

To configure a InterBase data source, do the following:

1 Start the ODBC Administrator.   A list of data sources appears.

2 If you are configuring a new data source, click **Add**.   A list of installed drivers appears.   Select INTERSOLV InterBase and click **OK**.

If you are configuring an existing data source, select the data source name and click **Setup**.

The setup dialog box appears.

3 Specify values as follows:

Data Source Name: A string that identifies this InterBase data source configuration in ODBC.INI. Examples include "Accounting" or "InterBase-Serv1."

Description: An optional long description of a data source name.   For example, "My Accounting Database" or "InterBase on Server #1."

Database Name: The name of the database to which you want to connect by default.

The following values are optional:

Database List: The list of databases that will be available in the logon dialog box. Separate the list with commas.

Default User Name: The default user name used to connect to your InterBase database.   This ID is case-sensitive.   Your ODBC application may override this value or you may override this value in the logon dialog box.

Lock Time Out: A value that specifies whether InterBase should wait for a lock to be freed before raising an error when processing a Select...For Update Of statement.   Values can be -1 (wait forever) or 0 (don't wait).   The default is -1.

4 Click **Translate** to perform a translation of your data from one character set to another. The Select Translator dialog box appears in which you can select a translator to peform the translation. INTERSOLV provides a translator named INTERSOLV OEM ANSI that translates your data from the IBM PC character set to the ANSI character set. Click **OK** to leave this dialog box and perform the translation.

The translators that are listed in this dialog box are determined by the values listed in the ODBC Translators section of your ODBCINST.INI file.

5 Click **OK** to write these values to ODBC.INI.   These values are now the defaults when you connect to the data source.   You can change the defaults by configuring your data source again.   You can override the defaults by connecting to the data source using a connection string with alternate values.

Connecting to InterBase Using a Logon Dialog Box

Some ODBC applications display a logon dialog box when you are connecting to a data source.   In these cases, the data source name has already been specified.

In the logon dialog box, do the following:

1   Type the name of the server containing the InterBase tables you want to access (case-sensitive) or click the arrow to the right of the box to select a server name you specified in the setup dialog box.

2   If required, type your case-sensitive user name.

3   If required, type your case-sensitive password for the system.

4   Click **OK** to log on to the InterBase database installed on the server you specified and to update the values in ODBC.INI.

Connecting to InterBase Using a Connection String

If your application requires a connection string to connect to a data source, you must specify the data source name that tells the driver which section of ODBC.INI to use for the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in ODBC.INI.   These values are not written to ODBC.INI.

You can specify either long or short names in the connection string.   The connection string has the form:

DSN=*data_source_name*[;*attribute=value*[;*attribute=value*]...]

An example of a connection string for InterBase is

DSN=ACCOUNTING;DB=PAYRLL;UID=JOHN;PWD=XYZZY

The following table gives the long and short names for each attribute, as well as a description.

The defaults listed in the table are initial defaults that apply when no value is specified in either the connection string or in the data source definition in ODBC.INI.   If you specified a value for the attribute when configuring the data source, that value is your default.

Attribute	Description
DataSourceName (DSN)	A string that identifies an InterBase data source configuration in ODBC.INI. Examples include "Accounting" or "InterBase-Serv1."
Description	An optional long description of a data source name. For example, "My Accounting Database" or "InterBase on Server #1."
Database (DB)	The name of the database to which you want to connect.
LogonID (UID)	The case-sensitive user name used to connect to your InterBase database.
Password (PWD)	A case-sensitive password.
LockTimeOut (LTO)	LockTimeOut={0 | -1}. Specifies whether InterBase should wait for a lock to be freed before raising an error when processing a Select...For Update Of statement. LockTimeOut=-1, the initial default, means wait forever. LockTimeOut=0 means no waiting for locks (return error).
BinarySegmentSize (BSS)	The segment size used to create a binary blob. The size of the segment determines how many bytes the segment can hold. However, this value does not reflect the maximum size of the blob. The default is 80.
CharSegmentSize (CSS)	The segment size used to create a text blob. Blobs are stored in segments. The size of the segment determines how many bytes the segment can hold. However, this value does not reflect the maximum size of the blob. The default is 80.

Data Types

InterBase data types are mapped to the standard ODBC data types as follows:

InterBase	ODBC
Blob (SegmentSize*, 0)	SQL_LONGVARBINARY
Blob (SegmentSize*, 1)	SQL_LONGVARCHAR
Char	SQL_CHAR
Date	SQL_DATE
Decimal	SQL_DECIMAL
Double Precision	SQL_DOUBLE
Float	SQL_REAL
Integer	SQL_INTEGER
Smallint	SQL_SMALLINT
Varchar	SQL_VARCHAR

* Determined by the value set for the BinarySegmentSize (BSS) or CharSegmentSize (CSS) attribute.
Notes: The Blob data types cannot be used in a Where clause and cannot be inserted into a column as a string.

The array data type is not supported.

Isolation and Lock Levels Supported

InterBase supports isolation levels 1 (read committed), 3 (serializable, the default), and 4 (versioning).

InterBase performs *optimistic locking*.   This means that InterBase does not really obtain locks on a record; instead, it uses a comparison method.   Whenever an Update operation is performed, the server compares the user's value of the record with the most current value in the table.   If the values are the same, the update succeeds.   Differing values indicate conflict, and the update fails.

ODBC Conformance Level

The InterBase driver supports the Core, Level 1, and Level 2 API functions listed in <u>Supported ODBC Functions</u>.   In addition, the following Level 2 functions are supported:
- SQLPrimaryKeys
- SQLProcedureColumns
- SQLProcedures

Number of Connections and Statements Supported

The InterBase database system supports multiple connections and multiple statements per connection.

MicroDecisionware Driver

The MicroDecisionware driver supports the IBM Database 2 (DB2) and InfoHub (formerly Teradata) database systems.   The MicroDecisionware Database Gateway makes an InfoHub database resemble a DB2 database. These topics refer to all databases accessed through the MicroDecisionware Gateway as DB2 databases.   Catalog functions are not currently supported for InfoHub.

The driver filename is QEMDI*nn*.DLL.

[System Requirements](#)

[Configuring Data Sources](#)

[Connecting to DB2 Using a Logon Dialog Box](#)

[Connecting to DB2 Using a Connection String](#)

[Data Types](#)

[Isolation and Lock Levels Supported](#)

[ODBC Conformance Level](#)

[Number of Connections and Statements Supported](#)

System Requirements

The MicroDecisionware driver must be used with MicroDecisionware's Database Gateway running on an OS/2 server to access DB2 and InfoHub databases.

Before you can use the MicroDecisionware driver, you must install the MicroDecisionware Database gateway and the MicroDecisionware Gateway client appropriate to your workstation.

You can test your connection to the database gateway by using ISQL or SAF.

If you attempt to configure a data source and you do not have W3DBLIB.DLL on your path or in your Windows SYSTEM directory, the following message appears:

The setup routines for the INTERSOLV MicroDecisionware ODBC driver could not be loaded. You may be low on memory and need to quit a few applications.

Configuring Data Sources

To configure a MicroDecisionware DB2 data source, do the following:

1 Start the ODBC Administrator.   A list of data sources appears.

2 If you are configuring a new data source, click **Add**.   A list of installed drivers appears.   Select INTERSOLV MDI and click **OK**.

If you are configuring an existing data source, select the data source name and click **Setup**.

The setup dialog box appears.

3 Specify values as follows:

Data Source Name: A string that identifies this MicroDecisionware data source configuration in ODBC.INI.   Examples include "Accounting" or "MicroDecisionware Serv1."

Description: An optional long description of a data source name.   For example, "My Accounting Database" or "MicroDecisionware on Server number 1."

Server Name: The name of the OS/2 server running the MicroDecisionware Database Gateway. This value will appear as the initial default in the logon dialog box.

The following values are optional:

Server List: The list of servers that will appear in the logon dialog box.   Separate the server names with commas.

Default Authorization ID: The default authorization ID used to connect to your DB2 database. This ID is case-sensitive.   An authorization ID is required only if security is enabled on your database.   Your ODBC application may override this value or you may override this value in the logon dialog box.

Groups: A list of the groups to which you belong, used to determine which tables you can access. Separate each group name with a comma.   Alternatively, enter the word ALL so that you can see all table names even if you cannot access the table.

Application Name: The name used to identify your application.

Workstation ID: The workstation ID used by the MicroDecisionware Database Gateway.

Cancel Behavior: A value to specify how a previously executed statement should be canceled.
- 0 fetches all remaining records if the statement was a Select.
- 1 cancels the statement by calling dbcancel.   This is the default.
- 2 closes the connection to the server for the statement.

Yield Proc: A numeric value that determines whether you can work in other applications when the gateway is busy.   This attribute is useful to users of ODBC applications.   Valid values are:
- 0 (peek and dispatch), which causes the driver to check the Windows message queue and send any messages to the appropriate Windows application.
- 1 (no yielding, the default), which does not let you work in other applications.
- 3 (dispatch via Windows Yield function), which turns control over to the Windows kernel.   The Windows kernel checks the message queue and sends any messages to the appropriate application window.

It is recommended that you use the value 1.   The values 0 and 3 do not work with all Windows applications.

Owner of Catalog Tables: The owner of views of the system tables.   The tables themselves are owned by SYSIBM.   If you have read access to the system tables, this option does not have to be changed.   If you do not have read access, the database administrator must create a view of the system tables in another account and give you permission to use the view.   Enter the authorization ID for the account that owns the views for the system tables.

Tablespace to Create Tables in: The name of the tablespace in which new tables are stored.   For example, DSNDB04.

DB2 Site Allows User to Log On More Than Once: A check box that specifies that your site lets a user log on to DB2 more than once.   Selecting this check box may improve performance.

NETAPI.DLL Library Available: A check box that specifies that the driver can use NETAPI.DLL to get the name of your workstation.   Most major PC networks support this feature.   If your network does not support this capability, deselect this option.   If you supply a workstation ID, this setting is ignored.

Use Catalog Stored Procedures: A check box that specifies whether the driver uses Catalog Stored Procedures or direct SQL to obtain system catalog information.

4   Click **Translate** to perform a translation of your data from one character set to another. The Select Translator dialog box appears in which you can select a translator to peform the translation. INTERSOLV provides a translator named INTERSOLV OEM ANSI that translates your data from the IBM PC character set to the ANSI character set. Click **OK** to leave this dialog box and perform the translation.

The translators that are listed in this dialog box are determined by the values listed in the ODBC Translators section of your ODBCINST.INI file.

5   Click **OK** to write these values to ODBC.INI.   These values are now the defaults when you connect to the data source.   You can change the defaults by configuring your data source again.   You can override the defaults by connecting to the data source using a connection string with alternate values.

Connecting to DB2 Using a Logon Dialog Box

Some ODBC applications display a logon dialog box when you are connecting to a data source.   In these cases, the data source name has already been specified.

In the logon dialog box, do the following:

1   Type the name of the OS/2 server running the Database Gateway you wish to access or select the name from the Gateway Server box, which displays the server names specified in the Setup dialog box.

2   If required, type your Authorization ID (case-sensitive).

3   If required, type your password for the system (case-sensitive).

4   Click **OK** to complete the logon and to update the values in ODBC.INI.

Connecting to DB2 Using a Connection String

If your application requires a connection string to connect to a data source, you must specify the data source name that tells the driver which ODBC.INI section to use for the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in ODBC.INI. These values are not written to ODBC.INI.

You can specify either long or short names in the connection string. The connection string has the form:

DSN=*data_source_name*[;*attribute=value*[;*attribute=value*]...]

An example of a connection string for the MicroDecisionware Database Gateway is

DSN=DB2 VIA MDI;SRVR=QESRVR;UID=JOHN;PWD=XYZZY;APP=Q4

The following table gives the long and short names for each attribute, as well as a description.

The defaults listed in the table are initial defaults that apply when no value is specified in either the connection string or in the data source definition in ODBC.INI. If you specified a value for the attribute when configuring the data source, that value is your default.

Attribute	Description
DataSourceName (DSN)	A string that identifies a MicroDecisionware data source configuration in ODBC.INI. Examples include "Accounting" or "MicroDecisionware Serv1."
ServerName (SRVR)	The name of the OS/2 server running the MicroDecisionware Database Gateway.
LogonID (UID)	The default logon ID (authorization ID) used to connect to your DB2 database. This ID is case-sensitive. A logon ID is required only if security is enabled on your database. If so, contact your system administrator to get your logon ID.
Password (PWD)	A case-sensitive password.
Groups (GRP)	A list of the group to which you belong, used to determine which tables you can access. Set Groups to the names of any groups to which you belong; separate each name with a comma. Groups=ALL lets your application see all table names even if you cannot access the table.
ApplicationName (APP)	The name used to identify your application.
WorkstationID (WKID)	The workstation ID used by the MicroDecisionware Database Gateway.
Cancel (CAN)	Cancel={0 | 1 | 2}. A value that specifies how a previously executed statement should be canceled.
	▪ Cancel=0 fetches all of the remaining records if the statement was a Select.
	▪ Cancel=1 cancels the statement by calling dbcancel. Set Cancel=1 if dbcancel is supported in your client/server configuration. This is the initial default.
	▪ Cancel=2 closes the connection to the server for the statement. Set Cancel=2 only if dbcancel is not supported for your configuration and the performance of fetching all remaining records is unacceptable.

Sysibm (SI)	The authorization ID for the system tables. The initial default is SYSIBM.
CreateDB (CDB)	The name of the tablespace in which new tables are stored. For example, CreateDB=DSNDB04.
MultipleLogon (MULT)	MultipleLogon={0	1}. A value that specifies whether the DB2 site lets users log on more than once. Set this attribute to 1 if this is the case; 0 otherwise.
Netapi (NAPI)	Netapi={0	1}. A value that specifies whether NETAPI.DLL is available. Most major PC networks support this feature. Netapi=0 indicates it is not available; Netapi=1 indicates it is available. If you supply a value for the attribute WorkstationID, this attribute is ignored.
YieldProc (YLD)	YieldProc={0	1	3}. This option, which determines if you can work in other applications when the gateway is busy, is useful to users of ODBC applications. ▪ YieldProc=0 (peek and dispatch) causes the driver to check the Windows message queue and to send any messages to the appropriate Windows application. ▪ YieldProc=1 (no yielding, the initial default) does not let you work in other applications. ▪ YieldProc=3 (dispatch via Windows Yield function) turns control over to the Windows kernel. The Windows kernel checks the message queue and sends any messages to the appropriate application window. It is recommended that you use YieldProc=1. YieldProc=0 and YieldProc=3 do not work with all Windows applications.
ModifySQL (MS)	ModifySQL={0	1}. This attribute is provided for backward compatibility. Specify ModifySQL=0 to have the driver understand SQL dialects found in earlier drivers. Specify ModifySQL=1 (the default) to have the driver modify the SQL statement to conform to ODBC specifications.
UseCatalogStored Procedures (UCSP)	UseCatalogStored Procedures={0	1}. This attribute determines whether the driver uses Catalog Stored Procedures or direct SQL to obtain system catalog information. If set to 0, the driver uses direct SQL. The initial default is 0.
ProfileName (PN)	For InfoHub only. This attribute is the InfoHub profile name. If you specify a name, the driver issues a Use profilename statement at data source connect time.

Data Types

The DB2 data types are mapped to the standard ODBC data types as follows:

DB2	ODBC
Char	SQL_CHAR
Char() For Bit Data	SQL_BINARY
Date	SQL_DATE
Decimal	SQL_DECIMAL
Float	SQL_DOUBLE
Integer	SQL_INTEGER
Long Varchar	SQL_LONGVARCHAR
Long Varchar For Bit Data	SQL_LONGVARBINARY
Smallint	SQL_SMALLINT
Time	SQL_TIME
Timestamp	SQL_TIMESTAMP
Varchar	SQL_VARCHAR
Varchar() For Bit Data	SQL_VARBINARY

Note: The Graphic, Vargraphic, and Long Vargraphic data types are not supported.   If you create DB2 tables using these types, you will not be able to read the records in those tables.

Isolation and Lock Levels Supported

DB2 supports isolation levels 1 (repeatable read) and 2 (read committed, the default).   DB2 supports page-level locking.

ODBC Conformance Level

The MicroDecisionware driver supports the Core, Level 1, and Level 2 API functions listed in [Supported ODBC Functions](#).   The driver supports the minimum SQL grammar.

Number of Connections and Statements Supported

If your site does not allow multiple connections, the MicroDecisionware gateway supports a single connection and one statement per connection.

If your site allows multiple connections and SQL_AUTOCOMMIT is 1, the MicroDecisionware gateway supports multiple connections and multiple statements per connection.

If your site allows multiple connections and SQL_AUTOCOMMIT is 0, the MicroDecisionware gateway supports multiple connections and one statement per connection.

Oracle Drivers

Two separate Oracle drivers support the Oracle 6 and Oracle 7 database systems.

The driver filename for Oracle 6 is QEORA*nn*.DLL, and for Oracle 7, QEOR7*nn*.DLL.

[Oracle 6 System Requirements](#)

[Oracel 7 System Requirements](#)

[Configuring Data Sources](#)

[Connecting to Oracle Using a Logon Dialog Box](#)

[Connecting to Oracle Using a Connection String](#)

[Oracle 6 Data Types](#)

[Oracle 7 Data Types](#)

[Isolation and Lock Levels Supported](#)

[ODBC Conformance Level](#)

[Number of Connections and Statements Supported](#)

Oracle 6 System Requirements

The Oracle SQL*Net product is required to access remote Oracle databases.

For the DOS-based SQL*Net TSRs, you must start the SQL*Net TSRs before you start Windows.   You also must include a statement in your AUTOEXEC.BAT file to enable Oracle to find the Oracle configuration file, CONFIG.ORA.   For example, if the file is located in the ORACLE6 directory, then the following statement must be included in your AUTOEXEC.BAT file:

SET CONFIG=C:\ORACLE6\CONFIG.ORA

To use the Windows-based SQL*Net DLLs with ORA6WIN.DLL, the directory containing the DLLs must be on your path.   Generally, these DLLs are installed in C:\ORAWIN.   The Oracle configuration file for Windows, ORACLE.INI, is placed in the Windows directory.

