**INFO: Improved ODBC DataType Mappings with Jet 4.0**

This article was previously published under Q214854

**SUMMARY**

The Microsoft Data Access Components (MDAC) version 2.1 installs the Microsoft Jet 4.0 database engine. The Microsoft Jet database engine is used by the Microsoft Access ODBC Driver, the Microsoft Jet OLE DB Provider, and the Microsoft Data Access Objects (DAO) to provide access to Microsoft Access-format databases as well as various ODBC and Jet ISAM data sources.  
  
For example, you can use ActiveX Data Objects (ADO), Remote Data Objects (RDO), or Data Access Objects (DAO) to gain access to Microsoft Jet by using any of the following methods:

ADO -> OLE DB -> Jet OLE DB Provider -> Jet

ADO -> OLE DB -> MSDASQL -> ODBC -> Access ODBC Driver -> Jet

DAO -> Jet

RDO -> ODBC -> Access ODBC Driver -> Jet

**Note**: "MSDASQL" above is the Microsoft ODBC Provider for OLE DB, which is an OLE DB Provider that can talk to any ODBC driver (and thus allow ADO to talk to any ODBC Driver).  
  
Microsoft Jet itself allows access to the following data sources:

Jet -> ODBC -> Non-ODBC Desktop Driver -> Any non-Jet data source

Jet -> Jet ISAM Driver -> Jet ISAM data source

Jet -> Microsoft Access database

**Note**: An "ODBC Desktop Driver" is any one of the Microsoft ODBC drivers that uses Microsoft Jet internally to access a data source. Using any of the ODBC Desktop Drivers from Jet is not supported. For example, using the Microsoft Excel ODBC Driver is not supported from Jet. Jet does support using the Microsoft Excel ISAM driver, so the Excel ISAM driver can be used instead in this particular case. You can verify whether an ODBC driver is an ODBC Desktop Driver by examining the driver file name in the ODBC Administrator control panel under the Drivers section. If the driver file name is Odbcjt32.dll, then the driver is an ODBC Desktop Driver and is not supported for use with Jet via ODBC.  
  
Microsoft Jet 4.0 provides an improved set of ODBC data type to Jet data type mappings versus the Microsoft Jet 3.5 database engine. For example, Jet 4.0 maps SQL\_DECIMAL and SQL\_NUMERIC type fields to a new Jet 4.0 data type called Decimal, providing a closer mapping to the actual ODBC data type. The Jet 4.0 Decimal data type is an exact numeric data type (called a scaled integer) that holds values from (10^28)-1 through -(10^28). With the Decimal data type, you can define fields with precision and scale from (1,0) up to (28,28). Jet 3.5 maps SQL\_DECIMAL and SQL\_NUMERIC fields to the closest Jet numeric data type depending upon the precision and scale of the ODBC field, which in certain cases results in mapping to a non-exact (floating point) numeric Jet data type, such as Double.

**MORE INFORMATION**

Before the Microsoft Jet database engine maps its data types to an ODBC table, it calls the ODBC API function SQLColumns to gather information about the ODBC data type, precision, and scale for each column in the table. Using this ODBC type information, Jet matches each ODBC data type with the appropriate Jet data type. This article summarizes the ODBC data type to Jet data type mappings used by Jet 3.5 and Jet 4.0.  
  
