Oracle® Provider for OLE DB
Developer's Guide
12c Release 1 (12.1) for Microsoft Windows
E18594-02

May 2013
## Contents

**Preface**
- Audience ......................................................................................................................... vii
- Documentation Accessibility ........................................................................................... vii
- Related Documents ......................................................................................................... viii
- Conventions .................................................................................................................... viii

**Changes in This Release for Oracle Provider for OLE DB Developer's Guide**
- Changes in Oracle Database 12c Release 1 (12.1) ......................................................... ix

**1 Introduction to Oracle Provider for OLE DB**
- Overview of OLE DB ........................................................................................................ 1-1
- OLE DB Design ................................................................................................................ 1-1
- Overview of OraOLEDB ..................................................................................................... 1-2
- System Requirements ...................................................................................................... 1-3
- OraOLEDB Installation .................................................................................................... 1-3
- Component Certifications ............................................................................................... 1-4

**2 Features of OraOLEDB**
- OraOLEDB Provider Specific Features ........................................................................ 2-1
  - Data Source .................................................................................................................. 2-1
  - Compatibility with OLE DB Services ........................................................................... 2-2
  - Connecting to an Oracle Database .............................................................................. 2-3
  - OraOLEDB-Specific Connection String Attributes .................................................... 2-3
  - Default Attribute Values ............................................................................................ 2-4
  - Distributed Transactions .............................................................................................. 2-5
  - Enhanced Failover Capability ..................................................................................... 2-5
  - Operating System Authentication ............................................................................... 2-5
  - Password Expiration ................................................................................................... 2-6
  - VCharNull ..................................................................................................................... 2-7
  - SPPrmDefVal ................................................................................................................ 2-7
- OraOLEDB Sessions ........................................................................................................ 2-7
- Transactions ................................................................................................................... 2-7
- Commands ..................................................................................................................... 2-8
- Stored Procedures ......................................................................................................... 2-8
- Preparing Commands .................................................................................................... 2-8
Command Parameters........................................................................................................... 2-8
OraOLEDB Custom Properties for Commands................................................................. 2-8
Stored Procedures and Functions Returning Rowsets................................................... 2-12
Multiple Rowsets ........................................................................................................... 2-12
Statement Caching .......................................................................................................... 2-14
Metadata Caching .......................................................................................................... 2-15
Command Timeout and Cancel Method ........................................................................... 2-15
Rowsets ........................................................................................................................... 2-16
To Create Rowsets .......................................................................................................... 2-16
Updatability ....................................................................................................................... 2-17
Server Data on Insert Property ......................................................................................... 2-17
To Search for Rows with IRowsetFind::FindNext ............................................................. 2-18
OraOLEDB-Specific Connection String Attributes for Rowsets ................................... 2-18
Tips for ADO Programmers .............................................................................................. 2-19
Schema Rowsets .............................................................................................................. 2-19
Date Formats ..................................................................................................................... 2-19
Case of Object Names ....................................................................................................... 2-20
Data Types ....................................................................................................................... 2-20
Binary Data Types ............................................................................................................ 2-20
TIMESTAMP Data Types .................................................................................................... 2-21
INTERVAL Data Types ...................................................................................................... 2-22
LOB Support ..................................................................................................................... 2-23
Unicode Support .............................................................................................................. 2-24
Types of Unicode Encoding .............................................................................................. 2-24
How Oracle Unicode Support Works ................................................................................ 2-24
Unicode Support Setup .................................................................................................... 2-25
Errors ............................................................................................................................... 2-25
OLEDB.NET Data Provider Compatibility .......................................................................... 2-26
Using the OLEDB.NET Attribute in a Connection String ................................................. 2-26
Using OraOLEDB Custom Properties ............................................................................. 2-26
Updating Oracle with DataTable Changes ....................................................................... 2-27
Using OraOLEDB with Visual Basic ............................................................................... 2-27
Setting Up the Oracle Database ....................................................................................... 2-27
Setting Up the Visual Basic Project .................................................................................. 2-28

A  Provider-Specific Information

Data Type Mappings in Rowsets and Parameters ............................................................ A-1
Properties Supported ........................................................................................................ A-2
Data Source Properties .................................................................................................... A-2
DataSourceInfo Properties ............................................................................................. A-2
Initialization and Authorization Properties ...................................................................... A-4
Session Properties .......................................................................................................... A-4
Rowset Properties .......................................................................................................... A-5
Rowset Property Implications ........................................................................................ A-8
Interfaces Supported ...................................................................................................... A-8
Data Source ..................................................................................................................... A-9
Session ............................................................................................................................ A-9
Glossary

Index
Preface

Based on an open standard, Oracle Provider for OLE DB (OraOLEDB) allows access to Oracle Databases. This documentation describes OraOLEDB's provider-specific features and properties.

This document describes the features of Oracle Database for Windows that apply to the Windows operating system.

This Preface contains these topics:

- **Audience**
- **Documentation Accessibility**
- **Related Documents**
- **Conventions**

**Audience**

*Oracle Provider for OLE DB Developer’s Guide* is intended for programmers developing applications to access an Oracle Database using Oracle Provider for OLE DB. This documentation is also valuable to systems analysts, project managers, and others interested in the development of database applications.

To use this document, you must be familiar with OLE DB and have a working knowledge of application programming using Microsoft C/C++, Visual Basic, or ActiveX Data Objects (ADO). Knowledge of Component Object Model (COM) concepts are also useful.

Readers should also be familiar with the use of Structured Query Language (SQL) to access information in relational database systems.

**Documentation Accessibility**


**Access to Oracle Support**

Related Documents

For more information, see these Oracle resources:

- Oracle Database Installation Guide for Microsoft Windows
- Oracle Database Release Notes for Microsoft Windows
- Oracle Database Platform Guide for Microsoft Windows
- Oracle Database Concepts
- Oracle Services for Microsoft Transaction Server Developer’s Guide for Microsoft Windows
- Oracle Database Net Services Administrator’s Guide
- Oracle Database New Features Guide
- Oracle Database Reference
- Oracle Database Globalization Support Guide

For information about Oracle error messages, see Oracle Database Error Messages. Once you find the specific range, you can search for the specific message. When connected to the Internet, you can search for a specific error message using the error message search feature of the Oracle online documentation.

Many of the examples in this book use the sample schemas of the seed database, which is installed by default when you install Oracle. Refer to Oracle Database Sample Schemas for information on how these schemas were created and how you can use them yourself.

To download free release notes, installation documentation, white papers, or other collateral, please visit the Oracle Technology Network (OTN)

http://www.oracle.com/technetwork/index.html

For the latest version of the Oracle documentation, including this guide, visit


For additional information, see:

http://www.microsoft.com

Conventions

The following text conventions are used in this document:

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>boldface</strong></td>
<td>Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.</td>
</tr>
<tr>
<td><em>italic</em></td>
<td>Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.</td>
</tr>
<tr>
<td>monospace</td>
<td>Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.</td>
</tr>
</tbody>
</table>
Changes in This Release for Oracle Provider for OLE DB Developer's Guide

This preface contains:

- Changes in Oracle Database 12c Release 1 (12.1)

Changes in Oracle Database 12c Release 1 (12.1)

The following are changes in Oracle Provider for OLE DB Developer’s Guide for Oracle Database 12c Release 1 (12.1).

New Features
There are no new features for Oracle Database 12c Release 1 (12.1).
This chapter introduces Oracle Provider for OLE DB (OraOLEDB).

This chapter contains these topics:

- Overview of OLE DB
- Overview of OraOLEDB
- System Requirements
- OraOLEDB Installation
- Component Certifications

**Overview of OLE DB**

OLE DB is an open standard data access methodology which utilizes a set of **Component Object Model (COM)** interfaces for accessing and manipulating different types of data. These interfaces are available from various database providers.

**OLE DB Design**

The design of OLE DB centers around the concept of a **consumer** and a **provider**. **Figure 1–1** is an illustration of the OLE DB system. The consumer represents the traditional client. The provider places data into a tabular format and returns it to the consumer.
OLE DB Data Providers
OLE DB data providers are a set of COM components that transfer data from a data source to a consumer. An OLE DB Provider places that data in a tabular format in response to calls from a consumer. Providers can be simple or complex. A provider may return a table, it may allow the consumer to determine the format of that table, and it may perform operations on the data.