To use ORA6WIN.DLL with the SQL*Net DLLs and have it find its error message file, you must include a statement in your AUTOEXEC.BAT file.   For example, if your Windows home directory is WIN31, then the following statement must be included in your AUTOEXEC.BAT file:

SET CONFIG=C:\WIN31\ORACLE.INI

Oracle 7 System Requirements

The Oracle SQL*Net product is required to access remote Oracle databases.

If you are using an Oracle 7 data source and have SQL*Net 2.0, you need ORA71WIN.DLL, CORE3WIN.DLL, and NLS23WIN.DLL.

If you do not have SQL*Net 2.0, you need ORA7WIN.DLL, ORA71WIN.DLL, COREWIN.DLL, CORE3WIN.DLL, USDMEM.DLL, and NLS23WIN.DLL.

These DLLs are included on with the drivers on your INTERSOLV release disks.   By default, the Setup program installs these files in your Windows SYSTEM directory.   If you install them in another directory, that directory must be on your path.

If you attempt to configure an Oracle data source and you do not have the above DLLs on your path or in your Windows SYSTEM directory, the following message appears:

The setup routines for the INTERSOLV Oracle7 ODBC driver could not be loaded. You may be low on memory and need to quit a few applications.

Configuring Data Sources

To configure an Oracle data source, do the following:

1   Start the ODBC Administrator.   A list of data sources appears.

2   If you are configuring a new data source, click **Add**.   A list of installed drivers appears.   Select INTERSOLV Oracle or INTERSOLV Oracle7 and click **OK**.

If you are configuring an existing data source, select the data source name and click **Setup**.

The setup dialog box appears.

3   Specify values as follows:

Data Source Name: A string that identifies this Oracle data source configuration in ODBC.INI. Examples include "Accounting" or "Oracle-Serv1."

Description: An optional long description of a data source name.   For example, "My Accounting Database" or "Oracle on Server number 1."

Server Name: The SQL*Net connection string designating the server and database to be accessed. The information required varies depending on the SQL*Net driver you are using.   The topic Connecting to Oracle Using a Connection String gives the format of the SQL*Net connection string.

The following values are optional:

Server List: The list of SQL*Net connection strings that will appear in the logon dialog box. Separate the strings with commas.   If the SQL*Net connection string contains a comma, enclose it in quotation marks; for example, "Serv,1", "Serv,2", "Serv,3."

Default User Name: The default user name used to connect to your Oracle database.   A default user name is required only if security is enabled on your database.   Your ODBC application may override this value or you may override this value in the logon dialog box or connection string.

Lock Time Out: A value that specifies whether Oracle should wait for a lock to be freed before raising an error when processing a Select...For Update Of statement.   Values can be -1 (wait forever) or 0 (don't wait).   The default is -1.

Array Size: The number of bytes the driver uses for fetching multiple rows.   Values can be 0 to 65536; the default is 60000.   Larger values increase throughput by reducing the number of times the driver fetches data across the network.   Smaller values increase response time, as there is less of a delay waiting for the server to transmit data.

Catalog Comments: Check this box if you want to retrieve the contents of the COMMENTS column in your Oracle tables.   Doing so may impact the performance of your queries.

4   Click **Translate** to perform a translation of your data from one character set to another. The Select Translator dialog box appears in which you can select a translator to peform the translation. INTERSOLV provides a translator named INTERSOLV OEM ANSI that translates your data from the IBM PC character set to the ANSI character set. Click **OK** to leave this dialog box and perform the translation.

The translators that are listed in this dialog box are determined by the values listed in the ODBC Translators section of your ODBCINST.INI file.

5   Click **OK** to write these values to ODBC.INI.   These values are now the defaults when you connect to the data source.   You can change the defaults by configuring your data source again.   You can override the defaults by connecting to the data source using a connection string with alternate values.

Connecting to Oracle Using a Logon Dialog Box

Some ODBC applications display a logon dialog box when you are connecting to a data source.   In these cases, the data source name has already been specified.

In the logon dialog box, do the following:

1   Type the SQL*Net connection string of the computer containing the Oracle database tables you want to access or select the string from the Server Name drop-down list, which displays the names you specified in the setup dialog box.

2   If required, type your Oracle user name.

3   If required, type your Oracle password.

4   Click **OK** to log on to the Oracle database installed on the server you specified and to update the values in ODBC.INI.

Connecting to Oracle Using a Connection String

If your application requires a connection string to connect to a data source, you must specify the data source name that tells the driver which ODBC.INI section to use for the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in ODBC.INI.   These values are not written to ODBC.INI.

You can specify either long or short names in the connection string.   The connection string has the form:

DSN=*data_source_name*[;*attribute=value*[;*attribute=value*]...]

An example of a connection string for Oracle is

DSN=Accounting;SRVR=X:QESRVR;UID=JOHN;PWD=XYZZY

If the server name contains a semicolon, enclose it in quotation marks:

DSN=Accounting;SRVR="X:QE;SRVR";UID=JOHN;PWD=XYZZY

The following table gives the long and short names for each attribute, as well as a description.

The defaults listed in the table are initial defaults that apply when no value is specified in either the connection string or in the data source definition in ODBC.INI.   If you specified a value for the attribute when configuring the data source, that value is your default.

Attribute	Description
DataSourceName (DSN)	A string that identifies an Oracle data source configuration in ODBC.INI. Examples include "Accounting" or "Oracle-Serv1."
LogonID (UID)	The logon ID (user name) that the application uses to connect to your Oracle database. A logon ID is required only if security is enabled on your database. If so, contact your system administrator to get your logon ID.
Password (PWD)	Your password.
LockTimeOut (LTO)	LockTimeOut={0 | 1}. A value that specifies whether Oracle should wait for a lock to be freed before raising an error when processing a Select...For Update statement. Values can be -1 (wait forever, the initial default) or 0 (don't wait).
ArraySize (AS)	The number of bytes the driver uses for fetching multiple rows. Values can be 0 to 65536. The initial default is 60000. Larger values increase throughput by reducing the number of times the driver fetches data across the network. Smaller values increase response time, as there is less of a delay waiting for the server to transmit data.
ServerName (SRVR)	The SQL*Net connection string designating the server and database to be accessed. The information required varies depending on the SQL*Net driver you are using. The SQL*Net connection string has the following form: *driver_prefix:computer_name*[:*sid*] *driver_prefix* is a letter identifying the network protocol being used. The driver prefix can be as follows: P (named pipes), X (SPX), B (NetBIOS), T (TCP/IP), D (DECNet), A (Oracle Async), or AT (AppleTalk). Check your Oracle documentation for

other protocols.

computer_name is the name of the Oracle Listener on your network.

sid is the Oracle System Identifier and refers to the instance of Oracle running on the host. This item is required when connecting to systems that support more than one instance of an Oracle database.

For local servers, the SQL*Net connection string has the form:

database_name

database_name identifies your Oracle database.

If the SQL*Net connection string contains semicolons, enclose it in quotation marks. See your SQL*Net documentation for more information.

CatalogComments (CC)	CatalogComments={0 | 1}. A value that specifies whether the driver returns the contents of the COMMENTS column for catalog functions. CatalogComments=1 returns COMMENTS. Retrieving the COMMENTS column may reduce the performance of your data catalog operations. CatalogComments=0 does not return COMMENTS (the initial default).

Oracle 6 Data Types

Oracle 6 data types are mapped to the standard ODBC data types as follows:

Oracle 6	ODBC
Char	SQL_VARCHAR
Date	SQL_TIMESTAMP
Long	SQL_LONGVARCHAR
Long Raw	SQL_LONGVARBINARY
Number	SQL_DOUBLE
Number(p,s)	SQL_DECIMAL
Raw	SQL_VARBINARY

Oracle 7 Data Types

The Oracle 7 data types are mapped to the standard ODBC data types as follows:

Oracle 7	ODBC
Char	SQL_CHAR
Date	SQL_TIMESTAMP
Long	SQL_LONGVARCHAR
Long Raw	SQL_LONGVARBINARY
Number	SQL_DOUBLE
Number(p,s)	SQL_DECIMAL
Raw	SQL_VARBINARY
Varchar2	SQL_VARCHAR

Isolation and Lock Levels Supported

Oracle 6 and 7 both support isolation level 2 (repeatable read) only.   Both versions support record-level locking.

ODBC Implementation Level

Both Oracle drivers support the Core, Level 1, and Level 2 API functions listed in [Supported ODBC Functions](#).

The Oracle 6 driver also supports the Level 2 function SQLBrowseConnect.

The Oracle 7 driver also supports the following Level 2 functions:
- SQLBrowseConnect
- SQLProcedures
- SQLPrimaryKeys

Both drivers support the core SQL grammar.

Number of Connections and Statements Supported

Oracle 6 and Oracle 7 support multiple connections and multiple statements per connection.

Paradox 4 Driver

The Paradox 4 driver supports Paradox 3.0, 3.5, 4.0, and 4.5 tables.

The driver filename is QEPDX*nn*.DLL.

System Requirements

You must install the Windows version of the Paradox Engine's dynamic link library (PXENGWIN.DLL) on your computer or on your network server before you can open Paradox tables.   INTERSOLV does not provide PXENGWIN.DLL.   You must purchase Borland's Paradox Engine version 3.0 or later, which contains this DLL.

If you attempt to configure a data source and you do not have PXENGWIN.DLL on your path or in your Windows SYSTEM directory, the following message appears:

> The setup routines for the INTERSOLV Paradox4File ODBC driver could not be loaded. You may be low on memory and need to quit a few applications.

In addition, the DOS SHARE program must be loaded to access Paradox tables.

Multiuser Access to Tables

You can query Paradox tables that reside in a shared directory on a network or that are to be shared among applications running on a local workstation.   If the tables are on a network server, multiple users can query these tables simultaneously.   To share Paradox tables among multiple users, the tables must be located in a shared directory on your network server.

Since the process of establishing shared directories varies from network to network, contact your system administrator to set up a shared directory and to configure each user's environment to access the directory properly.   See the *Borland Paradox Engine User's Guide* for information on specifying the file-sharing characteristics of the tables accessed by Paradox.   You specify these settings using the Paradox Engine Configuration utility (PXENGCFG.EXE).

Whenever you open a Paradox table that another user opens at the same time, the consistency of the data becomes an issue if both individuals are updating the table.

Locking

The Paradox 4 driver has two locking levels: record locking and table locking.   Tables that have no primary key always have a prevent write lock placed on the table when the table is opened.   The table lock is escalated to a write lock when an operation that changes the table is attempted.

Tables that have primary keys use record locking.   The locking level is escalated from record locking to a table write lock if the transaction runs out of record locks.   You can configure the number of record locks permitted by using the Paradox Engine Configuration utility (PXENGCFG.EXE).   The maximum number of record locks is 128.

If you set the connection string attribute ReadConsistency to 1, record locking is turned off and the locking that exists is the same as tables with no primary key.   When this attribute is set to 0, record locking is used on tables that have a primary key.

Primary keys provide the greatest concurrency for tables that are accessed and modified by multiple users.

Note   If a table lock is placed on a Paradox table, the Paradox 4 driver prevents users from updating and deleting records but does not prevent them from inserting records into the locked table.

Configuring Data Sources

To configure a Paradox 4 data source, do the following:

1   Start the ODBC Administrator.   A list of data sources appears.

2   If you are configuring a new data source, click **Add**.   A list of installed drivers appears.   Select INTERSOLV Paradox4File and click **OK**.

If you are configuring an existing data source, select the data source name and click Setup.

The setup dialog box appears.

3   Specify values as follows:

Data Source Name: A string that identifies this Paradox data source configuration in ODBC.INI. Examples include "Accounting" or "Paradox-Serv1."

Description: An optional long description of a data source name.   For example, "My Accounting Database" or "Paradox Files on Server number 1."

Database Directory: The directory in which the Paradox files are stored.   If a directory is not specified, the current working directory is used.

The following values are optional:

Sort Order: The order in which records are sorted in a Paradox index.   This option is used for compatibility with international versions of Paradox.   Valid values are ASCII (the default), INTL, NORDAN, NORDAN4, and SWEDFIN.   NORDAN4 is only for Paradox 4.0 and 4.5 tables.

File Open Cache: A numeric value to specify the maximum number of unused file opens to cache. For example, the value 4 specifies that when a user opens and closes four tables, the tables are not actually closed.   The driver keeps them open so that if another query uses one of these tables, the driver does not have to perform another open, which is expensive.   The advantage of file open caching is increased performance.   The disadvantage is that a user who specifies file locking on open may get a locking conflict even though no one appears to have the file open.   The default is 0, which means no file open caching.

Create Type: The table version for Create Table statements.   The default is 4 for Paradox 4.0 or 4.5 tables.   Specify 3 for Paradox 3.0 or 3.5 tables.

Lock While Reading Blobs: This check box, when selected, indicates that records containing blob data types (BLOB, OLE, Graphic, Formatted Memo, and Memo) are to be locked before they are fetched.

International Sort: A setting to indicate the order in which records are retrieved when you issue a Select statement with an Order By clause.   Select this check box to use the international sort order as defined by your operating system.   International sort order is case-insensitive (*a* precedes *B*); the sorting of accented characters is also affected (see your operating system documentation).   Leave this box blank to use ASCII sort order.   ASCII sort order is case-sensitive, where uppercase letters precede lowercase letters (*B* precedes *a*).

Leave this box blank to use the ASCII sort order, where uppercase letters precede lower case letters (*B* precedes *a*).

Read Consistency Isolation Level: This check box, when selected, indicates that tables are locked to assure that all records can be reread within a statement.

4   Click **Translate** to perform a translation of your data from one character set to another. The Select Translator dialog box appears in which you can select a translator to peform the translation. INTERSOLV provides a translator named INTERSOLV OEM ANSI that translates your data from the IBM PC character set to the ANSI character set. Click **OK** to leave this dialog box and perform the translation.

The translators that are listed in this dialog box are determined by the values listed in the ODBC Translators section of your ODBCINST.INI file.

5   Click **OK** to write these values to ODBC.INI.   These values are now the defaults when you connect to the data source.   You can change the defaults by configuring your data source again.   You can override the defaults by connecting to the data source using a connection string with alternate values.

Connecting to Paradox Using a Connection String

If your application requires a connection string to connect to a data source, you must specify the data source name that tells the driver which ODBC.INI section to use for the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in ODBC.INI.   These values are not written to ODBC.INI.

You can specify either long or short names in the connection string.   The connection string has the form:

DSN=*data_source_name*[;*attribute=value*[;*attribute=value*]...]

An example of a connection string for Paradox is

DSN=PARADOX TABLES;DB=C:\ODBC\EMP;PW=ABC,DEF,GHI

The following table gives the long and short names for each attribute, as well as a description.

The defaults listed in the table are initial defaults that apply when no value is specified in either the connection string or in the data source definition in ODBC.INI.   If you specified a value for the attribute when configuring the data source, that value is your default.

Attribute	Description
DataSourceName (DSN)	A string that identifies a Paradox data source configuration in ODBC.INI. Examples include "Accounting" or "Paradox-Serv1."
Database (DB)	The directory in which the Paradox files are stored.
FileOpenCache (FOC)	A value that determines the maximum number of unused table opens to cache. For example, when FileOpenCache=4, and a user opens and closes 4 tables, the tables are not actually closed. The driver keeps them open so that if another query uses one of these tables, the driver does not have to perform another open, which is expensive. The advantage of file open caching is increased performance. The disadvantage is that a user who tries to open the table exclusively may get a locking conflict even though no one appears to have the table open. The initial default is 0.
CreateType (CT)	CreateType={3 | 4}. The table version for Create Table statements. Specify CreateType=3 to create Paradox 3.0 or 3.5 tables, CreateType=4 for Paradox 4.0 or 4.5 tables (the initial default).
SortOrder (SO)	SortOrder={ASCII | INTL | NORDAN | NORDAN4 | SWEDFIN}. The initial default is ASCII. The order in which records are sorted in a Paradox index. This option is used for compatibility with international versions of Paradox. The options are . NORDAN4 is only for Paradox 4.0 and 4.5 tables.
LockToReadBlob (LTRB)	LockToReadBlob={0 | 1}. A value that affects only tables that have blob data types (BLOB, OLE, Graphic, Formatted Memo, and Memo). With LockToReadBlob=1, in shared environments, records with blob data types are locked while the data type is cloned to assure a consistent

	snapshot of the data type and record. Disabling this feature by setting LockToReadBlob=0 (the initial default) may improve performance.
	See the "LockToReadBlob" section that follows this table for more information on this attribute.
ReadConsistency (RC)	ReadConsistency={0	1}. A value that specifies whether tables are locked to assure that all records can be reread within a statement. ReadConsistency=1 implies the table is always locked to prevent other writes. This assures that a record can be reread within a statement. ReadConsistency=0, the initial default, disables this behavior if it was specified during data source configuration.
IntlSort (IS)	IntlSort={0	1}. A value that determines the order in which records are retrieved when you issue a Select statement. If IntlSort=1, the driver uses the international sort order as defined by your operating system. The sort is case-insensitive (*a* precedes *B*); the sorting of accented characters is also affected (check your operating system documentation). If IntlSort=0 (the initial default), the driver uses the ASCII sort order, where uppercase letters precede lowercase letters (*B* precedes *a*).
UltraSafeCommit (USF)	UltraSafeCommit={0	1}. A value that determines when the driver flushes its changes to disk. If UltraSafeCommit=1, the driver does this at each COMMIT. This decreases performance. The initial default is 0. This means that the driver flushes its changes to disk when the table is closed or when internal buffers are full. In this case, a machine "crash" before closing a table may cause recent changes to be lost.
Passwords (PW)	A password or list of passwords. You can add 1 to 50 passwords into the system using a comma-separated list of passwords. Passwords are case-sensitive. For example, Passwords=psw1, psw2, psw3.
DeferQueryEvaluation (DQ)	DeferQueryEvaluation={0	1}. A value that determines when a query is evaluated--after all records are read or each time a record is fetched.
	If DeferQueryEvaluation=0, the driver generates a result set when the first record is fetched. The driver reads all records, evaluates each one against the Where clause, and compiles a result set containing the records that satisfy the search criteria. This process slows performance when the first record is fetched, but activity performed on the result set after this point is much faster, since the result set has already been created. You do not see any additions, deletions, or

changes in the database that occur while working from this result set.