**Note:** The Jet data type LongBinary is listed as "Ole Object" in the Microsoft Access table designer user interface.

**ODBC to Jet Data Type Mappings**

ODBC SQL Type Precision Scale Jet 3.5 Type Jet 40 Type

-----------------------------------------------------------

SQL\_BIT N/A N/A Boolean Boolean

SQL\_TINYINT N/A N/A Byte\* Byte\*

SQL\_TINYINT N/A N/A Integer\* Integer\*

SQL\_SMALLINT N/A N/A Integer Integer

SQL\_INTEGER N/A N/A Long Long

SQL\_REAL N/A N/A Single Single

SQL\_FLOAT N/A N/A Double Double

SQL\_DOUBLE N/A N/A Double Double

SQL\_DECIMAL 0 To 4 0 Integer Decimal

SQL\_DECIMAL 5 To 9 0 Long Decimal

SQL\_DECIMAL 10 to 15 0 Double Decimal

SQL\_DECIMAL <=15 >0 Double Decimal

SQL\_DECIMAL 16 To 28 N/A Text Decimal

SQL\_DECIMAL > 28 N/A Text Text

SQL\_NUMERIC 0 To 4 0 Integer Decimal

SQL\_NUMERIC 5 To 9 0 Long Decimal

SQL\_NUMERIC 10 to 15 0 Double Decimal

SQL\_NUMERIC <=15 >0 Double Decimal

SQL\_NUMERIC 16 To 28 N/A Text Decimal

SQL\_NUMERIC > 28 N/A Text Text

SQL\_CHAR <= 255 N/A Text Text

SQL\_CHAR > 255 N/A Memo Memo

SQL\_VARCHAR <= 255 N/A Text Text

SQL\_VARCHAR > 255 N/A Memo Memo

SQL\_LONGVARCHAR N/A N/A Memo Memo

SQL\_WCHAR <= 255 N/A Unsupported Text

SQL\_WCHAR > 255 N/A Unsupported Memo

SQL\_WVARCHAR <= 255 N/A Unsupported Text

SQL\_WVARCHAR > 255 N/A Unsupported Memo

SQL\_WLONGVARCHAR N/A N/A Unsupported Memo

SQL\_DATE N/A N/A DateTime DateTime

SQL\_TIME N/A N/A DateTime DateTime

SQL\_TIMESTAMP N/A N/A DateTime DateTime

SQL\_BINARY <=255 N/A Binary Binary

SQL\_BINARY 256 To 510 N/A LongBinary Binary

SQL\_BINARY > 510 N/A LongBinary LongBinary

SQL\_VARBINARY <=255 N/A Binary Binary

SQL\_VARBINARY 256 To 510 N/A LongBinary Binary

SQL\_VARBINARY > 510 N/A LongBinary LongBinary

SQL\_LONGVARBINARY N/A N/A LongBinary LongBinary

SQL\_GUID N/A N/A Text Guid

\* An unsigned SQL\_TINYINT maps to a Jet Byte, a signed SQL\_TINYINT

maps to an Jet Integer.

**Special ODBC-to-Jet Data Type Mappings For SQL Server**

If Microsoft Jet is talking to the Microsoft SQL Server ODBC driver, then the following additional data type mappings occur:

ODBC SQL Type Precision Scale Jet 3.5 Type Jet 40 Type

-----------------------------------------------------------

SQL\_DECIMAL 10 4 Currency Currency

SQL\_DECIMAL 19 4 Currency Currency

SQL\_NUMERIC 10 4 Currency Currency

SQL\_NUMERIC 19 4 Currency Currency

**Code For Verifying Data Type Mappings**

The following sample code can be used to verify most of the mappings in the tables above. Run the code using DAO 3.5 and DAO 3.6 (obtained with Office 2000) to verify the Jet 3.5 and Jet 4.0 data type mappings, respectively.  
  
**NOTE:** The code requires a Microsoft SQL Server 7.0 database server; you need to adjust the connection string if the SQL Server database is on a different machine (change the SERVER=(Local); token to SERVER=Server Name;).

Option Explicit

' Various constant strings.

Const strConnect = "ODBC;Driver=SQL Server;SERVER=(Local);" & \_

"DATABASE=Pubs;UID=sa;PWD=;"

Const strSelectSQL = "select \* from tmpAllTypes"

Const strDropTableSQL = "drop table tmpAllTypes"

' Outputs a listing of ODBC -> Jet Data Type mappings using a

' SQL Server 7.0 table as the data source.

Sub ODBCJetMapTest()

Dim eng As New DAO.DBEngine

Dim qd As DAO.QueryDef

Dim db As DAO.Database

Dim rs As DAO.Recordset

Dim f As DAO.Field

Dim strSQL As String

' Verify DAO version used.

' Reference "Microsoft DAO 3.51 Object Library" to see

' Jet 3.5 mappings or "Microsoft DAO 3.6 Object Library"

' to view Jet 4.0 ODBC mappings.

Debug.Print "ODBCJetMapTest is using DAO version " & \_

eng.Version & "."