Each provider implements a standard set of COM interfaces to handle requests from the consumer. A provider may implement optional COM interfaces to provide additional functionality.

With the standard interfaces, any OLE DB consumer can access data from any provider. Because of COM components, consumers can access them in any programming language that supports COM, such as C++, Visual Basic, and Java.

OLE DB Data Consumers
The OLE DB data consumer is any application or tool that utilizes OLE DB interfaces of a provider to access a broad range of data.

Overview of OraOLEDB
Oracle Provider for OLE DB (OraOLEDB) is an OLE DB data provider that offers high performance and efficient access to Oracle data by OLE DB consumers.

In general, this developer’s guide assumes that you are using OraOLEDB through OLE DB or ADO.
With the advent of the .NET framework, support has been provided for using the OLEDB.NET Data Provider with OraOLEDB. With the proper connection attribute setting, an OLEDB.NET Data Provider can utilize OraOLEDB to access Oracle Database.

See Also: "OLEDB.NET Data Provider Compatibility" on page 2-26 for further information on support for OLEDB.NET Data Provider

System Requirements

The following items are required on a system to use Oracle Provider for OLE DB:

- Windows Operating System:

  Oracle supports 32-bit Oracle Provider for OLE DB on x86, AMD64, and Intel EM64T processors on these operating systems.


  Oracle supports 32-bit Oracle Provider for OLE DB and 64-bit Oracle Provider for OLE DB for Windows x64 on these operating systems.

- Access to an Oracle Database (Oracle 10.2 or later)

- Oracle Client and Oracle Net Services (included with Oracle Provider for OLE DB installation).

- Redistributable files provided with Microsoft Data Access Components (MDAC) 2.1 or higher are required by the provider. These files are available at the Microsoft Web site:
  

- Oracle Services for Microsoft Transaction Server release 12.1 or higher. This is required for consumers using Microsoft Transaction Server (MTS) or COM+.

OraOLEDB Installation

Oracle Provider for OLE DB is included as part of your Oracle installation. It contains the features and demos that illustrate how to use this product to solve real-world problems.

See Also: The Oracle Database Installation Guide for Microsoft Windows for installation instructions

During the installation process, the files listed in Table 1–1 are installed on the system. Some files have ver in their name to indicate the release version.
Component Certifications

Oracle provides support information for components on various platforms, lists compatible client and database versions, and identifies patches and workaround information.

Find the latest certification information at My Oracle Support (formerly OracleMetaLink):

https://support.oracle.com
This chapter describes components of Oracle Provider for OLE DB (OraOLEDB) and how to use the components to develop OLE DB consumer applications.

This chapter contains these topics:
- OraOLEDB Provider Specific Features
- Using OraOLEDB with Visual Basic

**OraOLEDB Provider Specific Features**

The following sections describe provider-specific features of OraOLEDB:
- Data Source
- OraOLEDB Sessions
- Commands
- Rowsets
- Data Types
- LOB Support
- Unicode Support
- Errors
- OLEDB.NET Data Provider Compatibility

Additional provider-specific information is provided in Appendix A, "Provider-Specific Information".

**Data Source**

A data source object in OraOLEDB is responsible for establishing the first connection to the Oracle Database. To establish the initial connection, the consumer must use the CoCreateInstance function to create an instance of the data source object. This function requires important information about the provider: class ID of the provider and executable context. The class ID of OraOLEDB is CLSID_OraOLEDB.

OraOLEDB is an in-process server. When calling CoCreateInstance, use the CLSCTX_INPROC_SERVER macro. For example:

```c
// create an instance of OraOLEDB data source object and
// obtain the IDBInitialize interface
hr = CoCreateInstance(CLSID_OraOLEDB, NULL,
                      CLSCTX_INPROC_SERVER, IID_IDBInitialize,
```
The code snippet above does not enable OLEDB Services when instantiating the Data Source object. To enable OLEDB services, see "Compatibility with OLE DB Services" on page 2-2 below.

**Note:** OraOLEDB does not support persistent data source objects.

After the successful creation of an instance of a data source object, the consumer application can initialize the data source and create sessions.

OraOLEDB supports connections to Oracle Databases releases. To connect to a specific database, the consumer is required to set the following properties of the `DBPROPSET_DBINIT` property set:

- `DBPROP_AUTH_USERNAME` with the user ID, such as `scott`
- `DBPROP_AUTH_PASSWORD` with the password, such as `tiger`
- `DBPROP_INIT_DATASOURCE` with the net service name, such as `myOraDb`

The consumer could also populate `DBPROP_INIT_PROMPT` with `DBPROMPT_PROMPT` which causes the provider to display a logon box for the user to enter the connect information.

Using `DBPROMPT_NOPROMPT` disables display of the logon box. In this case, incomplete logon information causes the provider to return a logon error. However, if this property is set to `DBPROMPT_COMPLETE` or `DBPROMPT_COMPLETEREQUIRED`, the logon box will be displayed only if the logon information is incomplete.

**Compatibility with OLE DB Services**

OraOLEDB is compatible with OLE DB Services that are available in OLE DB version 2.0 and later. OLE DB Services contains useful services such as automatic transaction enlistment, Client Cursor Engine (CCE), connection and session pooling, which can enhance application performance, and others.

OLE DB Services can be used with OraOLEDB through C++/COM or ADO.

By default, the `OLEDB_SERVICES` registry entry for OraOLEDB is set, under the `CLSID` of OraOLEDB, to `0xffffffff` (that is, -1), which enables all services. Certain OLE DB Services can also be disabled or enabled programmatically through the `DBPROP_INIT_OLEDBSERVICES` property setting.

**See Also:**

for more information on OLE DB Services and how to enable or disable specific services

**ADO Applications with OLE DB Services** ADO automatically enables OLE DB Services. Thus, ADO applications do not need any special code to use OLEDB Services.

**C++/COM Applications with OLE DB Services** For C++/COM applications, some additional steps are needed to use OLE DB Services.

The following code snippet shows one way that C++/COM applications can enable OLE DB Services. The code shows the OLE DB consumer creating an instance of the `CLSID_MSDAINITIALIZE` class through `CoCreateInstance()`, obtaining the
IDataInitialize interface from that object, and then creating an OLE DB data source object through that interface.

```csharp
// Instantiate the CLSID_MSDAINITIALIZE class and request for the
// IID_IDataInitialize interface from it
hr = CoCreateInstance(CLSID_MSDAINITIALIZE, NULL, CLSCTX_INPROC_SERVER,
                     IID_IDataInitialize, (void**)&pIDataInitialize);

// Set properties, datasource name, userid, and password, etc.
...

// Create an OLEDB data source object using the interface obtained from the
// CLSID_MSDAINITIALIZE class.
hr = pIDataInitialize->CreateDBInstance(CLSID_OraOLEDB, NULL,
                                        CLSCTX_INPROC_SERVER, NULL,IID_IDBInitialize,(IUnknown**)&pIDBInitialize);
...

// If connection/session pooling was enabled, pIDBInitialize->Release()
// releases the connection/session back to the pool.
// pIDBInitialize->Release() should not be called until the application no
// longer need to use connection/session pooling and the rest of
// the OLE DB Services that were enabled for the application.
//
// pIDBInitialize->Release();
```

**Connecting to an Oracle Database**

To connect to an Oracle Database using OraOLEDB, the OLE DB connection string must be as follows:

```
*Provider=OraOLEDB.Oracle;User ID=user;Password=pwd;Data Source=constr;*
```

When connecting to a remote database, Data Source must be set to the correct net service name which is the alias in the tnsnames.ora file. For more information, refer to *Oracle Database Net Services Administrator's Guide*.

**OraOLEDB-Specific Connection String Attributes**

OraOLEDB offers provider-specific connection string attributes, which are set in the same way as the Provider and User ID are set. The provider-specific connection string attributes are:

- **CacheType** - specifies the type of cache used to store the rowset data on the client. See "OraOLEDB-Specific Connection String Attributes for Rowsets" on page 2-18.