If DeferQueryEvaluation=1 (the default), the driver evaluates the query each time a record is fetched, and stops reading through the records when it finds one that matches the search criteria.   This setting avoids the slowdown while fetching the first record, but each fetch takes longer because of the evaluation taking place. The data you retrieve reflects the latest changes to the database.   However, a result set is still generated if the query is a Union of multiple Select statements, if it contains the Distinct keyword, or if it has an Order By or Group By clause.

ModifySQL (MS)	ModifySQL={0	1}. This attribute is provided for backward compatibility with earlier versions of Q+E products. It determines whether the driver modifies SQL statements to conform to ODBC specifications or passes the SQL statement directly to Paradox. Specify ModifySQL=1 to have the driver modify the SQL statement to conform to ODBC specifications. Specify ModifySQL=0 to have the driver understand SQL dialects found in earlier drivers. The default is 1.

LockToReadBlob

The LockToReadBlob connection string option determines whether the Paradox 4 driver locks records containing a blob data type.   LockToReadBlob affects only tables containing blob data types (that is, BLOB, OLE, Graphic, Formatted Memo, and Memo) for which you use record locking.

LockToReadBlob=1 ensures greater data integrity but tends to be slow with large Paradox tables.   With LockToReadBlob=1, when the driver reads a record containing a blob data type, it locks the record, duplicates (clones) the corresponding blob data type, and then unlocks the record.   This precaution ensures that the user receives a consistent snapshot of a record and its corresponding field.

With LockToReadBlob=0, the driver neither locks the record nor duplicates the blob data type.   This setting improves performance but causes an error when the following sequence occurs:

1   You read part of a record containing a blob data type.

2   Another user reads the same record and updates the blob data type of this record.

3   You attempt to read the blob data type to finish reading the record.

When this sequence occurs, the following error is displayed:

Error on input or output to file: Another user modified the BLOB.

If you receive this message, you must reexecute the query.

Data Types

The following table shows how the Paradox data types are mapped to the standard ODBC data types. These Paradox data types can be used in a Create Table statement.

Paradox	ODBC
Alphanumeric	SQL_VARCHAR
BLOB	SQL_LONGVARBINARY
Currency	SQL_DOUBLE
Date	SQL_DATE
Formatted Memo	SQL_LONGVARBINARY
Graphic	SQL_LONGVARBINARY
Memo	SQL_LONGVARCHAR
Number	SQL_DOUBLE
OLE	SQL_LONGVARBINARY
Short	SQL_SMALLINT

Note: BLOB, Formatted Memo, Graphic, Memo, and OLE data types are valid only for Paradox 4.0 or greater. None of these data types except Memo can be used in a Where clause or as key fields of an index. The Memo data type can be used in Where clauses, but only the first 512 characters are used for the comparison purposes.

Column Names

Column names in SQL statements can be up to 25 characters and are case-insensitive.   If a column name contains a special character, does not begin with a letter, or is a reserved word, surround it with the grave character (`) (ASCII 96).   For example,

SELECT `last name` FROM emp

Select Statement

You use a SQL Select statement to specify the columns and records to be read.   The Paradox 4 driver supports all of the Select statement clauses.

Password Protection

Paradox supports two types of passwords: master and auxiliary. The Paradox 4 driver supports master passwords only and can manage up to 50 passwords.

Paradox tables may be encrypted to provide limited access to users who do not know the password. The driver maintains a list of passwords for each connection. The driver can access only encrypted tables for which a password appears in this list. You can supply a password in three ways: by typing it in the Password dialog box (which appears when the driver does not have the password to open an encrypted table), by including it in a connection string (with the Passwords attribute), or by using the Add Password statement.

Paradox provides five statements that manage passwords for Paradox tables. These statements are specific to the Paradox driver:

ENCRYPT *filename* USING PASSWORD *password*

ADD PASSWORD *password*

DECRYPT *filename* USING PASSWORD *password*

REMOVE PASSWORD *password*

REMOVE ALL PASSWORDS

filename can be a simple filename or a full pathname. If a simple filename is given, the file must be in the directory specified with the Database connection string attribute. The .DB extension is not required.

password is a case-sensitive text string up to 15 characters in length, including blanks. If your password includes lower-case letters or nonalphanumeric characters, enclose it in single quotation marks (').

Encrypting a Paradox Table

The Encrypt statement associates a password with a table. For example,

ENCRYPT emp USING PASSWORD test

Accessing an Encrypted Paradox Table

To access an encrypted Paradox table, add the password to the list of passwords Paradox maintains for that connection. To do so, you can

- Issue an Add Password statement before you access the table. For example,

  ADD PASSWORD test
  SELECT * FROM emp

- Specify the passwords using the Passwords attribute at connection time.

If you do not add the password, the driver prompts you for it when you access the table.

Decrypting a Paradox Table

The Decrypt statement disassociates a password from a table. You no longer need to enter the password to open the table. For example,

DECRYPT emp USING PASSWORD test

Removing a Password from Paradox

The Remove Password statement removes a password from the list Paradox maintains for the connection. For example,

REMOVE PASSWORD test

Removing All Passwords from Paradox

The Remove All Passwords statement removes the list of passwords Paradox maintains.

If you remove a password from Paradox and do not decrypt the table, you must continue entering the password to open the table.

Index Files

An index is used to read records in sorted order and to improve performance when selecting records and joining tables.   Paradox indexes are stored in separate files and are either *primary* or *non-primary*.

Primary Index

A primary index is made up of one or more fields from the Paradox table.   The primary key fields of a primary index consist of one or more consecutive fields in the table, beginning with the first field in the table.   A table can have only one primary index.

Collectively, the primary key fields uniquely identify each record in the Paradox table.   Thus, no two records in a Paradox table can share the same values in their primary key fields.   Once a primary index is created for a Paradox table, the table's records are re-ordered based on the primary key fields.   At the time a primary index is created, if any records have matching primary key field values, Paradox deletes all but the first record.   Paradox creates this index as maintained; that is, if you modify, add, or delete records in the table, the primary index is updated automatically to reflect these changes.

A primary index is a single file with the same name as the table on which it is based but with a .PX extension.

To lock records, you must have a primary index.

Non-Primary Index

Unlike a primary key field of a primary index, a non-primary key field does not uniquely identify each record.   Thus, two or more records in a Paradox table can share the same value in their non-primary key field.   A Paradox table can have more than one non-primary index, as long as each one is based on different fields.

A non-primary index is defined by specifying one or more fields in the Paradox table that constitute the non-primary key fields.   It allows Paradox to sort each record in the table according to the value of the non-primary key fields.

There are two kinds of non-primary indexes: maintained and non-maintained.   The difference between the two is that a maintained index is automatically updated when the table is changed, whereas a non-maintained index is not.   Instead, a non-maintained index is tagged out-of-date and is updated when the index is used again.

You must have a primary index on a table before you create a maintained, non-primary index.   Although you can create a non-maintained index on a table that has no primary index, the non-maintained index is ignored.   The index also is ignored during statements that modify the table because the index is not updated and is therefore incorrect.   The only time a non-maintained index is used is when you issue a read-only Select statement on a locked table that has a primary index.

For Paradox 3.*x*, a single non-primary index consists of a pair of files with the same name as the table on which the non-primary index is based; one of these files has an .X*nn* extension while the other has a .Y*nn* extension (where the hexadecimal number *nn* corresponds to the field number of the non-primary key field for the non-primary index).

For Paradox 4.*x*, single-field non-primary indexes that are case sensitive have the same name as their associated table and are assigned file extensions .X01 through .XFF, depending on the number of the field on which the index is based.   Single-field non-primary indexes that are case insensitive and composite indexes have the same name as the table on which they are based.   They are assigned file extensions sequentially starting with .XG0 (with hexadecimal increments).

Create and Drop Index Statements

The Paradox driver supports SQL statements to create and delete indexes. The Create Index Primary statement is used to create primary indexes. The Create Index statement is used to create non-primary indexes. The Drop Index statement is used to delete indexes.

Create Index Primary Statement

The syntax for creating a primary index is

CREATE [UNIQUE] INDEX PRIMARY
ON *table_name* (*column* [,*column*...])

The UNIQUE keyword is optional; the index is unique whether or not you include this keyword.

table_name is the name of the table on which the index is to be based.

column is the name of a column that is included as a the key field for the index. The column list must contain one or more consecutive fields in the table, beginning with the first field in the table.

For example,

CREATE UNIQUE INDEX PRIMARY ON emp (emp_id)

Be careful when you create a primary key because any rows that have a primary key duplication are deleted when you execute the Create Index Primary statement.

Create Index Statement

The syntax for creating a non-primary index is

CREATE INDEX *index_name* [/NON_MAINTAINED] [/CASE_INSENSITIVE] ON *table_name* (*column* [, *column*...])

index_name identifies the index. If the name contains blanks or special characters, or does not begin with a letter, surround it with the grave character (`) (ASCII 96).

The NON_MAINTAINED switch makes the index non-maintained. The default is to create a maintained index.

The CASE_INSENSITIVE switch makes the index case-insensitive. The default is to create a case-sensitive index.

table_name is the name of the table on which the index is to be based.

column is the name of a column that is included as a the key field for the index.

Paradox 3.0 and 3.5 tables cannot have composite or case-insensitive indexes. When you create a non-primary index for Paradox 3.0 and 3.5 tables, follow these rules:
- Specify only one column name.
- Do not use the CASE_INSENSITIVE switch.
- Use the column name as the index name.

For example,

CREATE INDEX last_name ON emp (last_name)

Drop Index Statement

The syntax for dropping a primary index is

DROP INDEX *path_name*.PRIMARY

For example,

DROP INDEX emp.PRIMARY

The syntax for dropping a non-primary index is

DROP INDEX *path_name.index_name*

path_name is the name of the table from which the index is being dropped. The pathname can be either the fully qualified pathname or, if the table is specified with the Database attribute of the connection string,

a simple table name.

index_name is the name that was given to the index when it was created.   If the name contains blanks or special characters, or does not begin with a letter, surround it with the grave character (`) (ASCII 96). Use the column name as the index name when dropping indexes from Paradox 3.5 or 3.0 tables.

For example,

DROP INDEX emp.last_name

Transactions

The Paradox 4 driver supports transactions.   A transaction is a series of database changes that is treated as a single unit.   In applications that don't use transactions, the Paradox driver immediately executes Insert, Update, and Delete statements on the database tables and the changes are automatically committed when the SQL statement is executed.   There is no way to undo such changes.   In applications that use transactions, inserts, updates, and deletes are held until a COMMIT or ROLLBACK is specified.   A COMMIT saves the changes to the database file; a ROLLBACK discards the changes.

Transactions affect the removal of record locking.   All locks are removed when SQLTransact is called with the Commit or Rollback option to end the active transaction.

Isolation and Lock Levels Supported

Paradox supports isolation level 1 (read committed).   It supports record- and table-level locking.

ODBC Conformance Level

The Paradox 4 driver supports the Core, Level 1, and Level 2 API functions listed in [Supported ODBC Functions](#).   It also supports backward and random fetching in SQLExtendedFetch.

The driver supports the minimum SQL grammar.

Number of Connections and Statements Supported

The Paradox database system supports multiple connections and multiple statements per connection.

Paradox 5 Driver

The Paradox 5 driver supports Paradox 3.0, 3.5, 4.0, 4.5, and 5.0 tables.

The driver filename is QEIDP*nn*.DLL.

System Requirements

To use the Paradox 5 driver, you must install the Borland Database Engine, which conforms to the IDAPI programming interface.   If you have the most recent version of Paradox for Windows or dBASE for Windows, then you should already have the necessary files, which include the following:
- IDAPI01.DLL
- IDPDX01.DLL
- IDRL0009.DLL
- ILD01.DLL

If you attempt to configure a data source and you do not have these files on your path or in your Windows SYSTEM directory, the following message appears:

The setup routines for the INTERSOLV ParadoxFile (*.db) ODBC driver could not be loaded. You may be low on memory and need to quit a few applications.

Multiuser Access to Tables

You can query Paradox tables that reside in a shared directory on a network or that are to be shared among applications running on a local workstation.   If the tables are on a network server, multiple users can query these tables simultaneously.   To share Paradox tables among multiple users, the tables must be located in a shared directory on your network server.

Two connection attributes identify the Paradox database you are accessing: Database (database directory) and NetDir (network directory).   The Database setting specifies the directory of Paradox tables that is the database.   If the Database setting you specify is a shared network directory, then Paradox requires a NetDir specification as well.   This value identifies the directory containing the PARADOX.NET file that corresponds to the Database setting you have specified.

Every user who accesses the same shared directory of tables must set the NetDir value to point to the same PARADOX.NET file.   If your connection string does not specify a NetDir value, then Paradox uses the NetDir value specified in the Paradox section of the IDAPI configuration file.   This makes it important that the NetDir specification in each user's IDAPI configuration file be set correctly.

Whenever you open a Paradox table that another user opens at the same time, the consistency of the data becomes an issue if both individuals are updating the table.

Locking

The Paradox 5 driver has two locking levels: record locking and table locking.   Tables that have no primary key always have a prevent write lock placed on the table when the table is opened.   The table lock is escalated to a write lock when an operation that changes the table is attempted.

Tables that have primary keys use record locking.   The locking level is escalated from record locking to a table write lock if the transaction runs out of record locks.

Primary keys provide the greatest concurrency for tables that are accessed and modified by multiple users.

Note   If a table lock is placed on a Paradox table, the Paradox 5 driver prevents users from updating and deleting records but does not prevent them from inserting records into the locked table.

Configuring Data Sources

To configure a Paradox 5 data source, do the following:

1   Start the ODBC Administrator.   A list of data sources appears.

2   If you are configuring a new data source, click **Add**.   A list of installed drivers appears.   Select INTERSOLV ParadoxFile and click **OK**.

If you are configuring an existing data source, select the data source name and click **Setup**.

The setup dialog box appears.

3   Specify values as follows:

Data Source Name: A string that identifies this Paradox data source configuration in ODBC.INI. Examples include "Accounting" or "Paradox5-Serv1."

Description: An optional long description of a data source name.   For example, "My Accounting Database" or "Paradox Files on Server number 1."

Database Directory: The directory in which the Paradox files are stored.   If a directory is not specified, the current working directory is used.   You can also specify aliases that are defined in your IDAPI configuration file, if you have one.   To do this, enclose the alias name in colons.   For example, to use the alias MYDATA, specify ":MYDATA:"

The following values are optional:

Network Directory: The directory containing the PARADOX.NET file that corresponds to the database you have specified.   If the Paradox database you are using is shared on a network, then every user who accesses it must set this value to point to the same PARADOX.NET file.   If not set here, this value is determined by the NetDir setting in the Paradox section of the IDAPI configuration file.   If you are not sure how to set this value, contact your network administrator.

File Open Cache: A numeric value to specify the maximum number of unused file opens to cache. For example, the value 4 specifies that when a user opens and closes four tables, the tables are not actually closed.   The driver keeps them open so that if another query uses one of these tables, the driver does not have to perform another open, which is expensive.   The advantage of file open caching is increased performance.   The disadvantage is that a user who specifies file locking on open may get a locking conflict even though no one appears to have the file open.   The default is 0, which means no file open caching.

Create Type: The Paradox table version desired for any Create Table statements that you execute. You can specify 3, 4, 5, or leave the box blank and use the default, which is determined by the Level setting in the Paradox section of the IDAPI configuration file.   The numeric values map to the major revision numbers of the Paradox family of products.

International Sort: A setting to indicate the order in which records are retrieved when you issue a Select statement with an Order By clause.   Select this check box to use the international sort order as defined by your operating system.   International sort order is case-insensitive (*a* precedes *B*); the sorting of accented characters is also affected (see your operating system documentation).   Leave this box blank to use ASCII sort order.   ASCII sort order is case-sensitive, where uppercase letters precede lowercase letters (*B* precedes *a*).

Leave this box blank to use the ASCII sort order, where uppercase letters precede lower case letters (*B* precedes *a*).

4   Click **Translate** to perform a translation of your data from one character set to another. The Select Translator dialog box appears in which you can select a translator to peform the translation. INTERSOLV provides a translator named INTERSOLV OEM ANSI that translates your data from the IBM PC character set to the ANSI character set. Click **OK** to leave this dialog box and perform the translation.

The translators that are listed in this dialog box are determined by the values listed in the ODBC Translators section of your ODBCINST.INI file.

5   Click **OK** to write these values to ODBC.INI.   These values are now the defaults when you connect to the data source.   You can change the defaults by configuring your data source again.   You can override the defaults by connecting to the data source using a connection string with alternate values.

Connecting to Paradox Using a Connection String

If your application requires a connection string to connect to a data source, you must specify the data source name that tells the driver which ODBC.INI section to use for the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in ODBC.INI.   These values are not written to ODBC.INI.

You can specify either long or short names in the connection string.   The connection string has the form:

DSN=*data_source_name*[;*attribute=value*[;*attribute=value*]...]

An example of a connection string for Paradox is

DSN=PARADOX TABLES;DB=C:\ODBC\EMP;PW=ABC,DEF,GHI

The following table gives the long and short names for each attribute, as well as a description.

The defaults listed in the table are initial defaults that apply when no value is specified in either the connection string or in the data source definition in ODBC.INI.   If you specified a value for the attribute when configuring the data source, that value is your default.