' Open SQL Server database connection.

Set db = eng.OpenDatabase("", False, False, strConnect)

' Verify SQL Server version.

Set qd = db.CreateQueryDef("")

qd.Connect = strConnect

qd.sql = "exec sp\_server\_info 500"

Set rs = qd.OpenRecordset()

Debug.Print "SQL Server version is " & \_

rs.Fields("attribute\_value") & \_

" (version 7.X or greater required)."

rs.Close

' Drop and re-create test table.

On Error Resume Next

db.Execute strDropTableSQL, dbSQLPassThrough

On Error GoTo 0

' Build our create table SQL.

strSQL = "CREATE TABLE tmpAllTypes("

' Common numeric mappings.

AddField strSQL, "SQL\_BIT", "bit", Empty

AddField strSQL, "SQL\_TINYINT", "tinyint", Empty

AddField strSQL, "SQL\_SMALLINT", "smallint", Empty

AddField strSQL, "SQL\_INTEGER", "int", Empty

AddField strSQL, "SQL\_REAL", "real", Empty

AddField strSQL, "SQL\_FLOAT", "float", Empty

' Decimal mappings.

AddField strSQL, "SQL\_DECIMAL", "decimal", Array(4, 0)

AddField strSQL, "SQL\_DECIMAL", "decimal", Array(5, 0)

AddField strSQL, "SQL\_DECIMAL", "decimal", Array(9, 0)

AddField strSQL, "SQL\_DECIMAL", "decimal", Array(10, 0)

AddField strSQL, "SQL\_DECIMAL", "decimal", Array(15, 0)

AddField strSQL, "SQL\_DECIMAL", "decimal", Array(16, 0)

AddField strSQL, "SQL\_DECIMAL", "decimal", Array(28, 0)

AddField strSQL, "SQL\_DECIMAL", "decimal", Array(4, 1)

AddField strSQL, "SQL\_DECIMAL", "decimal", Array(5, 1)

AddField strSQL, "SQL\_DECIMAL", "decimal", Array(9, 1)

AddField strSQL, "SQL\_DECIMAL", "decimal", Array(10, 1)

AddField strSQL, "SQL\_DECIMAL", "decimal", Array(15, 1)

AddField strSQL, "SQL\_DECIMAL", "decimal", Array(16, 1)

AddField strSQL, "SQL\_DECIMAL", "decimal", Array(28, 1)

' SQL Server specific mappings -> Currency.

AddField strSQL, "SQL\_DECIMAL", "decimal", Array(10, 4)

AddField strSQL, "SQL\_DECIMAL", "decimal", Array(19, 4)

' Numeric mappings.

AddField strSQL, "SQL\_NUMERIC", "numeric", Array(4, 0)

AddField strSQL, "SQL\_NUMERIC", "numeric", Array(5, 0)

AddField strSQL, "SQL\_NUMERIC", "numeric", Array(9, 0)

AddField strSQL, "SQL\_NUMERIC", "numeric", Array(10, 0)

AddField strSQL, "SQL\_NUMERIC", "numeric", Array(15, 0)

AddField strSQL, "SQL\_NUMERIC", "numeric", Array(16, 0)

AddField strSQL, "SQL\_NUMERIC", "numeric", Array(28, 0)

AddField strSQL, "SQL\_NUMERIC", "numeric", Array(4, 1)

AddField strSQL, "SQL\_NUMERIC", "numeric", Array(5, 1)

AddField strSQL, "SQL\_NUMERIC", "numeric", Array(9, 1)

AddField strSQL, "SQL\_NUMERIC", "numeric", Array(10, 1)

AddField strSQL, "SQL\_NUMERIC", "numeric", Array(15, 1)

AddField strSQL, "SQL\_NUMERIC", "numeric", Array(16, 1)

AddField strSQL, "SQL\_NUMERIC", "numeric", Array(28, 1)

' Character mappings.