- **ChunkSize** - specifies the size of LONG or LONG RAW column data stored in the provider's cache. See "OraOLEDB-Specific Connection String Attributes for Rowsets" on page 2-18.

- **DistribTX** - enables or disables distributed transaction enlistment capability. See "Distributed Transactions" on page 2-5.

- **FetchSize** - specifies the size of the fetch array in rows. See "OraOLEDB-Specific Connection String Attributes for Rowsets" on page 2-18.

- **OLEDB.NET** - enables or disables compatibility with OLEDB.NET Data Provider. See “OLEDB.NET Data Provider Compatibility” on page 2-26.

- **OSAuthent** - specifies whether operating system authentication will be used when connecting to an Oracle Database. See “Operating System Authentication” on page 2-5.
■ **PLSQLRSet** - enables or disables the return of a rowset from PL/SQL stored procedures. See "OraOLEDB Custom Properties for Commands" on page 2-8.

■ **PwdChgDlg** - enables or disables displaying the password change dialog box when the password expires. See "Password Expiration" on page 2-6.

■ **UseSessionFormat** - specifies whether to use the default NLS session formats or let OraOLEDB override some of these formats for the duration of the session. Valid values are 0 (FALSE) and 1 (TRUE). The default is FALSE which lets OraOLEDB override some of the default NLS session formats. If the value is TRUE, OraOLEDB uses the default NLS session formats.

Note that this connection attribute does not appear under the `\\HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\KEY_HOMENAME\OLEDB` registry key.

■ **VCharNull** - enables or disables the NULL termination of VARCHAR2 OUT parameters from stored procedures.

■ **SPPrmDefVal** - specifies whether to use the default value or a NULL value if the application has not specified a stored procedure parameter value.

■ **NDataType** - specifies whether any of the parameters bound to the command are of N data types, which include NCHAR, NVARCHAR2, or NCLOB. See "NDatatype" on page 2-9.

Note that this connection attribute does not appear under the `\\HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\KEY_HOMENAME\OLEDB` registry key.

■ **SPPrmsLOB** - specifies whether one or more parameters bound to the stored procedures are of LOB data type, which include CLOB, BLOB, or NCLOB. See "SPPrmsLOB" on page 2-10.

Note that this connection attribute does not appear under the `\\HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\KEY_HOMENAME\OLEDB` registry key.

■ **StmtCacheSize** – specifies the maximum number of statements that can be cached. See "Statement Caching" on page 2-14.

■ **MetaDataCacheSize** - specifies the maximum number of SELECT statements for which the metadata can be cached. See "Metadata Caching" on page 2-15.

■ **DeferUpdChk** - specifies whether or not to defer the updateability check to support updating read-only disconnected rowsets. See DeferUpdChk under "OraOLEDB-Specific Connection String Attributes for Rowsets" on page 2-18.

■ **DBNotifications** - specifies whether or not to subscribe to the high availability events. See "Enhanced Failover Capability" on page 2-5.

■ **DBNotificationPort** - specifies the port number, which is opened to listen to the Database notifications. See "Enhanced Failover Capability" on page 2-5.

### Default Attribute Values

The default values for these attributes are located under the `\\HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\KEY_HOMENAME\OLEDB` registry key, where KEY_HOMENAME is the Oracle home.

The registry default values are read by OraOLEDB from the registry when the provider is loaded into memory. If Oracle-specific connection string attributes are not provided at connection time, then the default registry values are used. However, if the attributes are provided, then these new values override the default registry values.

These attributes can be set by setting the `DBPROP_INIT_PROVIDERSTRING` property, provided in the `DBPROPSET_DBINIT` property set. For example:
Distributed Transactions

The DistribTX attribute specifies whether sessions are enabled to enlist in distributed transactions. Valid values are 0 (disabled) and 1 (enabled). The default is 1 which indicates that sessions are enabled for distributed transaction enlistments.

Applications using Microsoft Distributed Transaction Coordinator must have DistribTX set to 1, the default.

Enhanced Failover Capability

This feature enhances failover capability.

These connection string attributes support enhanced failover capability.

- **DBNotifications**

  The DBNotifications attribute specifies whether or not to subscribe to high availability events. Valid values are 0 (FALSE) and 1 (TRUE). The default is FALSE, which indicates that OraOLEDB does not subscribe to high availability events. If this attribute is not provided at the connection time, then the default registry value is used.

- **DBNotificationPort**

  The DBNotificationPort attribute specifies the port number, which is used to listen to the database notifications. The valid value is an unsigned integer.

  DBNotificationPort is effective only if the DBNotifications attribute is set to TRUE, either through the connection string attribute or by registry entry. The default for the DBNotificationPort attribute is 0, which implies that OraOLEDB opens a valid port randomly. OraOLEDB does not validate the port number, so it is the responsibility of the application to specify a valid port number.

Enabling Failover Capability Through Registry Entry

- **DBNotifications**

  The DBNotifications registry entry specifies whether or not to subscribe to high availability events. Valid values are 0 (FALSE) and 1 (TRUE). The default value is FALSE, OraOLEDB does not subscribe. This registry entry value is used when the DBNotifications connection string attribute is not set. It is located under the `\HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\KEY_HOMENAME\OLEDB` registry key.

Operating System Authentication

The OSAuthent attribute specifies whether operating system authentication will be used when connecting to an Oracle Database. Valid values are 0 (disabled) and 1 (enabled). The default is 0, which indicates that operating system authentication is not used.

Operating system authentication is the feature by which Oracle uses the security mechanisms of the operating system to authorize users. For more information on this subject and how to set it up on Windows clients, refer to the information on authenticating database users on Windows in *Oracle Database Platform Guide for Microsoft Windows*.

After the Windows client has been set up properly for operating system authentication, this feature may be enabled by OraOLEDB clients by setting any of the following:
- DBPROP_AUTH_USERNAME to /
- DBPROP_INIT_PROVIDERSTRING to OSAuthent=1;
- OSAuthent in the registry to 1

**Password Expiration**

Oracle provides a Password Expiration feature which allows database administrators to force users to change their passwords regularly. The PwdChgDlg attribute enables or disables the displaying of the password change dialog box, whenever a logon fails due to an expired password. When enabled, the provider displays the dialog box to change the password. When disabled, the logon fails with an error message. The valid values are 0 (disabled) and 1 (enabled). The default is 1 (enabled). For more information on the Password Expiration feature, see *Oracle Database Security Guide*.

**Example: Connecting to an Oracle Database Using ADO**

The following examples illustrate how to connect to an Oracle Database using OraOLEDB and ADO.

---

**Note:** If Data Source, User ID, and Password are provided with the Open method, then ADO ignores those ConnectionString attributes.

---

**Connect Using ConnectionString**

```vba
Dim con As New ADODB.Connection
con.ConnectionString = "Provider=OraOLEDB.Oracle;Data Source=MyOraDb;" & _
    "User ID=scott;Password=tiger;"
con.Open
```

**Connect Without Using ConnectionString**

```vba
Dim con As New ADODB.Connection
con.Provider = "OraOLEDB.Oracle"
con.Open "MyOraDb", "scott", "tiger"
```

**Connect and Set Provider-specific Attributes**

```vba
Dim con As New ADODB.Connection
con.Provider = "OraOLEDB.Oracle"
con.ConnectionString = "FetchSize=200;CacheType=Memory;" & _
    "OSAuthent=0;PLSQLRSet=1;Data Source=MyOraDb;" & _
    "User ID=scott;Password=tiger;"
con.Open
```

**Operating System-Authenticated Connect Setting User ID to /**

```vba
Dim con As New ADODB.Connection
con.Provider = "OraOLEDB.Oracle"
con.Open "MyOraDb", "/", ""
```

**Operating System-Authenticated Connect Using OSAuthent**

```vba
Dim con As New ADODB.Connection
con.Provider = "OraOLEDB.Oracle"
con.ConnectionString = "Data Source=MyOraDb;OSAuthent=1;"
con.Open
```
**VCharNull**

The **VCharNull** attribute enables or disables the NULL termination of **VARCHAR2 OUT** parameters from stored procedures. Valid values are 0 (disabled) and 1 (enabled). The default is 1, which indicates that **VARCHAR2 OUT** parameters are NULL terminated. A value of 0 indicates that **VARCHAR2 OUT** parameters are padded with spaces.