Attribute	Description
DataSourceName (DSN)	A string that identifies a Paradox data source configuration in ODBC.INI. Examples include "Accounting" or "Paradox5-Serv1."
Database (DB)	The directory in which the Paradox files are stored.
	For this attribute, you can also specify aliases that are defined in your IDAPI configuration file, if you have one. To do this, enclose the alias name in colons. For example, to use the alias MYDATA, specify "Database=:MYDATA:"
NetDir (ND)	The directory containing the PARADOX.NET file that corresponds to the database you have specified. If the Paradox database you are using is shared on a network, then every user who accesses it must set this value to point to the same PARADOX.NET file. If not specified, this value is determined by the NetDir setting in the Paradox section of the IDAPI configuration file. If you are not sure how to set this value, contact your network administrator.
FileOpenCache (FOC)	The maximum number of unused table opens to cache. For example, when FileOpenCache=4, and a user opens and closes 4 tables, the tables are not actually closed. The driver keeps them open so that if another query uses one of these tables, the driver does not have to perform another open, which is expensive. The advantage of using file open caching is increased performance. The disadvantage is that a user who tries to open the table exclusively may get a locking conflict even though no one appears to have the table open. The initial default is 0.
CreateType (CT)	CreateType={3 | 4 5 | null (blank)}. The table version for Create Table statements. There are four valid values for this connection string: 3, 4,

	5 and null (blank). The numeric values map to the major revision numbers of the Paradox family of products. To override another CreateType setting chosen during data source configuration with the default create type determined by the Level setting in the Paradox section of the IDAPI configuration file, set CreateType= (null).
IntlSort (IS)	IntlSort={0 | 1}. A value that determines the order that records are retrieved when you issue a Select statement. If IntlSort=1, the driver uses the international sort order as defined by your operating system. The sort is case-insensitive (*a* precedes *B*); the sorting of accented characters is also affected (check your operating system documentation). If IntlSort=0 (the default), the driver uses the ASCII sort order, where uppercase letters precede lowercase letters (*B* precedes *a*).
UltraSafeCommit (USF)	UltraSafeCommit={0 | 1}. A value that determines when the driver flushes its changes to disk. If UltraSafeCommit=1, the driver does this at each COMMIT. This decreases performance. The default is 0. This means that the driver flushes its changes to disk when the table is closed or when internal buffers are full. In this case, a machine "crash" before closing a table may cause recent changes to be lost.
Passwords (PW)	A password or list of passwords. You can add 1 to 50 passwords into the system using a comma-separated list of passwords. Passwords are case-sensitive. For example, Passwords=psw1, psw2, psw3.
DeferQueryEvaluation (DQ)	DeferQueryEvaluation={0 | 1}. A value that determines when a query is evaluated--after all records are read or each time a record is fetched.
	If DeferQueryEvaluation=0, the driver generates a result set when the first record is fetched. The driver reads all records, evaluates each one against the Where clause, and compiles a result set containing the records that satisfy the search criteria. This process slows performance when the first record is fetched, but activity performed on the result set after this point is much faster, since the result set has already been created. You do not see any additions, deletions, or changes in the database that occur while working from this result set.
	If DeferQueryEvaluation=1 (the default), the driver evaluates the query each time a record is fetched, and stops reading through the records when it finds one that matches the search criteria. This setting avoids the slowdown while

fetching the first record, but each fetch takes longer because of the evaluation taking place. The data you retrieve reflects the latest changes to the database.   However, a result set is still generated if the query is a Union of multiple Select statements, if it contains the Distinct keyword, or if it has an Order By or Group By clause.

Data Types

The following table shows how the Paradox 5 data types are mapped to the standard ODBC data types. It also identifies the types supported by Paradox versions 3.*x* and 4.*x*.   These Paradox 5 data types can be used in a Create Table statement.

Paradox	ODBC	3.*x* Support	4.*x* Support
Alpha	SQL_VARCHAR	Yes	Yes
AutoIncrement	SQL_INTEGER	No	No
BCD	SQL_DECIMAL	No	No
Binary*	SQL_LONGVARBINARY	No	Yes
Bytes*	SQL_BINARY	No	No
Date	SQL_DATE	Yes	Yes
Formatted Memo*	SQL_LONGVARBINARY	No	Yes
Graphic*	SQL_LONGVARBINARY	No	Yes
Logical*	SQL_BIT	No	No
Long Integer	SQL_INTEGER	No	No
Memo*	SQL_LONGVARCHAR	No	Yes
Money	SQL_DOUBLE	Yes	Yes
Number	SQL_DOUBLE	Yes	Yes
OLE*	SQL_LONGVARBINARY	No	Yes
Short	SQL_SMALLINT	Yes	No
Time	SQL_TIME	No	No
TimeStamp	SQL_TIMESTAMP	No	No

* Cannot be used in an index.   Of these types, only Logical can be used in a Where clause.

Select Statement

You use a SQL Select statement to specify the columns and records to be read.   The Paradox 5 driver supports all of the Select statement clauses.   This section describes the information that is specific to the Paradox 5 driver, which is column names.

Column Names

Paradox column names are case-insensitive and their maximum length is 25 characters.   If a column name contains a special character, does not begin with a letter, or is a reserved word, surround it with the grave character (`) (ASCII 96).   For example,

SELECT `last name` FROM emp

Password Protection

Paradox supports two types of passwords: master and auxiliary.   The Paradox 5 driver supports master passwords only and can manage up to 50 passwords.

Paradox tables may be encrypted to provide limited access to users who do not know the password. The driver maintains a list of passwords for each connection.   The driver can access only encrypted tables for which a password appears in this list.   You can supply a password in three ways: by typing it in the Password dialog box (which appears when the driver does not have the password to open an encrypted table), by including it in a connection string (with the Passwords attribute), or by using the Add Password statement.

Paradox provides five statements that manage passwords for Paradox tables.   These statements are specific to the Paradox driver:

ENCRYPT *filename* USING PASSWORD *password*

ADD PASSWORD *password*

DECRYPT *filename* USING PASSWORD *password*

REMOVE PASSWORD *password*

REMOVE ALL PASSWORDS

filename can be a simple filename or a full pathname.   If a simple filename is given, the file must be in the directory specified with the Database connection string attribute.   The .DB extension is not required.

password is a case-sensitive text string up to 15 characters in length, including blanks.   If your password includes lower-case letters or nonalphanumeric characters, enclose it in single quotation marks (').

Encrypting a Paradox Table

The Encrypt statement associates a password with a table.   For example,

ENCRYPT emp USING PASSWORD test

Accessing an Encrypted Paradox Table

To access an encrypted Paradox table, add the password to the list of passwords Paradox maintains for that connection.   To do so, you can
- Issue an Add Password statement before you access the table.   For example,

   ADD PASSWORD test
   SELECT * FROM emp

- Specify the passwords using the Passwords attribute at connection time.

If you do not add the password, the driver prompts you for it when you access the table.

Decrypting a Paradox Table

The Decrypt statement disassociates a password from a table.   You no longer need to enter the password to open the table.   For example,

DECRYPT emp USING PASSWORD test

Removing a Password from Paradox

The Remove Password statement removes a password from the list Paradox maintains for the connection.   For example,

REMOVE PASSWORD test

Removing All Passwords from Paradox

The Remove All Passwords statement removes the list of passwords Paradox maintains.

If you remove a password from Paradox and do not decrypt the table, you must continue entering the password to open the table.

Index Files

An index is used to read records in sorted order and to improve performance when selecting records and joining tables. Paradox indexes are stored in separate files and are either *primary* or *non-primary*.

Primary Index

A primary index is made up of one or more fields from the Paradox table. The primary key fields of a primary index consist of one or more consecutive fields in the table, beginning with the first field in the table. A table can have only one primary index.

Collectively, the primary key fields uniquely identify each record in the Paradox table. Thus, no two records in a Paradox table can share the same values in their primary key fields. Once a primary index is created for a Paradox table, the table's records are re-ordered based on the primary key fields. At the time a primary index is created, if any records have matching primary key field values, Paradox deletes all but the first record. Paradox creates this index as maintained; that is, if you modify, add, or delete records in the table, the primary index is updated automatically to reflect these changes.

A primary index is a single file with the same name as the table on which it is based but with a .PX extension.

To lock records, you must have a primary index.

Non-Primary Index

Unlike a primary key field of a primary index, a non-primary key field does not uniquely identify each record. Thus, two or more records in a Paradox table can share the same value in their non-primary key field. A Paradox table can have more than one non-primary index, as long as each one is based on different fields.

A non-primary index is defined by specifying one or more fields in the Paradox table that constitute the non-primary key field. It allows Paradox to sort each record in the table according to the values of the non-primary key fields.

There are two kinds of non-primary indexes: maintained and non-maintained. The difference between the two is that a maintained index is automatically updated when the table is changed, whereas a non-maintained index is not. Instead, a non-maintained index is tagged out-of-date and is updated when the index is used again.

You must have a primary index on a table before you create a maintained, non-primary index.

The Paradox 5 driver uses non-maintained indexes only for read-only queries on locked tables. A primary index is not required for the non-maintained index to be used.

For Paradox 3.*x*, a single non-primary index consists of a pair of files with the same name as the table on which the non-primary index is based; one of these files has an .X*nn* extension while the other has a .Y*nn* extension (where the hexadecimal number *nn* corresponds to the field number of the non-primary key field for the non-primary index).

For Paradox 4.*x* and 5, single-field non-primary indexes that are case sensitive have the same name as their associated table and are assigned file extensions .X01 through .XFF, depending on the number of the field on which the index is based. Single-field non-primary indexes that are case insensitive and composite indexes have the same name as the table on which they are based. They are assigned file extensions sequentially starting with .XG0 (with hexadecimal increments).

Create and Drop Index Statements

The Paradox driver supports SQL statements to create and delete indexes.   The Create Index Primary statement is used to create primary indexes.   The Create Index statement is used to create non-primary indexes.   The Drop Index statement is used to delete indexes.

Create Index Primary Statement

The syntax for creating a primary index is

CREATE [UNIQUE] INDEX PRIMARY
                ON *table_name* (*column* [,*column*...])

The UNIQUE keyword is optional; the index is unique whether or not you include this keyword.

table_name is the name of the table on which the index is to be based.

column is the name of a column that is included as a the key field for the index.   The column list must contain one or more consecutive fields in the table, beginning with the first field in the table.

For example,

CREATE UNIQUE INDEX PRIMARY ON emp (emp_id)

Be careful when you create a primary key because any rows that have a primary key duplication are deleted when you execute the Create Index Primary statement.

Create Index Statement

The syntax for creating a non-primary index is

CREATE INDEX *index_name* [/NON_MAINTAINED] [/CASE_INSENSITIVE] ON *table_name* (*column* [, *column*...])

index_name identifies the index.   If the name contains blanks or special characters, or does not begin with a letter, surround it with the grave character (`) (ASCII 96).

The NON_MAINTAINED switch makes the index non-maintained.   The default is to create a maintained index.

The CASE_INSENSITIVE switch makes the index case-insensitive.   The default is to create a case-sensitive index.

table_name is the name of the table on which the index is to be based.

column is the name of a column that is included as a the key field for the index.

Paradox 3.0 and 3.5 tables cannot have composite or case-insensitive indexes.   When you create a non-primary index for Paradox 3.0 and 3.5 tables, follow these rules:
- Specify only one column name.
- Do not use the CASE_INSENSITIVE switch.
- Use the column name as the index name.

For example,

CREATE INDEX last_name ON emp (last_name)

Drop Index Statement

The syntax for dropping a primary index is

DROP INDEX *path_name*.PRIMARY

For example,

DROP INDEX emp.PRIMARY

The syntax for dropping a non-primary index is

DROP INDEX *path_name.index_name*

path_name is the name of the table from which the index is being dropped.   The pathname can be either the fully qualified pathname or, if the table is specified with the Database attribute of the connection string,

a simple table name.

index_name is the name that was given to the index when it was created.   If the name contains blanks or special characters, or does not begin with a letter, surround it with the grave character (`) (ASCII 96). Use the column name as the index name when dropping indexes from Paradox 3.5 or 3.0 tables.

For example,

DROP INDEX emp.last_name

Transactions

The Paradox 5 driver supports transactions.   A transaction is a series of database changes that is treated as a single unit.   In applications that don't use transactions, the Paradox 5 driver immediately executes Insert, Update, and Delete statements on the database tables and the changes are automatically committed when the SQL statement is executed.   There is no way to undo such changes.   In applications that use transactions, inserts, updates, and deletes are held until a COMMIT or ROLLBACK is specified.   A COMMIT saves the changes to the database file; a ROLLBACK discards the changes.

Transactions affect the removal of record locking.   All locks are removed when SQLTransact is called with the Commit or Rollback option to end the active transaction.

Isolation and Lock Levels Supported

Paradox 5 supports isolation levels 1 (read committed) and 3, (serializable).   It supports record- and table-level locking.

ODBC Conformance Level

The Paradox 5 driver supports the Core, Level 1, and Level 2 API functions listed in Supported ODBC Functions.   In addition, it supports the following Level 2 functions:
- SQLSetPos
- SQLPrimaryKeys

Number of Connections and Statements Supported

The Paradox database system supports multiple connections and multiple statements per connection.

PROGRESS Driver

The PROGRESS driver supports version 6 of the PROGRESS database system.

The driver filename is QEPRO*nn*.DLL.

System Requirements

Configuring Data Sources

Connecting to PROGRESS Using a Logon Dialog Box

Connecting to PROGRESS Using a Connection String

Data Types

Isolation and Lock Levels Supported

ODBC Conformance Level

Number of Connections and Statements Supported

System Requirements

You must have the PROGRESS Windows Runtime client software to access a local- or server-based PROGRESS database.

If you are using LAN Workplace TCP/IP, HP TCP/IP, or Wollongong TCP/IP, or if you are reinstalling a PROGRESS driver that uses the network protocols NetBIOS, SPX, or FTP TCP/IP, you must follow these steps after you install the driver but before you configure a data source:

1 Exit Windows.

2 Change to your install directory, which is C:\QEODBC by default.

3 Enter the following command.

C:> PROGRESS \WINDOWS\SYSTEM option

You must specify the Windows SYSTEM directory.   If your machine has a different name for the Windows directory, use that name.   The option you choose depends on the network protocol you are using.

If you have	*option* is
NetBIOS, SPX, FTP TCP/IP	WS
LAN Workplace TCP/IP	LAN
HP TCP/IP	HP
Wollongong TCP/IP	WOL

Configuring Data Sources

To configure a PROGRESS data source, do the following:

1   Start the ODBC Administrator.   A list of data sources appears.

2   If you are configuring a new data source, click **Add**.   A list of installed drivers appears.   Select INTERSOLV PROGRESS and click **OK**.

If you are configuring an existing data source, select the data source name and click **Setup**.

The setup dialog box appears.

3   Specify values as follows:

Data Source Name: A string that identifies this PROGRESS data source configuration in ODBC.INI. Examples include "Accounting" or "PROG-Serv1."

Description: An optional long description of a data source name.   For example, "My Accounting Database" or "PROGRESS on Server number 1."

Database Name: The name of the database to which you want to connect by default.

The following values are optional:

Logon Options: The startup options.   The format for specifying these options is

[-db *database*] [-u *userid*] [-p *password*] [-1]

Use the -1 option (note that this is the number 1) to access a local database.

Alternatively, you can specify these options in a parameter file.   In this case, you need not specify startup options here; instead, you specify the parameter filename in the Parameter File field.

Default User Name: The default user name used to connect to your PROGRESS database.   A user name is required only if security is enabled on your database.   Your ODBC application may override this value or you may override this value in the logon dialog box or connection string.

Parameter File: The name of the parameter file that contains the logon options.   See also the description of the Logon Options field.

4   Click **Translate** to perform a translation of your data from one character set to another. The Select Translator dialog box appears in which you can select a translator to peform the translation. INTERSOLV provides a translator named INTERSOLV OEM ANSI that translates your data from the IBM PC character set to the ANSI character set. Click **OK** to leave this dialog box and perform the translation.

The translators that are listed in this dialog box are determined by the values listed in the ODBC Translators section of your ODBCINST.INI file.

5   Click **OK** to write these values to ODBC.INI.   These values are now the defaults when you connect to the data source.   You can change the defaults by configuring your data source again.   You can override the defaults by connecting to the data source using a connection string with alternate values.

Connecting to PROGRESS Using a Logon Dialog Box

Some ODBC applications display a logon dialog box when you are connecting to a data source.   In these cases, the data source name has already been specified.

In the logon dialog box, do the following:

1   Type the name of the database to which you want to connect.

2   If required, type your user name.

3   If required, type your password.

   User name and password are optional parameters.   You can log on to the database without these parameters, but you may not be able to create, delete, or manipulate the data.

4   Click **Options** to display the PROGRESS Logon Options dialog box.

   If the options are stored in a parameter file, then type its name in the Parameter File box.   The options are read from that file.   To display the options from the file, move to the Logon Options box.

   To access more than one database at a time, you must specify all the database names.   The format is

   [-db *database*] [-u *userid*] [-p *password*] [-1]

   Use the -1 option (note that this is the number 1) to access a local database.

   If you change the options and click **OK**, the modified options are saved to the parameter file.   You can click **Cancel** and not save the changes to the file, but the options you specified will not be used for this connection.

5   Click **OK** to complete the logon and to update the values in ODBC.INI.

Connecting to PROGRESS Using a Connection String

If your application requires a connection string to connect to a data source, you must specify the data source name that tells the driver which ODBC.INI section to use for the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in ODBC.INI.   These values are not written to ODBC.INI.

You can specify either long or short names in the connection string.   The connection string has the form:

DSN=*data_source_name*[;*attribute=value*[;*attribute=value*]...]"

An example of a connection string for PROGRESS is

DSN=PROGRESS ON PION1;DB=PAYROLL;UID=JOHN;PWD=XYZZY;PF=OPTS2

The following table gives the long and short names for each attribute, as well as a description.