AddField strSQL, "SQL\_CHAR", "char", Array(255)

AddField strSQL, "SQL\_CHAR", "char", Array(256)

AddField strSQL, "SQL\_VARCHAR", "varchar", Array(255)

AddField strSQL, "SQL\_VARCHAR", "varchar", Array(256)

AddField strSQL, "SQL\_WCHAR", "nchar", Array(255)

AddField strSQL, "SQL\_WCHAR", "nchar", Array(256)

AddField strSQL, "SQL\_WVARCHAR", "nvarchar", Array(255)

AddField strSQL, "SQL\_WVARCHAR", "nvarchar", Array(256)

' Binary mappings.

AddField strSQL, "SQL\_BINARY", "binary", Array(255)

AddField strSQL, "SQL\_BINARY", "binary", Array(256)

AddField strSQL, "SQL\_BINARY", "binary", Array(510)

AddField strSQL, "SQL\_BINARY", "binary", Array(511)

AddField strSQL, "SQL\_LONGVARBINARY", "image", Empty

' Date mappings.

AddField strSQL, "SQL\_TIMESTAMP", "datetime", Empty

' Specialized mappings.

AddField strSQL, "SQL\_GUID", "uniqueidentifier", Empty, ")"

' Create table.

db.Execute strSQL, dbSQLPassThrough

' Open recordset on table and dump out ODBC -> Jet mappings.

Set rs = db.OpenRecordset(strSelectSQL, dbOpenForwardOnly, \_

dbReadOnly, dbReadOnly)

For Each f In rs.Fields

Debug.Print f.Name & " maps to " & GetJetTypeString(f.Type) & "."

Next f

rs.Close

On Error Resume Next

db.Execute strDropTableSQL, dbSQLPassThrough

On Error GoTo 0

db.Close

End Sub

' Function to return string constant for Jet Type.

Function GetJetTypeString(lngDataTypeEnum As Long) As String

Dim strReturn As String

strReturn = "UNKNOWN"

Select Case lngDataTypeEnum

Case dbBigInt: strReturn = "dbBigInt"

Case dbBinary: strReturn = "dbBinary"

Case dbBoolean: strReturn = "dbBoolean"

Case dbByte: strReturn = "dbByte"

Case dbChar: strReturn = "dbChar"

Case dbCurrency: strReturn = "dbCurrency"

Case dbDate: strReturn = "dbDate"

Case dbDecimal: strReturn = "dbDecimal"

Case dbDouble: strReturn = "dbDouble"

Case dbFloat: strReturn = "dbFloat"

Case dbGUID: strReturn = "dbGUID"

Case dbInteger: strReturn = "dbInteger"

Case dbLong: strReturn = "dbLong"

Case dbLongBinary: strReturn = "dbLongBinary"

Case dbMemo: strReturn = "dbMemo"

Case dbNumeric: strReturn = "dbNumeric"

Case dbSingle: strReturn = "dbSingle"

Case dbText: strReturn = "dbText"

Case dbTime: strReturn = "dbTime"

Case dbTimeStamp: strReturn = "dbTimeStamp"

Case dbVarBinary: strReturn = "dbVarBinary"

End Select

GetJetTypeString = strReturn

End Function

' Function to append a SQL token to a SQL string.

Sub AddField(sql As String, FieldName As String, SQLType As String, \_

PS As Variant, Optional Terminator As String = ",")

If IsEmpty(PS) Then

sql = sql & FieldName & " " & SQLType

Else

sql = sql & FieldName & "\_" & Format(PS(0), "00")

If UBound(PS) = 0 Then

sql = sql & " " & SQLType

sql = sql & "(" & PS(0) & ")"

Else

sql = sql & "\_" & Format(PS(1), "00") & " " & SQLType

sql = sql & "(" & PS(0) & "," & PS(1) & ")"

End If

End If

sql = sql & Terminator

End Sub

**REFERENCES**

For more information on ODBC and Jet data type mappings, see Chapter 9 "Developing Client/Server Application" of the *Microsoft Jet Database Programmer's Guide*, Second Edition.

**The information in this article applies to:**

* Microsoft Data Access Components 2.1
* Microsoft Data Access Components 2.5
* Microsoft Data Access Components 2.6
* Microsoft OLE DB Provider for Jet 4.0

**Last Reviewed:** 12/5/2003 (4.0)

**Keywords:** kbDatabase kbinfo kbJET kbProvider KB214854