The default value for this attribute is located under the `\HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\KEY_HOMENAME\OLEDB` registry key, where **HOMENAME** is the Oracle home. If this attribute is not provided at the connection time, then the default registry value is used.

Note that with this connection attribute enabled, applications need to pad the stored procedure **IN** and **IN OUT CHAR** parameters with spaces explicitly, if the parameter is to be used in a **WHERE** clause.

**SPPrmDefVal**

The **SPPrmDefVal** attribute specifies whether to use the default value or a NULL value if the application has not specified a stored procedure parameter value. Valid values are 0 (FALSE) and 1 (TRUE). The default is FALSE, which enables OraOLEDB to pass a NULL value. If the value is TRUE, then OraOLEDB uses the default value.

The default value for this attribute is located under the `\HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\KEY_HOMENAME\OLEDB` registry key. If this attribute is not provided at connection time, then the default registry value is used.

**OraOLEDB Sessions**

An OraOLEDB session object represents a single connection to an Oracle Database. The session object exposes the interfaces that allow data access and manipulation.

The first session created on the initialized data source inherits the initial connection established by `IDBInitialize::Initialize()`. Subsequent sessions that are created establish their own independent connections to the particular Oracle Database specified by the data source properties.

Each session object also defines a transaction space for a data source. All command and rowset objects created from a particular session object are part of the transaction of that session.

After all references to the session object are released, the session object is removed from memory and the connection is dropped.

**Transactions**

OraOLEDB supports local and distributed transactions, which provide explicit commit and abort transactional operations.

OraOLEDB does not support nested transactions. In addition, a local transaction cannot be started if the session is currently enlisted in a distributed transaction. This also applies to distributed transactions if the session is currently enlisted in a local transaction.

**Local Transactions** OraOLEDB supports the **ITransactionLocal** interface for explicit transactions. By default, OraOLEDB is in an autocommit mode, meaning that each unit of work done on the database is automatically or implicitly committed. With the use of the **ITransactionLocal** interface, consumers may explicitly start a transaction for a particular session, allowing a unit of work to be explicitly committed or aborted by the consumer.
OraOLEDB supports the Read Committed (Cursor Stability) isolation level. In this level, the changes made by other transactions are not visible until those transactions are committed.

**Distributed Transactions**  OraOLEDB consumers must install Oracle Services for Microsoft Transaction Server (MTS) release 10.2 or higher to be able to participate in Microsoft Transaction Server (or COM+) transactions or to enlist in a distributed transaction coordinated by Microsoft Distributed Transaction Coordinator (MS DTC). For setup and configuration information on Oracle Services for MTS, see *Oracle Services for Microsoft Transaction Server Developer’s Guide for Microsoft Windows*.

OraOLEDB ignores *IsoLevel, IsoFlags, and pOtherOptions* parameters when *ITransactionJoin::JoinTransaction()* is called. These options must be provided when the consumer acquires a transaction object from MS DTC with the *ITransactionDispenser::BeginTransaction()* method call.

However, if *IsoFlags* is nonzero, then *XACT_E_NOISORETAIN* is returned.

**Commands**

OraOLEDB supports ANSI SQL as supported by Oracle Database and the ODBC SQL syntax.

**Stored Procedures**

When executing an Oracle PL/SQL *stored procedure* using a command, use Oracle native syntax or the ODBC procedure call escape sequence in the command text:

- Oracle native syntax: `BEGIN credit_account(123, 40); END;`
- ODBC syntax: `{CALL credit_account(123, 40)}`

**Preparing Commands**

OraOLEDB validates and fetches the metadata only for SELECT SQL statements.

**Command Parameters**

When using Oracle ANSI SQL, parameters in the command text are preceded by a colon. In ODBC SQL, parameters are indicated by a question mark (?).

OraOLEDB supports input, output, and input and output parameters for PL/SQL stored procedures and stored functions. OraOLEDB supports input parameters for SQL statements.

---

*Note:* OraOLEDB supports only positional binding.

---

**OraOLEDB Custom Properties for Commands**

OraOLEDB custom properties for commands are grouped under the custom property set `ORAPROPSET_COMMANDS`. It provides these properties:

**Table 2–1  Custom Properties for Commands**

<table>
<thead>
<tr>
<th>For Visual Basic Users</th>
<th>For C++ Users</th>
</tr>
</thead>
<tbody>
<tr>
<td>PLSQLRSet</td>
<td>ORAPROP_PLSQLRSet</td>
</tr>
<tr>
<td>NDatatype</td>
<td>ORAPROP_NDatatype</td>
</tr>
<tr>
<td>SPPrmsLOB</td>
<td>ORAPROP_SPPrmsLOB</td>
</tr>
</tbody>
</table>
### PLSQLRSet

This property is similar to the `PLSQLRSet` connection string attribute.

The property specifies whether OraOLEDB must return a rowset from the PL/SQL stored procedure. If the stored procedure, provided by the consumer, returns a rowset, `PLSQLRSet` must be set to `TRUE` (enabled). This property should be set to `FALSE` after the command has been run. By default, the property is set to `FALSE` (disabled).

Consumers should use the property over the attribute, as the property can be set at the command object rather than at the session. By setting it at the command object, the consumer is able to set the property only for the command object executing stored procedures which are returning rowsets. With the attribute, the consumer needed to set it even if only one of many stored procedures being executed by the ADO application returned a rowset. The use of this property should provide a performance boost to applications making use of the attribute previously.

#### Example: Setting the Custom Property PLSQLRSet

```vba
Dim objRes As NEW ADODB.Recordset
Dim objCon As NEW ADODB.Connection
Dim objCmd As NEW ADODB.Command

....
objCmd.ActiveConnection = objCon
objCmd.CommandType = adCmdText

' Enabling the PLSQLRSet property indicates to the provider
' that the command returns one or more rowsets
objCmd.Properties("PLSQLRSet") = TRUE

' Assume Employees.GetEmpRecords() has a REF CURSOR as
' one of the arguments
objCmd.CommandText = "\n\{ CALL Employees.GetEmpRecords(?,?) \n"

' Execute the SQL
set objRes = objCmd.Execute

' It is a good idea to disable the property after execute as the
' same command object may be used for a different SQL statement
objCmd.Properties("PLSQLRSet") = FALSE
```

### NDatatype

This property allows the consumers to specify whether any of the parameters bound to the command are of Oracle's N data types (`NCHAR`, `NVARCHAR2`, or `NCLOB`). This information is required by OraOLEDB to detect and bind the parameters. This property should not be set for commands executing `SELECT` statements. However, this property must be set for all other SQL statements, such as `INSERT`, `UPDATE`, and `DELETE`.

The use of this property should be limited to SQL statements containing parameters of N data type as setting it incurs a processing overhead of at least one round-trip to the database. By default, this property is set to `FALSE`. 

---

**Table 2–1 (Cont.) Custom Properties for Commands**

<table>
<thead>
<tr>
<th>For Visual Basic Users</th>
<th>For C++ Users</th>
</tr>
</thead>
<tbody>
<tr>
<td>AddToStmtCache</td>
<td>ORAPROP_AddToStmtCache</td>
</tr>
</tbody>
</table>
**Note:** OraOLEDB does not support parameters of N data types in the `WHERE` clause of SQL statements.

**Note:** Consumers are required to use the ODBC procedure call escape sequence to call stored procedures or functions having N data type parameters.