The defaults listed in the table are initial defaults that apply when no value is specified in either the connection string or in the data source definition in ODBC.INI.   If you specified a value for the attribute when configuring the data source, that value is your default.

Attribute	Description
DataSourceName (DSN)	A string that identifies a PROGRESS data source configuration in ODBC.INI. Examples include "Accounting" or "PROG-Serv1."
Database (DB)	The name of the database to which you want to connect.
Options (OPTS)	The startup options. The format is [-db *database*] [-u *userid*] [-p *password*] [-1] Use the -1 option (the number 1) to access a local database. See the PROGRESS manual *System Administration II* for available startup options. Alternatively, you can specify these options in a parameter file. In this case, you need not specify startup options here; instead, you specify the name of the parameter file using the PfName attribute.
LogonID (UID)	The default logon ID (user name) used to connect to your PROGRESS database. A logon ID is required only if security is enabled on your database. If so, contact your system administrator to get your logon ID. This ID is case-sensitive.
Password (PWD)	A case-sensitive password.
PfName (PF)	The name of parameter file containing the startup options.

Data Types

The PROGRESS data types are mapped to the standard ODBC data types as follows:

PROGRESS	ODBC
Character	SQL_CHAR
Date	SQL_DATE
Decimal	SQL_DECIMAL
Integer	SQL_INTEGER
Logical	SQL_BIT
Numeric	SQL_NUMERIC
Smallint	SQL_INTEGER

Isolation and Lock Levels Supported

PROGRESS supports isolation level 1 (read committed).   PROGRESS supports record-level locking.

ODBC Conformance Level

The PROGRESS driver supports the Core, Level 1, and Level 2 API functions listed in <u>Supported ODBC Functions</u>.   The driver supports the core SQL grammar.

Number of Connections and Statements Supported

The PROGRESS database system supports a single connection and multiple statements per connection.

Scalable SQL Driver

The Scalable SQL (formerly NetWare SQL) driver supports the Scalable SQL database system in the Windows environment.

The driver filename is QEXQL*nn*.DLL.

[System Requirements](#)

[Configuring Data Sources](#)

[Connecting to Scalable SQL Using a Logon Dialog Box](#)

[Connecting to Scalable SQL Using a Connection String](#)

[Data Types](#)

[Isolation and Lock Levels Supported](#)

[ODBC Conformance Level](#)

[Number of Connections and Statements Supported](#)

[For Programmers Only](#)

System Requirements

You must start the Scalable SQL Requester Utility, NSREQ.EXE, before running Windows.   NSREQ.EXE must have a minimum data size setting of 4096, which is the default value.   For example, to start NSREQ with a data size of 32K, enter the following at the DOS prompt:

NSREQ /D:32767

To gain access to Scalable SQL databases using a client system, you must have the following files: WXQLCALL.DLL and NSREQ.EXE.

Configuring Data Sources

To configure a Scalable SQL data source, do the following:

1   Start the ODBC Administrator.   A list of data sources appears.

2   If you are configuring a new data source, click **Add**.   A list of installed drivers appears.   Select INTERSOLV ScalableSQL and click **OK**.

      If you are configuring an existing data source, select the data source name and click Setup.

      The setup dialog box appears.

3   Specify values as follows:

      Data Source Name: A string that identifies this Scalable SQL data source configuration in ODBC.INI.   Examples include "Accounting" or "Scalable SQL-Serv1."

      Description: An optional long description of a data source name.   For example, "My Accounting Database" or "Scalable SQL on Server number 1."

      Data Dictionary Path: The directory or path of the data dictionary files you want to access.   Your ODBC application uses this path to connect to Scalable SQL and give you access to the data dictionary files in that directory.   This may be the same as your Data Files Path.

      Data Files Path: The directory or path of the data files you want to access.   Your ODBC application uses this path to connect to Scalable SQL and give you access to the data files in that directory.

      The following values are optional:

      Default User Name: The default user name used to connect to your Scalable SQL database.   This value is case-sensitive.   Your ODBC application may override this value or you may override this value in the Logon dialog box or connection string.   A user name is required only if security is enabled on your database.

      Array Size: The number of bytes the driver uses for fetching multiple rows.   Values can be 0 to 65536 bytes; the default is 4000.   Larger values increase throughput by reducing the number of times the driver fetches data across the network.   Smaller values increase response time, as there is less of a delay waiting for the server to transmit data.

      Blank Escape: Scalable SQL lets you specify blanks as part of table or field names.   The blank escape character is used to replace these blanks.   The default blank escape character is a carat (^). Other values are tilde (~) and underscore (_).

4   Click **Translate** to perform a translation of your data from one character set to another. The Select Translator dialog box appears in which you can select a translator to peform the translation. INTERSOLV provides a translator named INTERSOLV OEM ANSI that translates your data from the IBM PC character set to the ANSI character set. Click **OK** to leave this dialog box and perform the translation.

      The translators that are listed in this dialog box are determined by the values listed in the ODBC Translators section of your ODBCINST.INI file.

5   Click **OK** to write these values to ODBC.INI.   These values are now the defaults when you connect to the data source.   You can change the defaults by configuring your data source again.   You can override the defaults by connecting to the data source using a connection string with alternate values.

Connecting to Scalable SQL Using a Logon Dialog Box

Some ODBC applications display a logon dialog box when you are connecting to a data source.   In these cases, the data source name has already been specified.

In the logon dialog box, do the following:

1   Type the database name or the complete pathname of the directory containing your data files in the Data Files Path box.

2   Type the database name or the complete pathname of the directory containing your data dictionary files in the Data Dictionary Path box.

3   If security has been enabled on your Scalable SQL database, type your user name and password.

4   Click **OK** to complete the logon and to update the values in ODBC.INI.

Connecting to Scalable SQL Using a Connection String

If your application requires a connection string to connect to a data source, you must specify the data source name that tells the driver which ODBC.INI section to use for the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in ODBC.INI.   These values are not written to ODBC.INI.

You can specify either long or short names in the connection string.   The connection string has the form:

DSN=*data_source_name*[;*attribute=value*[;*attribute=value*]...]

An example of a connection string for Scalable SQL is

DSN=NETSQL TABLES;UID=JOHN;PWD=XYZZY

The following table gives the long and short names for each attribute, as well as a description.

The defaults listed in the table are initial defaults that apply when no value is specified in either the connection string or in the data source definition in ODBC.INI.   If you specified a value for the attribute when configuring the data source, that value is your default.

Attribute	Description
DataSourceName (DSN)	A string that identifies a Scalable SQL data source configuration in ODBC.INI. Examples include "Accounting" or "Scalable SQL-Serv1."
DataDictPath (DICT)	The directory or path of the data dictionary files that you want to access. Your ODBC application uses this path to connect to Scalable SQL and give you access to the data dictionary files in that directory. This value may be the same as the value of DataFilesPath.
DataFilesPath (FILES)	The directory or path of the data files you want to access. Your ODBC application uses this path to connect to Scalable SQL and give you access to the data files in that directory.
LogonID (UID)	The default logon ID (user name) used to connect to your Scalable SQL database. This value is case-sensitive. This setting is required only if security is enabled on your database system. If so, contact your system administrator to get your logon ID.
Password (PWD)	A case-sensitive password.
BlankEscape (BLANK)	Scalable SQL lets you specify blanks as part of table or field names. The blank escape character is used to replace these blanks. The default blank escape character is a carat (^). Other values are tilde (~) and underscore (_).
ArraySize (AS)	The number of bytes the driver uses for fetching multiple rows. Values can be 0 to 65536. The initial default is 4000. Larger values increase throughput by reducing the number of times the driver fetches data across the network. Smaller values increase response time, as there is less of a delay waiting for the server to transmit data.
ModifySQL (MS)	This attribute is provided for backward compatibility with earlier versions of Q+E products. It determines whether the driver modifies SQL statements to

conform to ODBC specifications or passes the SQL statement directly to Scalable SQL.   Specify ModifySQL=1 to have the driver modify the SQL statement to conform to ODBC specifications. Specify ModifySQL=0 to have the driver understand SQL dialects found in earlier drivers.   The default is 1.

Data Types

Scalable SQL data types are mapped to the standard ODBC data types as follows:

Scalable SQL	ODBC
Autoinc	SQL_SMALLINT
Autoinc(4)	SQL_INTEGER
Bfloat(4)	SQL_REAL
Bfloat(8)	SQL_DOUBLE
Bit	SQL_BIT
Character	SQL_CHAR
Date	SQL_DATE
Decimal	SQL_DECIMAL
Float(4)	SQL_REAL
Float(8)	SQL_DOUBLE
Int	SQL_SMALLINT
Int(1)	SQL_TINYINT
Int(4)	SQL_INTEGER
Logical	SQL_BIT
Logical(2)	SQL_BIT
Lstring	SQL_VARCHAR
Lvar	SQL_LONGVARCHAR
Money	SQL_DECIMAL
Note	SQL_LONGVARCHAR
Numeric	SQL_NUMERIC
Numericsts	SQL_NUMERIC
Time	SQL_TIME
Zstring	SQL_VARCHAR

Isolation and Lock Levels Supported

Scalable SQL supports isolation levels 1 (read committed, the default) and 3 (serializable).   Scalable SQL supports page-level locking.

ODBC Conformance Level

The Scalable SQL driver supports the Core, Level 1, and Level 2 API functions listed in <u>Supported ODBC Functions</u>.   In addition, the following Level 2 functions are supported:

- SQLBrowseConnect
- SQLProcedures

The driver supports the minimum SQL grammar.

Number of Connections and Statements Supported

The Scalable SQL database system supports single connections and multiple statements per connection.

For Programmers Only

Scalable SQL allows multiple applications to share a single connection to the database. To implement this functionality, your application must retrieve the session ID value assigned to the initial connection, and then pass that ID as part of the connection string from other applications that share the connection.

To do this, first establish a connection by logging on to Scalable SQL with the standard DataDictPath, DataFilesPath, LogonID, and Password values. Then, get the session ID assigned to that connection by calling SQLGetConnectOption with 1051 as the *fOption* argument.

To share the connection, pass its session ID in the connection strings of other applications. The attribute SessionID (SID) is provided for this purpose.

SQL Server Driver

The SQL Server driver supports the SQL Server database system available from Microsoft and Sybase, Inc.   It also supports the Sybase Net gateway to DB2.

The driver filename is QESS*nn*.DLL.

[System Requirements](#)

[Configuring Data Sources](#)

[Connecting to SQL Server Using a Logon Dialog Box](#)

[Connecting to SQL Server Using a Connection String](#)

[Data Types](#)

[Isolation and Lock Levels Supported](#)

[ODBC Conformance Level](#)

[Number of Connections and Statements Supported](#)

System Requirements

You must have the appropriate DB-Library and Net-Library installed to gain access to Microsoft SQL Server or Sybase SQL Server databases.

Your database must support catalog stored procedures.

The DB-Library for Windows is W3DBLIB.DLL.   The Net-Library you need depends on the network protocol used to connect to SQL Server.   For example, Named Pipes requires DBNMP3.DLL, FTP's TCP/IP requires WDBFTPTC.DLL, and Novell's LAN Workplace for DOS requires WDBNOVTC.DLL. Contact your Microsoft SQL Server or Sybase SQL Server vendor to obtain the appropriate DB-Library and Net-Library.

WDBPING is a Windows tool provided with net-libraries to test connectivity from your client workstation to the database server.   Use this tool to test your connection.

If you attempt to configure a data source and you do not have W3DBLIB.DLL on your path or in your Windows SYSTEM directory, the following message appears:

> The setup routines for the INTERSOLV SQL Server ODBC driver could not be loaded. You may be low on memory and need to quit a few applications.

Configuring Data Sources

To configure a SQL Server data source, do the following:

1   Start the ODBC Administrator.   A list of data sources appears.

2   If you are configuring a new data source, click **Add**.   A list of installed drivers appears.   Select INTERSOLV SQLServer and click **OK**.

If you are configuring an existing data source, select the data source name and click **Setup**.

The setup dialog box appears.

3   Specify values as follows:

Data Source Name: A string that identifies this SQL Server data source configuration in ODBC.INI.   Examples include "Accounting" or "SQL Server-Serv1."

Description: An optional long description of a data source name.   For example, "My Accounting Database" or "SQL Server on Server number 1."

Server Name: The name of the server that contains the desired database.

Database Name: The name of the database to which you want to connect by default.   If you do not specify a value, the default database defined by SQL Server is used.

The following values are optional:

Server List: A comma-separated list of servers that will appear in the logon dialog box.

Database List: The databases that will be available in the SQL Server Logon Options dialog box.   Separate the names with commas.

Default Logon ID: The default logon ID used to connect to your SQL Server database.   This ID is case-sensitive.   A logon ID is required only if security is enabled on your database.   Your ODBC application may override this value or you may override this value in the logon dialog box or connection string.

Language: The national language to be used by the client.   The default is English.

Application Name: The name SQL Server uses to identify your application.

Workstation ID: The workstation ID used by the client.

Cursor Cache Size: The number of cursors the cursor cache can hold.   The driver creates a cache of statements; each statement represents an open connection to SQL Server.   The cursor cache increases performance but uses database resources.   The default is 1 (one cursor).

Yield Proc: A numeric value that determines whether you can work in other applications when SQL Server is busy.   This attribute is useful to users of ODBC applications.   Valid values are

- 0 (peek and dispatch), which causes the driver to check the Windows message queue and send any messages to the appropriate Windows application.
- 1 (no yielding, the default), which does not let you work in other applications.
- 3 (dispatch via Windows Yield function), which turns control over to the Windows kernel.   The Windows kernel checks the message queue and sends any messages to the appropriate application window.

It is recommended that you use the value 1.

Character Conversion: This value controls the character set conversion between SQL Server (version 4.8 or later) and a client application.   If you omit this value, no character conversion takes place on your server.

Common values include iso-1 for ISO-8859-1, cp850 for Code Page 850, roman8 for Roman8 character set, and SJIS for a Japanese character set.   See your SQL Server documentation for a complete list of values.

Cancel Behavior: A value that specifies how a previously executed statement should be canceled. Valid values are

- 0 fetches all of the remaining records if the statement was a Select.
- 1 cancels the statement by calling dbcancel.   This is the default and should be used if dbcancel is supported in your client/server configuration.
- 2 closes the connection to the server for the statement.   Use this value only if dbcancel is not supported for your configuration and the performance of fetching all remaining records is unacceptable.

Using Gateway: Select this check box if you are using Sybase Net-Gateway to access a DB2 database with this data source.

NETAPI.DLL Library Available: The driver uses NETAPI.DLL to get the name of your workstation. Most major PC networks support this feature.   If your network supports this capability, select this option.   If you supply a workstation ID, this field is ignored.

Two-Phase Commit: This check box, when selected, enables you to have two active statements within a transaction, using the SQL Server two-phase commit services.   The active statements may deadlock if they reference the same SQL Server table.

4 Click **Translate** to perform a translation of your data from one character set to another. The Select Translator dialog box appears in which you can select a translator to peform the translation. INTERSOLV provides a translator named INTERSOLV OEM ANSI that translates your data from the IBM PC character set to the ANSI character set. Click **OK** to leave this dialog box and perform the translation.

The translators that are listed in this dialog box are determined by the values listed in the ODBC Translators section of your ODBCINST.INI file.

5 Click **OK** to write these values to ODBC.INI.   These values are now the defaults when you connect to the data source.   You can change the defaults by configuring your data source again.   You can override the defaults by connecting to the data source using a connection string with alternate values.

Connecting to SQL Server Using a Logon Dialog Box

Some ODBC applications display a logon dialog box when you are connecting to a data source.   In these cases, the data source name has already been specified.   For SQL Server, the dialog box is as follows:

In this dialog box, do the following:

1   Type the name of the server containing the SQL Server database tables you want to access (case-sensitive) or select the name from the Server Name drop-down list, which displays the server names you specified in the setup dialog box.

2   If required, type your case-sensitive login ID.

3   If required, type your case-sensitive password for the system.

4   Optionally, click **Options** to display the SQL Server Logon Options dialog box and specify the initial SQL Server database to connect to and the name of your workstation.

5   Click **OK** to log on to the SQL Server database installed on the server you specified and to update the values in ODBC.INI.

Connecting to SQL Server Using a Connection String

If your application requires a connection string to connect to a data source, you must specify the data source name that tells the driver which ODBC.INI section to use for the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in ODBC.INI.   These values are not written to ODBC.INI.

You can specify either long or short names in the connection string.   The connection string has the form:

DSN=*data_source_name*[;*attribute=value*[;*attribute=value*]...]

An example of a connection string for SQL Server is

DSN=Accounting;DB=PAYROLL;UID=JOHN;PWD=XYZZY

The following table gives the long and short names for each attribute, as well as a description.

The defaults listed in the table are initial defaults that apply when no value is specified in either the connection string or in the data source definition in ODBC.INI.   If you specified a value for the attribute when configuring the data source, that value is your default.

Attribute	Description
DataSourceName (DSN)	A string that identifies a SQL Server data source configuration in ODBC.INI. Examples include "Accounting" or "SQL Server-Serv1."
ServerName (SRVR)	The name of the server containing the SQL Server tables you want to access.
Database (DB)	The name of the database to which you want to connect.
LogonID (UID)	The case-sensitive logon ID used to connect to your SQL Server database. A logon ID is required only if security is enabled on your database. If so, contact your system administrator to get your logon ID.
Password (PWD)	A case-sensitive password.
Language (LANG)	The national language to be used by the client. The initial default is English.
ApplicationName (APP)	The name SQL Server uses to identify your application.
WorkstationID (WKID)	The workstation ID used by the client.
CursorCacheSize (CCS)	The number of cursors the cursor cache can hold. The driver creates a cache of statements; each statement represents an open connection to SQL Server. The cursor cache increases performance but uses database resources. The initial default is 1.
YieldProc (YLD)	YieldProc={0 | 1 | 3}. A numeric value that determines if you can work in other applications when SQL Server is busy. This attribute is useful to users of ODBC applications. Valid values are ▪ YieldProc=0 (peek and dispatch) causes the driver to check the Windows message queue and send any messages to the appropriate Windows application. ▪ YieldProc=1 (no yielding, the initial default) does not let you work in other applications.

	▪ YieldProc=3 (dispatch via Windows Yield function) turns control over to the Windows kernel. The Windows kernel checks the message queue and sends any messages to the appropriate application window.
	It is recommended that you use YieldProc=1. YieldProc=0 and YieldProc=3 do not work with all Windows applications.
CharConv (CC)	A value that controls the character set conversion between SQL Server (version 4.8 or later) and a client application. Common values include iso-1 for ISO-8859-1, cp850 for Code Page 850, roman8 for the Roman8 character set, and SJIS for a Japanese character set. See your SQL Server documentation for a complete list of values.
Cancel (CAN)	Cancel={0 | 1 | 2}. A value that specifies how a previously executed statement should be canceled. Valid values are ▪ Cancel=0 fetches all remaining records if the statement was a Select. ▪ Cancel=1 cancels the statement by calling dbcancel. Set Cancel=1 if dbcancel is supported in your client/server configuration. This is the initial default. ▪ Cancel=2 closes the connection to the server for the statement. Set Cancel=2 only if dbcancel is not supported for your configuration and the performance of fetching all remaining records is unacceptable.
Gateway (GW)	Gateway={0 | 1}. A value that specifies whether you are using the Sybase Net-Gateway to access a DB2 database with this data source. Set Gateway=1 if this is the case. Otherwise, set Gateway=0 (the initial default).
TwoPhaseCommit (TPC)	TwoPhaseCommit={0 | 1}. This attribute lets you have two or more active statements within a transaction, using the SQL Server two-phase commit services. Set TwoPhaseCommit=1 to use two-phase commit. The active statements may deadlock if they reference the same SQL Server table. Otherwise, set TwoPhaseCommit=0 (the initial default).
Netapi (NAPI)	Netapi={0 | 1}. A value that specifies whether NETAPI.DLL is available. Netapi=0, the initial default, indicates it is not available; Netapi=1 indicates it is available. If you supply a value for the WorkstationID attribute, this attribute is ignored.
ModifySQL (MS)	ModifySQL={0 | 1}. This attribute is provided for backward compatibility. It determines whether the driver modifies SQL statements to conform to ODBC specifications or passes the SQL statement directly to SQL Server. Specify ModifySQL=1 to have the

driver modify the SQL statement to conform to ODBC
specifications.   Specify ModifySQL=0 to have the
driver understand SQL dialects found in earlier
drivers.   The default is 1.

Data Types

The SQL Server data types are mapped to the standard ODBC data types as follows:

SQL Server	ODBC Data Type
binary	SQL_BINARY
bit	SQL_BIT
char	SQL_CHAR
datetime	SQL_TIMESTAMP
float	SQL_FLOAT
image	SQL_LONGVARBINARY
int	SQL_INTEGER
money	SQL_DECIMAL
real	SQL_REAL
smalldatetime	SQL_TIMESTAMP
smallint	SQL_SMALLINT
smallmoney	SQL_DECIMAL
sysname	SQL_VARCHAR
text	SQL_LONGVARCHAR
timestamp	SQL_VARBINARY
tinyint	SQL_TINYINT
varbinary	SQL_VARBINARY
varchar	SQL_VARCHAR

Isolation and Lock Levels Supported

SQL Server supports isolation levels 1 (read committed) and 3 (serializable). SQL Server supports page-level locking.

ODBC Conformance Level

The SQL Server driver supports the Core, Level 1, and Level 2 API functions listed in <u>Supported ODBC Functions</u>.   In addition, the following Level 2 functions are supported:

- SQLBrowseConnect
- SQLColumnPrivileges
- SQLForeignKeys
- SQLPrimaryKeys
- SQLProcedureColumns
- SQLProcedures
- SQLTablePrivileges

The driver supports the minimum SQL grammar.

Number of Connections and Statements Supported

The SQL Server database system supports multiple connections.   With two-phased commit, SQL Server supports multiple statements per connection.   Otherwise, SQL Server supports a single statement per connection if SQL_AUTOCOMMIT is 0 and multiple statements per connection if SQL_AUTOCOMMIT is 1.

SQLBase Driver

The SQLBase driver supports the Gupta SQLBase database system in the Windows and OS/2 environments.   It also supports Gupta's SQLHost gateway to DB2.

The driver filename is QEGUP*nn*.DLL.

[System Requirements](#)

[Configuring Data Sources](#)

[Connecting to InterBase Using a Logon Dialog Box](#)

[Connecting to InterBase Using a Connection String](#)

[Data Types](#)

[Isolation and Lock Levels Supported](#)

[ODBC Conformance Level](#)

[Number of Connections and Statements Supported](#)

System Requirements

The SQLBase driver requires the SQLBase SQLAPIW.DLL and one or more SQLBase communication files (such as DBWSERVR.EXE for local databases and SQLNBIOW.DLL for NetBIOS) to provide access to local and remote SQLBase databases.   The directory containing these files must be on your path.

If you attempt to configure a data source and you do not have SQLAPIW.DLL on your path or in your Windows SYSTEM directory, the following message appears:

The setup routines for the INTERSOLV SQLBase ODBC driver could not be loaded. You may be low on memory and need to quit a few applications.

Configuring Data Sources

To configure a SQLBase data source, do the following:

1   Start the ODBC Administrator.   A list of data sources appears.

2   If you are configuring a new data source, click **Add**.   A list of installed drivers appears.   Select INTERSOLV SQLBase and click **OK**.

If you are configuring an existing data source, select the data source name and click **Setup**.

The setup dialog box appears.

3   Specify values as follows:

Data Source Name: A string that identifies this SQLBase data source configuration in ODBC.INI. Examples include "Accounting" or "SQLBase-Serv1."

Description: An optional long description of a data source name.   For example, "My Accounting Database" or "SQLBase on Server number 1."

Database Name: The name of the database to which you want to connect by default.

The following values are optional:

Server Name: The name of the server that contains the desired database.   Specify the word LOCAL if you are using the local server.

The server name is not required to log on.   The logon dialog box appears more quickly if you do not specify a server name.   If you do specify a server name, the Database Name drop-down list in the logon dialog box is populated with the names of the databases available on that server.

Server List: A comma-separated list of servers that will appear in the Logon dialog box.   Specify LOCAL to add the local server to the list.

Default User Name: The default user name used to connect to your SQLBase database.   A user name is required only if security is enabled on your database.   Your ODBC application may override this value or you may override this value in the logon dialog box or connection string.

Cursor Cache Size: The number of cursors the cursor cache can hold.   The default is 6.

Yield Proc: A numeric value that determines whether you can work in other applications when SQLBase is busy.   This attribute is useful to users of ODBC applications.   Valid values are

- 0 (peek and dispatch), which causes the driver to check the Windows message queue and send any messages to the appropriate Windows application.
- 1 (no yielding, the default), which does not let you work in other applications.
- 2 (SQLBase's yield procedure), which uses SQLBase's default yield procedure.
- 3 (dispatch via Windows Yield function), which turns control over to the Windows kernel.   The Windows kernel checks the message queue and sends any messages to the appropriate application window.

It is recommended that you use the value 1.

Input Message Size: The number of bytes in the input message buffer.   The default is determined by SQLBase.   Increasing this value retrieves more records across the network in a single fetch.

Lock Time Out: The number of seconds SQLBase should wait for a lock to be freed before raising an error.   Values can be -1 (wait forever) to 1800; the default is 300.

No Recovery: Select this check box to disable transaction recovery.   Selecting this box is dangerous because your database can become inconsistent in the event of a system crash.

Release Plan: A number that determines whether a lock is maintained on a table when the cursors accessing the table are freed. Freeing the lock on the table results in a request to the server, which can decrease performance. No locks are freed when this option is set to the default, 0. Locks are

freed when this option is set to 1.

4 Click **Translate** to perform a translation of your data from one character set to another. The Select Translator dialog box appears in which you can select a translator to peform the translation. INTERSOLV provides a translator named INTERSOLV OEM ANSI that translates your data from the IBM PC character set to the ANSI character set. Click **OK** to leave this dialog box and perform the translation.

The translators that are listed in this dialog box are determined by the values listed in the ODBC Translators section of your ODBCINST.INI file.

5 Click **OK** to write these values to ODBC.INI. These values are now the defaults when you connect to the data source. You can change the defaults by configuring your data source again. You can override the defaults by connecting to the data source using a connection string with alternate values.

Connecting to SQLBase Using a Logon Dialog Box

Some ODBC applications display a logon dialog box when you are connecting to a data source.

In the logon dialog box, do the following:

1   Optionally, type the name of the server containing the SQLBase database tables you want to access or select the name from the Server Name drop-down list, which displays the server names you specified in the Setup dialog box.   Type the word LOCAL to access a local SQLBase database.

2   Type the name of the database you want to access.   If you specified a server name, you can select the name from the Database Name drop-down list.

3   If required, type your user name.

4   If required, type your password.

5   Click **OK** to complete the logon and to update the values in ODBC.INI.

Connecting to SQLBase Using a Connection String

If your application requires a connection string to connect to a data source, you must specify the data source name that tells the driver which ODBC.INI section to use for the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in ODBC.INI.   These values are not written to ODBC.INI.

You can specify either long or short names in the connection string.   The connection string has the form:

DSN=*data_source_name*[;*attribute=value*[;*attribute=value*]...]

An example of a connection string for SQLBase is

DSN=SQLBASE TABLES;SRVR=QESRVR;DB=PAYROLL;UID=JOHN;PWD=XYZZY

The following table gives the long and short names for each attribute, as well as a description.

The defaults listed in the table are initial defaults that apply when no value is specified in either the connection string or in the data source definition in ODBC.INI.   If you specified a value for the attribute when configuring the data source, that value is your default.

Attribute	Description
DataSourceName (DSN)	A string that identifies a SQLBase data source configuration in ODBC.INI. Examples include "Accounting" or "SQLBase-Serv1."
Database (DB)	The name of the database to which you want to connect.
LogonID (UID)	The default logon ID (user name) used to connect to your SQLBase database. If so, contact your system administrator to get your logon ID. A logon ID is required only if security is enabled on your database.
ServerName (SRVR)	The name of the server containing the SQLBase database tables you want to access. Specify ServerName=LOCAL if you are using the local server.
Servers (SRVRLIST)	A comma-separated list of servers with which to prompt the user in a Logon dialog box. Specify LOCAL to add the local server to the list.
Password (PWD)	A case-sensitive password.
CursorCacheSize (CCS)	The number of cursors the cursor cache can hold. The cursor cache increases performance and uses few database resources. The initial default is 6.
InputMessageSize (IMS)	The number of bytes of the input message buffer. Increasing this value retrieves more records across the network in a single fetch. The initial default is determined by SQLBase.
LockTimeOut (LTO)	The number of seconds SQLBase should wait for a lock to be freed before raising an error. Values can be -1 to 1800; -1 means wait forever. The initial default is 300.
DB2IsolationLevel (DIL)	DB2IsolationLevel={CS | RR}. The DB2 isolation level SQLHost is using, provided for use with SQLHost connections. Set DB2IsolationLevel=CS when DB2 is using the Cursor Stability isolation level; DB2IsolationLevel=RR (the default) when DB2 is

using Repeatable Read.

YieldProc (YLD)	YieldProc={0 | 1 | 2 | 3}. This option, which determines if you can work in other applications when SQLBase is busy. This attribute is useful to users of ODBC applications. Valid values are

- YieldProc=0 (peek and dispatch) causes the driver to check the Windows message queue and send any messages to the appropriate Windows application.
- YieldProc=1 (no yielding, the initial default) does not let you work in other applications.
- YieldProc=2 (SQLBase's yield procedure) uses SQLBase's default yield procedure.
- YieldProc=3 (dispatch via Windows Yield function) turns control over to the Windows kernel. The Windows kernel checks the message queue and sends any messages to the appropriate application window.

It is recommended that you use YieldProc=1. YieldProc=0, YieldProc=2, and YieldProc=3 do not work with all Windows applications.

NoRecovery (NR)	NoRecovery={0 | 1}. A value that enables or disables transaction recovery. NoRecovery=0 (the initial default) enables recovery; NoRecovery=1 disables recovery. NoRecovery=1 improves performance but is dangerous because your database can become inconsistent in the event of a system crash. See your SQLBase documentation for information on this option.
ReleasePlan (RP)	ReleasePlan={0 | 1}. A number that determines whether a lock is maintained on a table when the cursors accessing the table are freed. Freeing the lock on the table results in a request to the server, which can decrease performance.

When set to 0, the default, no locks on tables are freed. When set to 1, locks are freed.

Data Types

The SQLBase data types are mapped to the standard ODBC data types as follows:

SQLBase	ODBC
Char	SQL_VARCHAR
Date	SQL_DATE
Decimal	SQL_DECIMAL
Double Precision	SQL_DOUBLE
Integer	SQL_INTEGER
Long Varchar	SQL_LONGVARCHAR
Number	SQL_DOUBLE
Real	SQL_REAL
Smallint	SQL_SMALLINT
Time	SQL_TIME
Timestamp	SQL_TIMESTAMP
Varchar	SQL_VARCHAR

Note: The SQLBase Real data type is replaced by Float(21) when using DB2 via the SQLHost gateway. The Graphic, Vargraphic, and Long Vargraphic DB2 data types are not supported.

Isolation and Lock Levels Supported

SQLBase supports isolation levels 1 (read committed, the default) and 3 (serializable).   SQLBase also supports an alternative isolation level 1 called cursor stability.   Your ODBC application can use this isolation level by calling SQLSetConnectOption (1040,1).

SQLBase supports page-level locking.

ODBC Conformance Level

The SQLBase driver supports the Core, Level 1, and Level 2 API functions listed in Supported ODBC Functions.   In addition, the following Level 2 functions are supported:
- SQLBrowseConnect
- SQLColumnPrivileges
- SQLForeignKeys
- SQLPrimaryKeys
- SQLSetPos
- SQLTablePrivileges

The driver also supports backward and random fetching in SQLExtendedFetch.   It supports the core SQL grammar.

Number of Connections and Statements Supported

The SQLBase database system supports multiple connections and multiple statements per connection.

Sybase System 10 Driver

The Sybase System 10 driver supports the SQL Server System 10 database system from Sybase.

The driver filename is QESYB*nn*.DLL.

[System Requirements](#)

[Configuring Data Sources](#)

[Connecting to Sybase System 10 Using a Logon Dialog Box](#)

[Connecting to Sybase   System 10 Using a Connection String](#)

[Data Types](#)

[Isolation and Lock Levels Supported](#)

[ODBC Conformance Level](#)

[Number of Connections and Statements Supported](#)

System Requirements

You must install the Sybase Open Client-Library and the appropriate Sybase Net-Library to gain access to System 10.

WDBPING is a Windows tool that is provided with net-libraries to test connectivity from your client workstation to the database server.   Use this tool to test your connection.

Configuring Data Sources

To configure a System 10 data source, do the following:

1   Start the ODBC Administrator.   A list of data sources appears.

2   If you are configuring a new data source, click **Add**.   A list of installed drivers appears.   Select INTERSOLV Sybase System 10 and click **OK**.

If you are configuring an existing data source, select the data source name and click **Setup**.

The setup dialog box appears.

3   Specify values as follows:

Data Source Name: A string that identifies this Sybase System 10 data source configuration in ODBC.INI.   Examples include "Accounting" or "Sys10-Serv1."

Description: An optional long description of a data source name.   For example, "My Accounting Database" or "System 10 on Server number 1."

Server Name: The name of the server that contains the System 10 tables you want to access.   If not supplied, the server name in the DSQUERY environment variable is used.

The following values are optional:

Server List: The list of servers that appear in the logon dialog box.   Separate the server names with commas.

Database Name: The name of the database to which you want to connect by default.   If you do not specify a value, the default is the database defined by the   system administrator for each user.

Database List: The databases that appear in the logon dialog box.   Separate the names with commas.

Default Logon ID: The default logon ID used to connect to your System 10 database.   This ID is case-sensitive.   A logon ID is required only if security is enabled for the database you are connecting to.   Your ODBC application may override this value or you may override this value in the logon dialog box or connection string.

Interfaces File: The pathname of the interfaces file.   The default is the normal Sybase interfaces file.

Password Encryption: A number that determines whether password encryption can be performed from the Open Client Library to the server (PasswordEncryption=1). When set to 0, the default, this cannot be done.

Charset: The name of a character set corresponding to a subdirectory in $SYBASE/charsets.   The default is the setting on the System 10 server.

Workstation ID: The workstation ID used by the client.

Language: The national language corresponding to a subdirectory in $SYBASE/locales.   The default is English.

Application Name: The name used by System 10 to identify your application.

Yield Proc: A numeric value that determines whether you can work in other applications when Sybase System 10 is busy.   This attribute is useful to users of ODBC applications.   Valid values are

0 (peek and dispatch), which causes the driver to check the Windows message queue and send any messages to the appropriate Windows application.

1 (no yielding, the default), which does not let you work in other applications.

3 (dispatch via Windows Yield function), which turns control over to the Windows kernel.   The Windows kernel checks the message queue and sends any messages to the appropriate

application window.

It is recommended that you use the value 1.

Optimize Prepare: A value that determines whether stored procedures are created on the server for every call to SQLPrepare. When set to 0, stored procedures are created for every call to SQLPrepare, which can decrease performance.

When set to 1, the initial default, the driver creates stored procedures only if the statement contains parameters. Otherwise, the statement is cached and executed directly at SQLExecute time. When set to 2, the driver never creates stored procedures.

Array Size: The number of rows the driver retrieves from the server for each fetch.   This is not the number of rows given to the user.   The default is 10 rows.

Select Method: A value that determines whether database cursors are used for Select statements. When set to 0, the initial default, database cursors are used. In some cases performance degradation can occur when performing large numbers of sequential Select statements because of the amount of overhead associated with creating database cursors.