**Example: Setting the Custom Property NDatatype**

Dim objCon As New ADODB.Connection
Dim objCmd As New ADODB.Command
Dim prEmpno As New ADODB.Parameter
Dim prEname As New ADODB.Parameter
...
objCmd.ActiveConnection = objCon
objCmd.CommandType = adCmdText

' Create and append the parameters to the command object
Set prEmpno = objCmd.CreateParameter("prEmpno", adSmallInt, adParamInput, , 8521)
' prEname is bound to a NVARCHAR2 column in the EMP table
Set prEname = objCmd.CreateParameter("prEname", adBSTR, adParamInput, , "Joe")
objCmd.Parameters.Append prEmpno
objCmd.Parameters.Append prEname

' Enabling the NDatatype property indicates to the provider
' that one or more of the bound parameters is of N datatype
objCmd.Properties("NDatatype") = TRUE

' Assume column ENAME in table EMP is of NVARCHAR2 type
objCmd.CommandText = '"INSERT INTO EMP (EMPNO, ENAME) VALUES (?, ?)"

' Execute the SQL
objCmd.Execute

' It is a good idea to disable the property after execute as the same command
' object may be used for a different SQL statement
objCmd.Properties("NDatatype") = FALSE

**SPPrmsLOB**

This property allows the consumer to specify whether one or more of the parameters bound to the stored procedures are of Oracle's LOB data type (CLOB, BLOB, or NCLOB). OraOLEDB requires this property to be set to TRUE, to fetch the parameter list of the stored procedure prior to execution. The use of this property limits the processing overhead to stored procedures having one or more LOB data type parameters. This property should be set to FALSE after the command has been executed. By default, the property is set to FALSE.

**Note:** Consumers are required to use the ODBC procedure call escape sequence to call stored procedures or functions having LOB data type parameters.

**Example: Setting the Custom Property SPPrmsLOB**

Dim objCon As New ADODB.Connection
Dim objCmd As New ADODB.Command
Dim prCLOB As NEW ADODB.Parameter
...
objCmd.ActiveConnection = objCon
objCmd.CommandType = adCmdText

' Create and append the parameters to the command object
Set prCLOB = objCmd.CreateParameter("prCLOB", adLongVarchar, adParamOutput, _
10000)

objCmd.Parameters.Append prCLOB

' Enabling the SPPrmsLOB property indicates to the provider
' that one or more of the bound parameters is of LOB data type
objCmd.Properties("SPPrmsLOB") = TRUE

' Assume the Stored Procedure requires a CLOB parameter
objCmd.CommandText = "( call storedproc(?) )"

' Execute the SQL
objCmd.Execute

' It is a good idea to disable the property after execute as the
' same command object may be used for a different SQL statement
objCmd.Properties("SPPrmsLOB") = FALSE

AddToStmtCache
This property allows the consumer to cache the executed statements when the
property is set to TRUE and statement caching is enabled. If the statement caching is
disabled or if this property is set to FALSE, then the executed statement is not cached.

This property is ignored if statement caching is disabled. Statement caching can be
enabled by setting the StmtCacheSize connection string attribute to a value greater
than zero. This property provides a way to selectively add statements to the cache
when statement caching is enabled. By default, the property is set to TRUE.

Example: Setting the Custom Property AddToStmtCache
Dim objCon As NEW ADODB.Connection
Dim objCmd As NEW ADODB.Command
...

' Statement caching is enabled by setting the 'StmtCacheSize'
' connection string attribute to a value greater than zero
objCon.ConnectionString = "StmtCacheSize=10;Data Source=MyOraDb;" & _
"User ID=scott;Password=tiger;"

objCon.Open
objCmd.ActiveConnection = objCon
objCmd.CommandType = adCmdText
objCmd.CommandText = "SELECT * FROM EMP"

' "SELECT * FROM EMP" statement would be added to the statement cache because
' StmtCacheSize connection string attribute value is greater than 0 and
' AddToStmtCache property value is TRUE by default.
objCmd.Execute

' Do not add "SELECT * FROM DEPT" to the statement cache
objCmd.CommandText = "SELECT * FROM DEPT"
objCmd.Properties("AddToStmtCache") = FALSE

' "SELECT * FROM DEPT" statement would not be added to the statement cache
objCmd.Execute
Stored Procedures and Functions Returning Rowsets

Oracle Provider for OLE DB allows consumers to execute a PL/SQL stored procedure with an argument of REF CURSOR type or a stored function returning a REF CURSOR value.

OraOLEDB returns a rowset for the REF CURSOR bind variable. Because there is no predefined data type for REF CURSOR in the OLE DB specification, the consumer must not bind this parameter.

If the PL/SQL stored procedure has one or more arguments of REF CURSOR type, OraOLEDB binds these arguments and returns a rowset for each argument of REF CURSOR type.

If the PL/SQL stored function returns a REF CURSOR or has an argument of REF CURSOR type, OraOLEDB binds these and returns a rowset for each REF CURSOR type.

To use this feature, stored procedures or functions must be called in the ODBC procedure call escape sequence.

The stored procedure or function being called could be either standalone or packaged. However, the REF CURSOR being returned must be explicitly defined in a package in the database.

Multiple Rowsets

OraOLEDB supports returning more than one rowset from a stored procedure. Consumers can use this feature to access all the REF CURSORS being returned by a stored procedure.

Example: Stored Procedure Returning Multiple Rowsets

PL/SQL Package

CREATE OR REPLACE PACKAGE Employees AS
  TYPE empcur IS REF CURSOR;

  PROCEDURE GetEmpRecords(p_cursor OUT empcur,
    q_cursor OUT empcur,
    indeptno IN NUMBER,
    p_errorcode OUT NUMBER);

  FUNCTION GetDept(inempno IN NUMBER,
    p_errorcode OUT NUMBER)
  RETURN empcur;
END Employees;

CREATE OR REPLACE PACKAGE BODY Employees AS

  PROCEDURE GetEmpRecords(p_cursor OUT empcur,
    q_cursor OUT empcur,
    indeptno IN NUMBER,
    p_errorcode OUT NUMBER) IS

    BEGIN
      p_errorcode := 0;
      OPEN p_cursor FOR
        SELECT *
        FROM emp
        WHERE deptno = indeptno
ORDER BY empno;

OPEN q_cursor FOR
SELECT empno
FROM emp
WHERE deptno = indeptno
ORDER BY empno;

EXCEPTION
WHEN OTHERS THEN
  p_errorcode := SQLCODE;
END GetEmpRecords;

FUNCTION GetDept(inempno IN NUMBER,
  p_errorcode OUT NUMBER)
RETURN empcur IS
  p_cursor empcur;
BEGIN
  p_errorcode := 0;
  OPEN p_cursor FOR
  SELECT deptno
  FROM emp
  WHERE empno = inempno;
  RETURN (p_cursor);
END GetDept;

END Employees;

ADO Program
Dim Con As New ADODB.Connection
Dim Rst1 As New ADODB.Recordset
Dim Rst2 As New ADODB.Recordset
Dim Rst3 As New ADODB.Recordset
Dim Cmd As New ADODB.Command
Dim Prm1 As New ADODB.Parameter
Dim Prm2 As New ADODB.Parameter
Con.Provider = "OraOLEDB.Oracle"
Con.ConnectionString = "Data Source=MyOraDb;" & 
  "User ID=scott;Password=tiger;"
Con.Open
Cmd.ActiveConnection = Con

' Although Employees.GetEmpRecords() takes four parameters, only
' two need to be bound because Ref cursor parameters are automatically
' bound by the provider.
Set Prm1 = Cmd.CreateParameter("Prm1", adSmallInt, adParamInput, , 30)
Cmd.Parameters.Append Prm1
Set Prm2 = Cmd.CreateParameter("Prm2", adSmallInt, adParamOutput)
Cmd.Parameters.Append Prm2

' Enable PLSQLRSet property
Cmd.Properties ("PLSQLRSet") = TRUE
Stored Procedures returning resultsets must be called using the ODBC escape sequence for calling stored procedures.

```vbnet
cmd.CommandText = "CALL Employees.GetEmpRecords(?, ?)"
```

' Get the first recordset
Set Rst1 = cmd.Execute

' Disable PLSQLRSet property
Cmd.Properties("PLSQLRSet") = FALSE

' Get the second recordset
Set Rst2 = Rst1.NextRecordset

' Just as in a stored procedure, the REF CURSOR return value must not be bound in a stored function.
Prm1.Value = 7839
Prm2.Value = 0

' Enable PLSQLRSet property
Cmd.Properties("PLSQLRSet") = TRUE

' Stored Functions returning resultsets must be called using the ODBC escape sequence for calling stored functions.
Cmd.CommandText = "CALL Employees.GetDept(?, ?)"