When set to 1, Select statements are executed directly without using database cursors. When set to 1, the data source is limited to one active statement and one active connection.

Packet Size: A number that determines the number of bytes per network packet transferred from the database server to the client. The correct setting of this attribute can improve performance.

When set to 0, the initial default, the driver uses the default packet size as specified in the System 10 server configuration.

When set to -1, the driver computes the maximum allowable packet size on the first connect to the data source and save the value in the odbc.ini file.

When set to x, an integer from 1 to 10, which indicates a multiple of 512 bytes (for example, Packet Size=6 means to set the packet size to 6 * 512 = 3072 bytes).

For you to take advantage of this connection attribute, you must configure the System 10 server for a maximum network packet size greater than or equal to the value you specified for Packet Size. For example,

```
sp_configure maximum network packet size, 5120
reconfigure
Restart System 10 Server
```

Note that the ODBC specification specifies a connect option, SQL_PACKET_SIZE, that offers this same functionality. To avoid conflicts with applications that may set both the connection string attribute and the ODBC connect option, they have been defined as mutually exclusive. If PacketSize is specified, you will receive a message Driver Not Capable if you attempt to call SQL_PACKET_SIZE. If you do not set Packet Size, then application calls to SQL_PACKET_SIZE are accepted by the driver.

4   Click **Translate** to perform a translation of your data from one character set to another. The Select Translator dialog box appears in which you can select a translator to peform the translation. INTERSOLV provides a translator named INTERSOLV OEM ANSI that translates your data from the IBM PC character set to the ANSI character set. Click **OK** to leave this dialog box and perform the translation.

The translators that are listed in this dialog box are determined by the values listed in the ODBC Translators section of your ODBCINST.INI file.

5   Click **OK** to write these values to ODBC.INI.   These values are now the defaults when you connect to the data source.   You can change the defaults by configuring your data source again.   You can override the defaults by connecting to the data source using a connection string with alternate values.

Connecting to System 10 Using a Logon Dialog Box

Some ODBC applications display a Logon dialog box when you are connecting to a data source.   In these cases, the data source name has already been specified.

In the logon dialog box, do the following:

1 Type the case-sensitive name of the server containing the System 10 database tables you want to access or select the name from the Server Name drop-down list, which displays the server names you specified in the setup dialog box.

2 If required, type your case-sensitive login ID.

3 If required, type your case-sensitive password for the system.

4 Type the name of the database you want to access (case-sensitive) or select the name from the Database drop-down list, which displays the names you specified in the setup dialog box.

5 Click **OK** to complete the logon and to update the values in ODBC.INI.

Connecting to System 10 Using a Connection String

If your application requires a connection string to connect to a data source, you must specify the data source name that tells the driver which ODBC.INI section to use for the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in ODBC.INI.   These values are not written to ODBC.INI.

You can specify either long or short names in the connection string.   The connection string has the form:

DSN=*data_source_name*[;*attribute=value*[;*attribute=value*]...]

An example of a connection string for Sybase System 10 is

DSN=SYS10 TABLES;SRVR=QESRVR;DB=PAYROLL;UID=JOHN;PWD=XYZZY

The following table gives the long and short names for each attribute, as well as a description.

The defaults listed in the table are initial defaults that apply when no value is specified in either the connection string or in the data source definition in ODBC.INI.   If you specified a value for the attribute when configuring the data source, that value is your default.

Attribute	Description
DataSourceName (DSN)	A string that identifies a single connection to a System 10 database. Examples include "Accounting" or "Sys10-Serv1."
ServerName (SRVR)	The name of the server containing the System 10 tables you want to access. If not supplied, the initial default is the server name in the DSQUERY environment variable.
LogonID (UID)	The default logon ID used to connect to your System 10 database. This ID is case-sensitive. A logon ID is required only if security is enabled on your database. If so, contact your system administrator to get your logon ID.
Password (PWD)	A case-sensitive password.
Database (DB)	The name of the database to which you want to connect.
Language (LANG)	The national language corresponding to a subdirectory in $SYBASE/locales.
Charset (CS)	The name of a character set corresponding to a subdirectory in $SYBASE/charsets.
WorkstationID (WKID)	The workstation ID used by the client.
ApplicationName (APP)	The name used by System 10 to identify your application.
InterfacesFile (IFILE)	The pathname to the interfaces file.
ArraySize (AS)	The number of rows the driver retrieves from the server for a fetch. This is not the number of rows given to the user. This increases performance by reducing network traffic. The initial default is 10 rows.
YieldProc (YLD)	YieldProc={0 | 1 | 3}. A numeric value that determines whether you can work in other applications when

Sybase System 10 is busy.   This attribute is useful to users of ODBC applications.   Valid values are

- YieldProc=0 (peek and dispatch) causes the driver to check the Windows message queue and to send any messages to the appropriate Windows application.
- YieldProc=1 (no yielding, the initial default) does not let you work in other applications.
- YieldProc=3 (dispatch via Windows Yield function) turns control over to the Windows kernel. The Windows kernel checks the message queue and sends any messages to the appropriate application window.

It is recommended that you use YieldProc=1. YieldProc=0 and YieldProc=3 do not work with all Windows applications.

OptimizePrepare (OP)	OptimizePrepare={0 | 1 | 2}. A value that determines whether stored procedures are created on the server for every call to SQLPrepare.
	When set to 0, stored procedures are created for every call to SQLPrepare. This setting can result in bad performance.
	When set to 1, the initial default, the driver creates stored procedures only if the statement contains parameters. Otherwise, the statement is cached and executed directly at SQLExecute time.
	When set to 2, the driver never creates stored procedures.
SelectMethod (SM)	SelectMethod={0 | 1}. A value that determines whether database cursors are used for Select statements. When set to 0, the initial default, database cursors are used. In some cases performance degradation can ocur when performing large numbers of sequential Select statements because of the amount of overhead associated with creating database cursors.
	When set to 1, Select statements are executed directly without using database cursors. When set to 1, the data source is limited to one active statement and one active connection.
Password Encryption (PE)	Password Encryption={0 | 1}. A number that determines whether password encryption can be performed from the Open Client Library to the server (PasswordEncryption=1). When set to 0, the default, this cannot be done.
PacketSize (PS)	PacketSize={-1 | 0 | x}. A number that determines the number of bytes per network packet transferred from the database server to the client. The correct setting of this attribute can improve performance.
	When set to 0, the initial default, the driver uses the

default packet size as specified in the System 10 server configuration.

When set to -1, the driver computes the maximum allowable packet size on the first connect to the data source and save the value in the odbc.ini file.

When set to x, an integer from 1 to 10, which indicates a multiple of 512 bytes (for example, PacketSize=6 means to set the packet size to 6 * 512 = 3072 bytes).

For you to take advantage of this connection attribute, you must configure the System 10 server for a maximum network packet size greater than or equal to the value you specified for PacketSize. For example,

```
sp_configure maximum network packet size, 5120
reconfigure
Restart System 10 Server
```

Note that the ODBC specification specifies a connect option, SQL_PACKET_SIZE, that offers this same functionality. To avoid conflicts with applications that may set both the connection string attribute and the ODBC connect option, they have been defined as mutually exclusive. If PacketSize is specified, you will receive a message Driver Not Capable if you attempt to call SQL_PACKET_SIZE. If you do not set PacketSize, then application calls to SQL_PACKET_SIZE are accepted by the driver.

Data Types

The System 10 data types are mapped to the standard ODBC data types as follows:

System 10	ODBC
binary	SQL_BINARY
bit	SQL_BIT
char	SQL_CHAR
datetime	SQL_TIMESTAMP
decimal	SQL_DECIMAL
float	SQL_FLOAT
image	SQL_LONGVARBINARY
int	SQL_INTEGER
money	SQL_DECIMAL
numeric	SQL_NUMERIC
real	SQL_REAL
smalldatetime	SQL_TIMESTAMP
smallint	SQL_SMALLINT
smallmoney	SQL_DECIMAL
sysname	SQL_VARCHAR
text	SQL_LONGVARCHAR
timestamp	SQL_VARBINARY
tinyint	SQL_TINYINT
varbinary	SQL_VARBINARY
varchar	SQL_VARCHAR

Note: The nchar, nvarchar, sensitivity, and sensitivity_boundary data types are not supported.

Isolation and Lock Levels Supported

System 10 supports isolation levels 1 (read committed, the default) and 3 (serializable).   It supports page-level locking.

ODBC Conformance Level

The Sybase System 10 driver supports the Core, Level 1, and Level 2 API functions listed in <u>Supported ODBC Functions</u>.   In addition, the following Level 2 functions are supported:

- SQLBrowseConnect
- SQLColumnPrivileges
- SQLForeignKeys
- SQLPrimaryKeys
- SQLProcedureColumns
- SQLProcedures
- SQLTablePrivileges

The driver supports the minimum SQL grammar.

Number of Connections and Statements Supported

The System 10 database system supports multiple connections and multiple statements per connection.

Text Driver

The Text driver supports ASCII text files.   These files can be printed directly or edited with text editors or word processors, since none of the data is stored in a binary format.

The driver filename is QETXT*nn*.DLL.

The Text driver executes SQL statements directly on the text files.   The driver supports Insert statements, and inserts the record at the end of the file.   However, you may not execute Update or Delete statements.

Users who are accessing the same text file must be running SHARE.EXE.   Otherwise, new records can be overwritten if more than one user inserts a new record at the same time.

Common Text File Formats

Defining Table Structure

Date Masks

Configuring Data Sources

Connecting to a Text Database Using a Connection String

Data Types

Select Statement

ODBC Conformance Level

Number of Connections and Statements Supported

Common Text File Formats

Some common formats for text files are listed in the following table:

Format	Description
Comma-separated values	Commas separate column values, and each line is a separate record. Column values can vary in length. These files often have the .CSV extension.
Tab-separated values	Tabs separate column values, and each line is a separate record. Column values can vary in length.
Character-separated values	Any printable character except single or double quotation marks can separate column values, and each line is a separate record. Column values can vary in length.
Fixed	No character separates column values. Instead, values start at the same position and have the same length in each line. The values appear in fixed columns if you display the file. Each line is a separate record.

Comma-, tab-, and character-separated files are called character-delimited files, since values are separated by a special character.

Configuring Data Sources

To configure a Text data source, do the following:

1   Start the ODBC Administrator.   A list of data sources appears.

2   If you are configuring a new data source, click **Add**.   A list of installed drivers appears.   Select INTERSOLV TextFile and click **OK**.

If you are configuring an existing data source, select the data source name and click **Setup** (if you are using Apple's ODBC Driver Manager on the Macintosh, this button is called **Modify**).

The setup dialog box appears.

3   Specify values as follows:

Data Source Name: A string that identifies this Text data source configuration in ODBC.INI. Examples include "Accounting" or "Text Files."

Description: An optional long description of a data source name.   For example, "My Accounting Files" or "My Text Files in the Accounting Directory."

Database Directory: The directory in which the text files are stored.   If none is specified, the current working directory is used.

The following values are optional:

Rows to Scan: The number of rows in a text file that the driver scans to determine the data types in the file.   If the value is 0,   all rows in the file are scanned.   The default is 25.

Default Table Type: The type of text file: comma-separated, tab-separated, character-separated, or fixed length.   This value tells the driver the default type, which is used when creating a new table and opening an undefined table.

Delimiter Character: The character used as a delimiter for character-separated files.   It can be any printable character except a single or double quote.   The initial default is a comma (,).

Action for Undefined Tables: Two radio buttons that indicate what action the driver should take when it encounters a file that has not been defined.   Set the Prompt for Definition radio button, if you want the driver to prompt the user when it encounters a file whose format is not defined.   Otherwise, set the Guess Definition radio button; in this case, the driver guesses the file's format.

Return Additional Tables: Select this check box to tell the driver to return files you have defined in functions like SQLTables, SQLColumns, etc.   *and* files with a given extension.   In Extension List, specify a comma-separated list of the extensions.   To have files with no extensions returned, specify NONE.   For example, if some of your files have the extensions TXT and CSV and others have no extension, specify TXT,CSV,NONE.

By default, when an application requests a list of tables, only files that have been defined are returned.

File Open Cache: A numeric value to specify the maximum number of unused file opens to cache. For example, the value 4 specifies that when a user opens and closes four tables, the tables are not actually closed.   The driver keeps them open so that if another query uses one of these tables, the driver does not have to perform another open, which is expensive.   The advantage of file open caching is increased performance.   The disadvantage is that a user who specifies file locking on open may get a locking conflict even though no one appears to have the file open.   The default is 0, which means no file open caching.

Cache Size: The number of 64K blocks the driver uses to cache database records.   The higher the number, the better the performance.   The maximum number of blocks you can set depends on the system memory available.   If the cache size is greater than 0, when browsing backwards, you will not be able to see updates made by other users until you reexecute the Select statement. The default is 4.

Column Names in First Line: Select this check box to tell the driver to look for column names in the first line of the file.

International Sort: A setting to indicate the order in which records are retrieved when you issue a Select statement with an Order By clause.   Select this check box to use the international sort order as defined by your operating system.   International sort order is case-insensitive (*a* precedes *B*); the sorting of accented characters is also affected (see your operating system documentation).   Leave this box blank to use the ASCII sort order.   ASCII sort order is case-sensitive, where uppercase letter precede lowercase letters (*B* precedes *a*).

Note: The Rows to Scan, Default Table Type, Delimiter Character, and Column Names in First Line settings apply only to tables *not* previously defined.   These fields also determine the attributes of new tables created with the Create Table statement.

4   Click **Translate** to perform a translation of your data from one character set to another. The Select Translator dialog box appears in which you can select a translator to peform the translation. INTERSOLV provides a translator named INTERSOLV OEM ANSI that translates your data from the IBM PC character set to the ANSI character set. Click **OK** to leave this dialog box and perform the translation.

The translators that are listed in this dialog box are determined by the values listed in the ODBC Translators section of your ODBCINST.INI file.

5   Click **OK** to write these values to ODBC.INI.   These values are now the defaults when you connect to the data source.   You can change the defaults by configuring your data source again.   You can override the defaults by connecting to the data source using a connection string with alternate values.

Click **Define** in this dialog box to define the structure of your text files.   Defining Table Structure discusses how to define the column names and data types in a table.

Defining Table Structure

Since text files do not all have the same structure, the driver provides the option of defining the structure of an existing file.   Although defining the structure is not mandatory, since the driver can attempt to guess the names and types of the columns, this feature is extremely useful.

Define the structure of a file as follows:

1   Click **Define** in the ODBC Text Driver Setup dialog box, which you can access through the ODBC Administrator.   The Define File dialog box appears.

2   Select the correct file and click **OK** to define the file's structure using the Define Table dialog box.

In the Define Table dialog box, Database Name and File display the name of the database directory that contains the file and the name of the file you previously selected, respectively.

3   In the Table Information section of this dialog box, specify information about the overall structure of the file:

   a   In the Table box, type a table name.   Values may be up to 32 characters in length and may not be the same as another defined table in the database.   This name is returned by SQLTables.   By default, it is the filename without its extension.

   b   Select the Column Names in First Line check box if the first line of the file contains column names.   Otherwise, do not select this check box.

   c   Open the Table Type box and select a table type: comma, tab, fixed, or character.

   d   If the table type is character-separated, type the character that separates the values in the Delimiter Character box.

4   In the Column Information section, define the types and names of the columns in the table.   The box in the upper-left corner of this section lists the defined columns.

If you specified a comma-separated, tab-separated, or character-separated table type, the **Guess** button appears in this section.   Click **Guess** to tell the driver to guess the fields.   The driver then displays what it thinks the fields are.   You can then modify the field definitions by specifying values in the Name, Type, Mask, Precision, and Scale boxes (as appropriate) and clicking **Modify**.   If you do not want the driver to guess the fields, take the following steps to define the fields:

   a   In the Name box, type the name of the field.

   b   Open the Type drop-down list and select the data type of the field.   If the field type is date, then you must select a date mask for the field or type one in.   See the following section, "Date Masks," for more information.

   c   In the Precision box, type the precision of the field.

   d   In the Scale box, type the scale of the field.

     The precision and scale values determine how numeric data is to be returned.

   e   If you specified a fixed-length table type, you may specify the length and offset in the Length and Offset boxes, or you can specify a parse string.   The length is the number of bytes the data takes up in storage; the offset is the number of bytes from the start of the table to the start of the field.

   f   Click **Add** to add this field definition to the list box.

To modify the selected field in the list box, click **Modify**.

To remove the selected field in the list box, click **Remove**.

If you specified a fixed-length table type, the **Parse** button is displayed.   If you have not entered values in the length and offset boxes, click **Parse** to define the columns of the table.   The Parse Table dialog box appears.

This dialog box displays the first line of the file.   You must mark where each field begins and ends by enclosing it in brackets.   These brackets indicate the position and length of each field value in the record.   You must do this for all the fields in the table.   Click **OK** when you are done.

5   Click **OK** to define the table.

Date Masks

Date masks tell the driver how a date is stored in a text file. When a value is inserted into a text file, the date is formatted so that it matches the mask. When reading a text file, the driver converts the formatted date into a date data type.

The following table lists the symbols to use when specifying the date mask:

Symbol	Description
m	Output the month's number (1-12).
mm	Output a leading zero if the month number is less than 10.
mmm, Mmm, MMM	Output the three-letter abbreviation for the month depending on the case of the M's (that is, jan, Jan, JAN).
mmmm, Mmmm, MMMM	Output the full month name depending on the case of the M's (that is, january, January, JANUARY).
d	Output the day number (1-31).
dd	Output a leading zero if the day number is less than 10.
ddd, Ddd, DDD	Output the three-letter day abbreviation depending on the case of the D's (that is, mon, Mon, MON).
dddd, Dddd, DDDD	Output the day depending on the case of the D's (that is, monday, Monday, MONDAY).
yy	Output the last two digits of the year.
yyyy	Output the full four digits of the year.
J	Output the Julian value for the date. The Julian value is the number of days since 4712 BC.
\ - . : , (space)	Special characters used to separate the parts of a date.
\	Output the next character. For example, if the mask is mm/dd/yyyy \A\D, the value appears as 10/01/1993 AD in the text file.
"string", 'string'	Output the string in the text file.