' Get the rowset
Set Rst3 = cmd.Execute

' Disable PLSQLRSet
Cmd.Properties("PLSQLRSet") = FALSE

' Clean up
Rst1.Close
Rst2.Close
Rst3.Close

Statement Caching
Statement caching eliminates the need to parse each SQL or PL/SQL statement before execution, by caching server cursors created during the initial statement execution. Subsequent executions of the same statement can reuse the parsed information from the cursor, and then execute the statement without reparsing, for better performance.

To see performance gains from statement caching, Oracle recommends caching only those statements that will be repeatedly executed. Furthermore, SQL or PL/SQL statements should use parameters rather than literal values. This will enable you to take full advantage of statement caching. This is because parsed information from parameterized statements can be reused, even if the parameter values change in subsequent executions. However, if the literal values in the statements are different, the parsed information cannot be reused unless the subsequent statements also have the same literal values.

StmtCacheSize Connection String Attribute
This attribute enables or disables OraOLEDB statement caching. By default, this attribute is set to 0 (disabled). If it is set to a value greater than 0, OraOLEDB statement caching is enabled and the value specifies the maximum number of statements that can be cached for a connection.
After a connection has been cached to the specified maximum cache size, the cursor least recently used is freed to make room to cache the newly-created cursor. This value should not be greater than the value of the OPEN_CURSORS parameter set in the init.ora database configuration file.

**AddToStmtCache Command Property**

This property is relevant only when statement caching is enabled. If statement caching is enabled and this property is set to true (default), then statements are added to the cache when they are executed. If statement caching is disabled or if this property is set to false, then the executed statement is not cached.

**Enabling Statement Caching Through the Registry**

To enable statement caching by default for all OraOLEDB applications running in a system without changing the application, set the registry key of `\HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\KEY_HOMENAME\OLEDB\StmtCacheSize` to a value greater than 0. Here, HOMENAME refers to the appropriate Oracle home. This value specifies the number of cursors that are to be cached on the server. By default, it is set to 10.

**Connections and Statement Caching**

Statement caching is managed separately for each connection. Therefore, for running the same statement on different connections, you need to parse once for each connection and cache a separate cursor for each connection.

**Metadata Caching**

This feature minimizes the retrieval of metadata for `SELECT` statements by caching the metadata during the initial statement execution. Subsequent executions of the same statement can reuse the cached metadata information for better performance. To see performance gains from metadata caching, Oracle recommends caching only those statements that are executed repeatedly.

---

**Note:** Metadata caching is managed separately for each connection. Therefore, to run the same statement on different connections, the metadata must be cached once for each connection.

---

**Enabling Metadata Caching Through the Connection String Attribute**

The `MetaDataCacheSize` attribute enables or disables OraOLEDB metadata caching. If it is set to a value greater than 0, OraOLEDB metadata caching is enabled and the value specifies the maximum number of statements for which the metadata can be cached for a connection. By default, this attribute is set to 10.

**Enabling Metadata Caching Through the Registry**

To enable metadata caching by default for all OraOLEDB applications running in a system, without changing the application, set the following registry key to a value greater than 0. By default, it is set to 10.

\`\HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\KEY_HOMENAME\ OLEDB\MetaDataCacheSize``

**Command Timeout and Cancel Method**

The `Cancel` method cancels the OraOLEDB command currently being executed. This method can be useful when the application needs to cancel a long running command during times of heavy network traffic or heavy server use.
Alternatively, by using the CommandTimeout property, developers can set a limit to the time that a command executes before OraOLEDB attempts to cancel it. OraOLEDB requires setting the EnableCmdTimeout registry value to 1 to enable CommandTimeout.

When using OLE DB, the default DPBROPCOMMANDTIMOUT is 0 seconds. When using ADO, the default CommandTimeout property is 30 seconds.

Enabling CommandTimeout Through the Registry  The installation adds a registry value called EnableCmdTimeout with the default value set to 0. Setting it to 0 disables command timeout and setting it to 1 enables it. The CommandTimeout property value setting takes effect only when EnableCmdTimeout is set to 1.

The registry value is:
\\HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\KEY_HOMENAME\OLEDB\EnableCmdTimeout

Rowsets

This section discusses using Rowsets with OraOLEDB.

To Create Rowsets

OraOLEDB supports IOpenRowset::OpenRowset and ICommand::Execute for creating rowsets.

To Create Rowsets with IOpenRowset::OpenRowset

When using IOpenRowset::OpenRowset, note the following guidelines:

- The pTableID parameter must contain a DBID structure that specifies a base table or a view.
- The DBID structure's eKind member must be set to DBKIND_GUID_NAME, DBKIND_NAME, or DBKIND_PGUID_NAME.
- The DBID structure's uName member must specify the base table or view name as a Unicode character string. It cannot be NULL.
- The pIndexID parameter of OpenRowset must be NULL.

To Create Rowsets with ICommand::Execute

OraOLEDB supports SQL SELECT statements that return rowsets. OraOLEDB also supports returning rowsets from PL/SQL stored procedures and functions.

By default, ADO creates a nonupdatable rowset from a command object. An updatable rowset can be created by setting the Updatability and IRowsetChange properties on the command object. The Updatability property can be set to the following values:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>update</td>
</tr>
<tr>
<td>2</td>
<td>delete</td>
</tr>
<tr>
<td>3</td>
<td>update and delete</td>
</tr>
<tr>
<td>4</td>
<td>insert</td>
</tr>
<tr>
<td>5</td>
<td>insert and update</td>
</tr>
<tr>
<td>6</td>
<td>insert and delete</td>
</tr>
</tbody>
</table>
The following ADO code sample sets the Updatability property on a command object to allow insert, delete, and update operations on the rowset object.

```vba
Dim Cmd As New ADODB.Command
Dim Rst As New ADODB.Recordset
Dim Con As New ADODB.Connection
...
Cmd.ActiveConnection = Con
Cmd.CommandText = "SELECT * FROM emp"
Cmd.CommandType = adCmdText
cmd.Properties("IRowsetChange") = TRUE
Cmd.Properties("Updatability") = 7
' creates an updatable rowset
Set Rst = cmd.Execute
```

**Updatability**

OraOLEDB supports both immediate and deferred update mode. However, insert and update operations cannot be deferred when the operation changes a nonscalar column, such as LONG, BLOB, or CLOB. When nonscalar column values are changed in a deferred update mode, the entire row is transmitted to the database as though the operation was in an immediate update mode. In addition, these operations cannot be undone with the Undo method (ADO) or IRowsetUpdate::Undo(). However, if they are in a transaction, they can be rolled back with RollbackTrans method (ADO) or ITransactionLocal::Abort().

Rowsets created using queries with joins are updatable by OraOLEDB only with the Client Cursor Engine enabled. C/C++ OLE DB consumers must enable this service to make these rowsets updatable. ADO consumers must specify the CursorLocation as adUseClient to make these rowsets updatable.

For example:

```vba
Dim objCon As New ADODB.Connection
Dim objRst As New ADODB.Recordset

objCon.Provider = "OraOLEDB.Oracle"
objCon.Open "MyOraDb", "scott", "tiger"
objRst.CursorLocation = adUseClient       'ADO Client Cursor
objRst.Open "select ename, dname " & _
  "from emp, dept " & _
  "where emp.deptno = dept.deptno", _
  objCon, adOpenStatic, adLockOptimistic, adCmdText

'Recordset created is updatable. Please note that CursorLocation
'needs to be explicitly set to adUseClient for this join recordset
'to be updatable.
```

**Server Data on Insert Property**

If DBPROP_SERVERDATAGNINSERT (Server Data on Insert) is set to TRUE using OraOLEDB, the consumer can obtain defaults, sequences, and triggered column values from newly inserted and updated rows, if the insert and update operations are made through the rowset.
Having `DBPROP_SERVERDATAONINSERT` set to `TRUE` may degrade performance for both insert and update executions using a rowset because OraOLEDB fetches row data from the database for the newly inserted and updated row. However, if `DBPROP_SERVERDATAONINSERT` is set to its default value of `FALSE`, only the explicitly provided values for insert and update operations are returned when column values are requested for those rows.