The following table shows some example date values, masks, and how the date appears in the text file.

Date	Mask	Value
1993-10-01	yyyy-mm-dd	1993-10-01
	m/d/yy	10/1/93
	Ddd, Mmm dd, yyyy	Fri, Oct 01, 1993

Connecting to a Text Database Using a Connection String

If your application requires a connection string to connect to a data source, you must specify the data source name that tells the driver which ODBC.INI section to use for the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in ODBC.INI.   These values are not written to ODBC.INI.

You can specify either long or short names in the connection string.   The connection string has the form:

DSN=*data_source_name*[;*attribute=value*[;*attribute=value*]...]

An example of a connection string for text files is

DSN=TEXT FILES;TT=CHARACTER;DC=&

The following table gives the long and short names for each attribute, as well as a description.

The defaults listed in the table are initial defaults that apply when no value is specified in either the connection string or in the data source definition in ODBC.INI.   If you specified a value for the attribute when configuring the data source, that value is your default.

Attribute	Description
DataSourceName (DSN)	A string that identifies a Text data source configuration in ODBC.INI. Examples include "Accounting" or "Text Files."
Database (DB)	The directory in which the text files are stored.
ScanRows (SR)	The number of rows in a text file that the driver scans to determine the column types in the file. If the value is 0, all rows in the file are scanned. The initial default is 25.
TableType (TT)	TableType={Comma	Tab	Character	Fixed}.The Text driver supports four types: comma-separated, tab-separated, character-separated, and fixed length. Setting this value tells the driver the default type, which is used when creating a new table and opening an undefined table.
Delimiter (DC)	The character used as a delimiter for character-separated files. It can be any printable character. The initial default is a comma (,).
UndefinedTable (UT)	The Text driver can perform two operations when it encounters a file that has not been defined. UndefinedTable=PROMPT tells the driver to display a dialog box that allows the user to describe the file's format. UndefinedTable=GUESS tells the driver to guess the file's format. This is the initial default.
ExtraExtensions (EE)	A list of additional filename extensions to be recognized as text tables. When an application requests a list of tables, only files that have been defined are returned. To have the driver also return names of undefined files, specify a comma-separated list of file extensions. To specify files with no extension, use the keyword NONE.
FileOpenCache (FOC)	The maximum number of unused file opens to cache. For example, when FileOpenCache=4, and a user opens and closes four files, the files are

	not actually closed. The driver keeps them open so that if another query uses one of these files, the driver does not have to perform another open, which is expensive. The advantage of using file open caching is increased performance. The disadvantage is that a user who tries to open the file exclusively may get a locking conflict even though no one appears to have the file open. The initial default is 0.
CacheSize (CSZ)	The number of 64K blocks the driver uses to cache database records. The higher the number, the better the performance. The maximum number of blocks you can set depends on the system memory available. If the cache size is greater than 0, when browsing backwards, you will not be able to see updates made by other users until you reexecute the Select statement. The default is 4.
FirstLineNames (FLN)	FirstLineNames={0 | 1}. A value that determines whether the driver looks for column names in the first line of the file. If FirstLineNames=1, the driver looks for column names in the first line of the file. If FirstLineNames=0 (the initial default), the first line is interpreted as the first record in the file.
IntlSort (IS)	IntlSort={0 | 1}. A value that determines the order that records are retrieved when you issue a Select statement with an Order By clause. If IntlSort=1, the driver uses the international sort order as defined by your operating system. The sort is case-insensitive (*a* precedes *B*); the sorting of accented characters is also affected (see your operating system documentation). If IntlSort=0 (the initial default), the driver uses the ASCII sort order, where uppercase letters precede lowercase letters (*B* precedes *a*).

Note: The ScanRows, TableType, Delimiter, and FirstLineNames attributes apply to tables that have *not* been defined.   These attributes also determine the characteristics of new tables created with the Create Table statement.

Data Types

The text file data types are mapped to the standard ODBC data types as follows:

Text	ODBC
Numeric	SQL_NUMERIC
Date	SQL_DATE
Varchar	SQL_VARCHAR

Select Statement

You use the SQL Select statement to specify the columns and records to be read.   The driver supports all Select statement clauses as described in SQL for Flat-File Drivers.

ODBC Conformance Level

The Text driver supports the Core, Level 1, and Level 2 API functions listed in <u>Supported ODBC Functions</u>.   It also supports backward and random fetching in SQLExtendedFetch.

The driver supports the minimum SQL grammar.

Number of Connections and Statements Supported

Text files support multiple connections and multiple statements per connection.

XDB Driver

The XDB driver supports the XDB database system.

The driver filename is QEXDB*nn*.DLL.

[System Requirements](#)

[Configuring Data Sources](#)

[Connecting to XDB Using a Logon Dialog Box](#)

[Connecting to XDB Using a Connection String](#)

[Data Types](#)

[Isolation and Lock Levels Supported](#)

[ODBC Conformance Level](#)

[Number of Connections and Statements Supported](#)

System Requirements

The XDB driver requires that you have
- XDB version 2.41 or higher.
- A protocol supported by XDB (such as NetBIOS) if you are using the client-server version of XDB. If you are using the local version of XDB, a protocol is not required.
- XDB client software to access the server.

To have access to network DLLs, you must run the CLIENT.BAT file supplied with the XDB server. This file must be run on each client.

Modify the [XDB] section of your WIN.INI file. For example, if XDB is installed on drive C, modify WIN.INI as follows:

xdbcfg=C:\XDB
xsrvcfg=C:\XDB

These settings must also be defined as environment variables in your AUTOEXEC.BAT file. For example,

SET xdbcfg=C:\XDB
SET xsrvcfg=C:\XDB

If you are using the local version of XDB, you must also define XDBSERVE as an environment variable:

SET xdbserve=local

If you attempt to configure a data source and you do not have XUTILW.DLL, MEMRESW.DLL, MEMR30WD.DLL, and WINROUTE.DLL on your path or in your Windows SYSTEM directory, the following message appears:

The setup routines for the INTERSOLV XDB ODBC driver could not be loaded. You may be low on memory and need to quit a few applications.

When setting up your User ID in XDB user administration, the user type must be set to install user.

Configuring Data Sources

To configure an XDB data source, do the following:

1   Start the ODBC Administrator.   A list of data sources appears.

2   If you are configuring a new data source, click **Add**.   A list of installed drivers appears.   Select INTERSOLV XDB and click **OK**.

    If you are configuring an existing data source, select the data source name and click Setup.

    The setup dialog box appears.

3   Specify values as follows:

    Data Source Name: A string that identifies this XDB data source configuration in ODBC.INI. Examples include "Accounting" or "XDB-Serv1."

    Description: An optional long description of a data source name.   For example, "My Accounting Files" or "XDB Database on Server number 1."

    Server Name: The name of the server containing the XDB database tables you want to access. You can leave this empty if you are using the local server.

    The following values are optional:

    Database Name: The name of the database to which you want to connect by default.

    Default Authorization ID: The default authorization ID used to connect to your XDB database.   An authorization ID is required only if security is enabled on your database.   Your ODBC application may override this value or you may override this value in the logon dialog box or connection string.

    Server List: The list of servers that appear in the logon dialog box.   Separate the server names with commas.

4   Click **Translate** to perform a translation of your data from one character set to another. The Select Translator dialog box appears in which you can select a translator to peform the translation. INTERSOLV provides a translator named INTERSOLV OEM ANSI that translates your data from the IBM PC character set to the ANSI character set. Click **OK** to leave this dialog box and perform the translation.

    The translators that are listed in this dialog box are determined by the values listed in the ODBC Translators section of your ODBCINST.INI file.

5   Click **OK** to write these values to ODBC.INI.   These values are now the defaults when you connect to the data source.   You can change the defaults by configuring your data source again.   You can override the defaults by connecting to the data source using a connection string with alternate values.

Connecting to XDB Using a Logon Dialog Box

Some ODBC applications display a logon dialog box when you are connecting to a data source.   In these cases, the data source name has already been specified.

In the logon dialog box, do the following:

1   Type the server name of the computer containing the XDB database tables you want to access or select the name from the Server Name drop-down list, which displays the server names you specified in the setup dialog box.   Server names contain the information necessary for an XDB network redirector to locate the database to be used.

2   Type the name of the XDB database containing the tables you want to access.

3   If required, type your user name.

4   If required, type your password.

5   Click **OK** to log on to the XDB database on the computer you specified and to update the values in ODBC.INI.

Connecting to XDB Using a Connection String

If your application requires a connection string to connect to a data source, you must specify the data source name that tells the driver which ODBC.INI section to use for the default connection information. Optionally, you may specify *attribute=value* pairs in the connection string to override the default values stored in ODBC.INI.   These values are not written to ODBC.INI.

You can specify either long or short names in the connection string.   The connection string has the form

DSN=*data_source_name*[;*attribute=value*[;*attribute=value*]...]

An example of a connection string for XDB is

DSN=XDB TABLES;SRVR=QESRVR;DB=PAYROLL;UID=JOHN;PWD=XYZZY

The following table gives the long and short names for each attribute, as well as a description.

Attribute	Description
DataSourceName (DSN)	A string that identifies an XDB data source configuration in ODBC.INI. Examples include "Accounting" or "XDB-Serv1."
LogonID (UID)	The default logon ID (authorization ID) used to connect to your XDB database. A logon ID is required only if security is enabled on your database. If so, contact your system administrator to get your logon ID.
Password (PWD)	Your password.
ServerName (SRVR)	The name of the server containing the XDB database tables you want to access.
Database (DB)	The name of the database to which you want to connect.

Data Types

The XDB data types are mapped to the standard ODBC data types as follows:

XDB	ODBC
Char	SQL_CHAR
Date	SQL_DATE
Decimal	SQL_DECIMAL
Float	SQL_DOUBLE
Int	SQL_INTEGER
Long Varchar	SQL_LONGVARCHAR
Money	SQL_DECIMAL
Smallint	SQL_SMALLINT
Time	SQL_TIME
Timestamp	SQL_TIMESTAMP
Varchar	SQL_VARCHAR

Isolation and Lock Levels Supported

XDB supports isolation levels 0 (read uncommitted), 1 (read committed, the default), and 2 (repeatable read).   XDB supports record-level locking.

ODBC Conformance Level

The XDB driver supports the Core, Level 1, and Level 2 API functions listed in [Supported ODBC Functions](#). It also supports the Level 2 API function SQLBrowseConnect.

The driver supports the core SQL grammar.

Number of Connections and Statements Supported

The XDB database system supports a single connection and multiple statements within that connection.

DataDirect ODBC Drivers Reference Help

Individual Drivers

This help file is your on-line documentation for the INTERSOLV DataDirect ODBC Drivers for Windows version 3.1. Click any of the following topics for information on that topic:

About INTERSOLV DataDirect Database Drivers

Help on Individual Drivers

Supported ODBC Functions

ODBC.INI

Error Messages

SQL for Flat-File Drivers

For important last-minute information about the drivers, see the README file in the installation directory.

Help on Individual Drivers

For Help on a specific driver, click one of the following topics:

[ALLBASE Driver](#)

[Btrieve Driver](#)

[DB2/2 Driver](#)

[dBASE Driver](#)

[Excel Drivers](#)

[INFORMIX 4 Driver](#)

[INFORMIX 5 Driver](#)

[INGRES Drivers](#)

[InterBase Driver](#)

[MicroDecisionware Driver](#)

[Oracle Drivers](#)

[Paradox 4 Driver](#)

[Paradox 5 Driver](#)

[PROGRESS Driver](#)

[Scalable SQL Driver](#)

[SQL Server Driver](#)

[SQLBase Driver](#)

[Sybase System 10 Driver](#)

[Text Driver](#)

[XDB Driver](#)

About INTERSOLV DataDirect ODBC Drivers

The INTERSOLV DataDirect ODBC drivers are compliant with the Microsoft Open Database Connectivity (ODBC) specification.   ODBC is a specification for an application program interface (API) that enables applications to access multiple database management systems using Structured Query Language (SQL).

ODBC permits maximum interoperability--a single application can access many different database management systems.   This enables an ODBC developer to develop, compile, and ship an application without targeting a specific type of data source.   Users can then add the database drivers, which link the application to the database management systems of their choice.

INTERSOLV provides database drivers for both relational and flat-file database systems.   The flat-file drivers provide full SQL support as discussed in SQL for Flat-File Drivers.

Supported ODBC Functions

This topic lists the ODBC API functions that the database drivers support

All database drivers are ODBC Level 1-compliant--they support all ODBC Core and Level 1 functions.   A limited set of Level 2 functions is also supported.   The drivers support the functions listed in the following table.   Any additions to these supported functions or differences in the support of specific functions are listed in the "ODBC Conformance Level" topic for each individual driver.

Core Functions	Level 1 Functions
SQLAllocConnect	SQLColumns
SQLAllocEnv	SQLDriverConnect
SQLAllocStmt	SQLGetConnectOption
SQLBindCol	SQLGetData
SQLBindParameter*	SQLGetFunctions
SQLCancel	SQLGetInfo
SQLColAttributes	SQLGetStmtOption
SQLConnect	SQLGetTypeInfo
SQLDescribeCol	SQLParamData
SQLDisconnect	SQLPutData
SQLDrivers*	SQLSetConnectOption
SQLError	SQLSetStmtOption
SQLExecDirect	SQLSpecialColumns
SQLExecute	SQLStatistics
SQLFetch	SQLTables
SQLFreeConnect	**Level 2 Functions**
SQLFreeEnv	SQLDataSources
SQLFreeStmt	SQLExtendedFetch (forward scrolling only)
SQLGetCursorName	SQLMoreResults
SQLNumResultCols	SQLNativeSql
SQLPrepare	SQLNumParams
SQLRowCount	SQLParamOptions*
SQLSetCursorName	SQLSetScrollOptions
SQLTransact	

ODBC.INI

<u>ODBC.INI Examples</u>

ODBC.INI is a standard text file that stores ODBC configuration and connection information.   It resides in the Windows directory.   You can access the file via either the ODBC Administrator program in the Windows Control Panel or any text file editor such as Windows Notepad.

ODBC.INI contains a [*section_name*] heading that is followed by optional *attribute*=*value* pairs, called entries.   Both the section name and the attributes are case insensitive.   Comment lines begin with a semicolon (;).

The ODBC.INI format, as specified by the Microsoft Open Database Connectivity (ODBC) specification, is as follows:

```
[ODBC Data Sources]   ;Lists data sources available to ODBC.
ds_name1=driver_desc1  ;Lists each data source name followed
               ;  by a description.
ds_name2=driver_desc2
...

[ds_name1]           ;Defines the actual ODBC Driver source;
               ;  for example, Oracle.
Driver=path/dll        ;Defines the path to the driver DLL.
Description=desc        ;Briefly describes the data source
...

[ds_name2]
Driver=path/dll
Description=desc
...
```

The [ODBC Data Sources] section is mandatory.   It provides the driver manager with a list of data sources that are supported for your connection requests.   You can change the names in this list, but each entry must match its corresponding [*ds_name*] section in ODBC.INI.

The [*ds_name*] sections contain a Driver= specification, which points to the location of the installed driver, as well as a Description= specification that describes the driver.   If you change the location of a driver, you can change the Driver= specification to match the new location.   You can also use just the name of the driver, and the driver manager will attempt to locate the driver based on information obtained from your environment.

You might need to assign other entries depending on the data source you are configuring.   The "Connecting to a Data Source Using a Connection String" topic for each individual driver lists the attributes that you can set.   Use the ODBC Administrator program to modify ODBC.INI in all environments that provide this interface.   This protects the file from becoming corrupted or nonfunctional.

ODBC.INI Examples

The following example shows an ODBC.INI file as defined by the ODBC specification:

```
;--------------------------------------------------
; ODBC.INI - INTERSOLV ODBC Driver Manager INI File
;--------------------------------------------------

[ODBC Data Sources]
qess=SQL Server
qedbf=dBASE
qeor7=Oracle

[qess]
Driver=qess06.dll
Description=INTERSOLV SQL Server driver
ServerName=alice
LogonID=test

[qedbf]
Driver=qedbf06.dll
Description=INTERSOLV dBASE driver
Database=C:\DBASE
[qeor7]
Driver=qeor706.dll
Description=INTERSOLV Oracle driver
ServerName=t:magna:V7
LogonID=test
```

Error Messages

Error messages may come from
- An ODBC driver
- The database system
- The driver manager

An error reported on an ODBC driver has the following format:

[*vendor*] [*ODBC_component*] *message*

ODBC_component is the component in which the error occurred.   For example, an error message from INTERSOLV's SQL Server driver would look like this:

[INTERSOLV] [ODBC SQL Server driver] Login incorrect.

If you get this type of error, check the last ODBC call your application made for possible problems or contact your ODBC application vendor.

An error that occurs in the data source includes the data source name, in the following format:

[*vendor*] [*ODBC_component*] [*data_source*] *message*

With this type of message, *ODBC_component* is the component that received the error from the data source indicated.   For example, you may get the following message from an Oracle data source:

[INTERSOLV] [ODBC Oracle driver] [Oracle] ORA-0919: specified length too long for CHAR column

If you get this type of error, you did something incorrectly with the database system.   Check your database system documentation for more information or consult your database administrator.   In this example, you would check your Oracle documentation.

The driver manager is a DLL that establishes connections with drivers, submits requests to drivers, and returns results to applications.   An error that occurs in the driver manager has the following format:

[*vendor*] [ODBC DLL] *message*

vendor can be Microsoft, Apple, or INTERSOLV.   For example, an error from the Microsoft driver manager might look like this:

[Microsoft] [ODBC DLL] Driver does not support this function

If you get this type of error, consult the Programmer's Reference for the Microsoft ODBC Software Development Kit available from Microsoft.