If the base table from which the rowset was created does not contain any defaults, sequences, or triggers, then it is highly recommended that `DBPROP_SERVERDATAONINSERT` retain its default value of `FALSE`.

The `DBPROP_SERVERDATAONINSERT` property does not affect the performance of insert and update operations using the command object.

To Search for Rows with `IRowsetFind::FindNext`

OraOLEDB only supports searches performed on `CHAR`, `DATE`, `FLOAT`, `NUMBER`, `RAW`, and `VARCHAR2` columns. Otherwise, `DB_E_NOTSUPPORTED` is returned.

When a search is done with a `NULL` value, only the `DBCOMPAREOPS_EQ` and `DBCOMPAREOPS_NE` compare operations are supported. Otherwise, `DB_E_NOTSUPPORTED` is returned.

OraOLEDB-Specific Connection String Attributes for Rowsets

OraOLEDB-specific connection string attributes which affect the performance of the rowset are:

- **CacheType** - specifies the type of caching used by the provider to store rowset data. OraOLEDB provides two caching mechanisms:
  - Memory - The provider stores all the rowset data in-memory. This caching mechanism provides better performance at the expense of higher memory utilization. The default is Memory.
  - File - The provider stores all the rowset data on disk. This caching mechanism limits memory consumption at the expense of performance.

- **ChunkSize** - This attribute specifies the size, in bytes, of the data in `LONG` and `LONG RAW` columns fetched and stored in the provider cache. Providing a high value for this attribute improves performance, but requires more memory to store the data in the rowset. Valid values are 1 to 65535. The default is 100.

- **FetchSize** - specifies the number of rows the provider will fetch at a time (fetch array). It must be set on the basis of data size and the response time of the network. If the value is set too high, then this could result in more wait time during the execution of the query. If the value is set too low, then this could result in many more round trips to the database. Valid values are 1 to 429,496, and 296. The default is 100.

- **DeferUpdChk** - The `DeferUpdChk` attribute specifies whether or not to defer the updateability check. This supports updating ADO read-only disconnected rowsets. Valid values are 0 (`FALSE`) and 1 (`TRUE`). The default is `FALSE`, which implies that OraOLEDB does not defer the check. If this attribute is not provided at the connection time, then the default registry value is used.

The default attribute values are set in the registry. For more information, see "Default Attribute Values" on page 2-4. The following ADO code example overrides the default attribute values:

```vbnet
Dim con As ADODB.Connection
Set con = New ADODB.Connection
```
con.ConnectionString = "Provider=OraOLEDB.Oracle;User ID=scott;" & _
"Password=tiger;Data Source=MyOraDB;" & _
"FetchSize=200;CacheType=File;"
con.Open

Tips for ADO Programmers

Setting the ADO Rowset property **LockType** to **adLockPessimistic** is not supported by Oracle Provider for OLE DB. If **LockType** is set to **adLockPessimistic**, then OraOLEDB behaves similar to when set as **adLockOptimistic**. This behavior occurs because OraOLEDB does not perform explicit locks on the rows being modified. However, when new data is submitted to the database, the database only performs the update if the rowset data was not already updated by another user, which means that dirty writes are not allowed. **LockType** values **adLockReadOnly**, **adLockBatchOptimistic**, and **adLockOptimistic** are supported by OraOLEDB.

Setting ADO Rowset property **CursorType** to **adOpenKeyset** or **adOpenDynamic** is not supported by Oracle Provider for OLE DB. OraOLEDB does not support either of the two as Oracle supports **Statement Level Read Consistency**, which ensures that the data returned by a query contains only committed data as of the time the query was executed. **CursorType** values **adOpenStatic** and **adOpenForwardOnly** are supported by OraOLEDB.

Schema Rowsets

The schema rowsets available through Oracle Provider for OLE DB are:

- **DBSCHEMA_COLUMNS**
- **DBSCHEMA_INDEXES**
- **DBSCHEMA_SCHEMATA**
- **DBSCHEMA_VIEWS**
- **DBSCHEMA_TABLES**
- **DBSCHEMA_PROVIDER_TYPES** (forward scroll only)
- **DBSCHEMA_FOREIGN_KEYS**
- **DBSCHEMA_PRIMARY_KEYS**
- **DBSCHEMA_PROCEDURES**
- **DBSCHEMA_PROCEDURE_PARAMETERS**

Date Formats

The date format for the Oracle session cannot be set using the **ALTER SESSION SET NLS_DATE_FORMAT** command. In Visual Basic, date formats are controlled by the Regional Settings properties in Windows Control Panel. For more information on Visual Basic date formats, refer to your Visual Basic documentation.

For Oracle Provider for OLE DB, if the **Connection** property **UseSessionFormat** is **FALSE**, which is a default value, then **NLS_DATE_FORMAT** is fixed for the session to 'YYYY-MM-DD HH24:MI:SS' by the provider. If you pass the date to Oracle Database as a string, the date must be in the 'YYYY-MM-DD HH24:MI:SS' format. If **UseSessionFormat** is **TRUE**, then **NLS_DATE_FORMAT** is not fixed by Oracle Provider for OLE DB and the default session **NLS_DATE_FORMAT** is used. For example:

```sql
SELECT * FROM EMP WHERE HIREDATE > '1981-06-15 17:32:12'
```
To use a different format, you need to use the SQL function, \texttt{TO\_DATE()}, to specify the format for dates passed as strings. For example:

\begin{verbatim}
SELECT * FROM EMP WHERE HIREDATE > TO\_DATE('15-JUN-81', 'DD-MON-YY')
\end{verbatim}

However, for dates passed as parameters, the date format is controlled by ADO, which is controlled by the Regional Settings in Windows Control Panel. In this case, \texttt{TO\_DATE()} should not be used. For example:

\begin{verbatim}
Private Sub Command1\_Click()
    Dim objCon As New ADODB\_Connection
    Dim objCmd As New ADODB\_Command
    Dim objRst  As New ADODB\_Recordset
    Dim pDate As New ADODB\_Parameter

    objCon\_Provider = 'OraOLEDB\_Oracle'
    objCon\_Open "MyOraDb", 'scott', 'tiger'
    Set pDate = objCmd\_CreateParameter('pDate', adDate, adParamInput)
    objCmd\_Parameters\_Append pDate
    objCmd\_CommandText = _
        "SELECT * FROM EMP WHERE HIREDATE > ?"
    objCmd\_ActiveConnection = objCon
    objCmd\_CommandType = adCmdText
    pDate\_Value = '06/15/1981'
    Set objRst = objCmd\_Execute

    ...
End Sub
\end{verbatim}

\section*{Case of Object Names}

The names of all objects (tables, columns, views, and so forth) in Oracle Database are case-sensitive. This allows the two objects \texttt{EMP} and \texttt{emp} to exist in the same namespace in the database.

The query, \texttt{SELECT ename FROM emp}, executes correctly even though the table name is \texttt{EMP} (all uppercase) in the database. However, if you want to specify object names in mixed case, you can do so by enclosing the name in double quotes. For example:

\begin{verbatim}
SELECT ename FROM "Emp"
\end{verbatim}

will execute successfully if the table name in the database is \texttt{Emp}. Double quotes preserve the case of the object names in Oracle Database.

\section*{Data Types}

The data types that OraOLEDB supports are listed in Table A–1 with Unicode and NonUnicode mappings.

\textbf{See Also:} For details about these and other data types, and time zones, see \textit{Oracle Database SQL Language Reference}

\subsection*{Binary Data Types}

\texttt{BINARY\_FLOAT} is a single-precision floating point data type (4 bytes), which is mapped to OLE DB \texttt{DBTYPE\_R4}.

\texttt{BINARY\_DOUBLE} is a double-precision floating point data type (8 bytes), which is mapped to OLE DB \texttt{DBTYPE\_R8}.
**TIMESTAMP Data Types**

This section discusses the Timestamp data types and then provides the following:

- Sample data illustrating insertion and retrieval operations using each of the Timestamp data types.
- A Visual Basic code example using the Timestamp data types.

Timestamp data types are mapped to the OLE DB `DBTYPE_DBTIMESTAMP`. The OLE DB `DBTYPE_DBTIMESTAMP` data type does not have `TIME_ZONE` information.

The Timestamp data types include:

- `TIMESTAMP`
- `TIMESTAMP WITH TIME ZONE`
- `TIMESTAMP WITH LOCAL TIME ZONE`

**Data Insertion**

For data insertion into a `TIMESTAMP WITH TIME ZONE` or `TIMESTAMP WITH LOCAL TIME ZONE` column, the time zone setting of the client is used.

OLE DB Timestamp data type cannot provide the time zone information. For insert operations, the default time zone from the client session is added to the `TIMESTAMP WITH TIME ZONE` column data.

**Data Retrieval**

For data retrieval, `TIME_ZONE` is dropped for `TIMESTAMP WITH TIME ZONE` columns, but `TIME_ZONE` is used for `TIMESTAMP WITH LOCAL TIME ZONE` columns.

The OLE DB Timestamp data type cannot store time zone information.

**Fractional Second**

Fractional second is not supported for `TIMESTAMP` data types binding with `Command` objects.

Note that using `ALTER SESSION` to change time zone information does not change the time zone information in the new and existing `Recordsets`, which use the client time zone setting from the Regional options of the operating system. The maximum `fractional_seconds_precision` of `TIMESTAMP` is 9 and the default precision is 6.

**ADO Consumers**

For the Timestamp data types, ADO consumers must specify the value of `CursorLocation` as `adUseServer` and use `Recordset` for DML operations.

**Examples of Timestamp Insert and Retrieval**

The following scenarios assume that the default precision of 6 is used.

**TIMESTAMP Column**

Insert Data: 4/16/2003 11:19:19 AM (No time zone)


Data Retrieval: 4/16/2003 11:19:19 AM

**TIMESTAMP WITH TIME ZONE Column**

Insert Data: 4/16/2003 11:19:19 AM (Time zone of the Client session is used)
Data in DB: 4/16/2003 11.19.19.000000 AM -07:00
Data Retrieval: 4/16/2003 11:19:19 AM (Time zone is dropped)

See Also:
- "Data Insertion" on page 2-21
- "Data Retrieval" on page 2-21

**TIMESTAMP WITH LOCAL TIME ZONE Column**

The following scenario assumes that the time zone of the client session is -04:00, currently on US EDT (Eastern daylight time). For an insert operation, the data in the `TIMESTAMP` column does not include time zone displacement, but its `TIMESTAMP` data is normalized to the database time zone -07:00, which is the same as US PDT (Pacific daylight time).

For a query, data is returned in the time zone of the client session. The time zone displacement is the difference (in hours and minutes) between the local time and the Coordinated Universal Time (UTC).

Insert Data: 4/16/2003 4:30:23 PM (Client time zone is -04:00)
Data in DB: 4/16/2003 01.30.23.000000 PM (Database time zone -07:00)
Data Retrieval: 4/16/2003 4:30:23 PM (Client time zone is -04:00)
Data Retrieval: 4/16/2003 3:30:23 PM (Client time zone is -05:00)
Data Retrieval: 4/16/2003 2:30:23 PM (Client time zone is -06:00)
Data Retrieval: 4/16/2003 1:30:23 PM (Client time zone is -07:00)

**Visual Basic Example**

```vbnet
Dim DT As Date
DT = Now()
con.ConnectionString = "Provider=OraOLEDB.Oracle.1;User ID=user_name;" & _
    "Password=pwd;Data Source=Oracle;"
con.Open
' Must use adUseServer
rec.CursorLocation = adUseServer
rec.ActiveConnection = con
rec.Open "select timestamp_column from test_table", con, adOpenDynamic, adLockOptimistic
rec.AddNew Array("timestamp_column"), Array(DT)
update data
rec.Update Array("timestamp_column"), Array("07/07/07 07:17:17 AM")
...
```

**INTERVAL Data Types**

The INTERVAL data types are mapped to OLE DB `DBTYPE_STR` data type. The INTERVAL data types include:

- INTERVAL YEAR TO MONTH
- INTERVAL DAY TO SECOND

For the INTERVAL YEAR TO MONTH column, the maximum `year_precision` is 9 and the default is 2. For INTERVAL DAY TO SECOND column, the maximum `day_precision` is 9
and the default is 2 and the maximum fractional_seconds_precision is 9, the default is 6.

---

**Note:** If the sign is not specified, then the default is +.

### INTERVAL YEAR TO MONTH

Usage: (sign) years-months

Examples:

- **2-3**
  2 years and 3 months
- **+2-3**
  2 years and 3 months
- **-2-3**
  negative 2 years and 3 months

### INTERVAL DAY TO SECOND

Usage: (sign) days hours:minutes:seconds.second_fraction

Examples:

- **7 10:20:30.123456**
  7 days, 10 hours, 20 minutes, and 30.123456 seconds
- **+7 10:20:30.123456**
  7 days, 10 hours, 20 minutes, and 30.123456 seconds
- **-7 10:20:30.123456**
  negative 7 days, 10 hours, 20 minutes, and 30.123456 seconds

### Visual Basic Example

```vbnet
... con.ConnectionString = "Provider=OraOLEDB.Oracle.1;User ID=user_name;" & 
  "Password=pwd;Data Source=Oracle;" con.Open 'no restriction on using adUseServer or adUseClient rec.CursorLocation = adUseServer rec.ActiveConnection = con rec.Open "select * from test_table2", con, adOpenDynamic, adLockOptimistic rec.AddNew Array("year_to_month_column", "day_to_second_column"), _ Array("8-1", "3 20:30:10.12") 'update data rec.Update Array("year_to_month_column", "day_to_second_column"), _ Array("2-3", "7 10:20:30.123456") ...
```

### LOB Support

The ISequentialStream interface is supported for all LONG, LONG RAW, and LOB (BLOB, CLOB, NCLOB, and BFILE) columns. The consumer can use this interface to read and write to all the LOB columns, except BFILE which is read-only. To have read and write
access to these columns, the SELECT SQL statement used to create the rowset should not contain a join.

---

**Note:** Although most of the LOB columns in an Oracle Database support LOBs over 2 GB of data storage, ADO limits the maximum column size to 2 GB.

---

Columns having the BFILE data type are not updatable in the Rowset interface. However, these columns can be updated using the command interface, if the update is limited to modifying the directory and name of the external file pointed to by the BFILE column. For example:

```sql
INSERT INTO topomaps (areanum, topomap)
VALUES (158, BFILENAME('mapdir', 'topo158.tps'))
```

For more information on LOBs, see Oracle Database SecureFiles and Large Objects Developer's Guide.

---

**Unicode Support**

OraOLEDB supports the Unicode character set. Using this feature, consumers can use OraOLEDB to access data in multiple languages on the same client computer. It can be especially useful in creating global Internet applications supporting as many languages as the Unicode standard entails. For example, you can write a single Active Server Page (ASP) application that accesses an Oracle Database to dynamically generate contents in Japanese, Arabic, English, Thai, and so on.

**Types of Unicode Encoding**

The Oracle Databases store the Unicode data in the UTF-8 encoding scheme, which is an ASCII compatible multibyte encoding of Unicode. Microsoft Windows uses the UTF-16 encoding, which is a 2-byte fixed-width encoding scheme. OraOLEDB transparently converts the data between the two encoding schemes allowing the consumers to deal with only UTF-16.

---

**Note:** The Unicode support is transparent to ADO consumers. OLE DB consumers using C or C++ need to explicitly specify DBTYPE_WSTR in their data type bindings when Unicode data in involved.

---

**How Oracle Unicode Support Works**

OraOLEDB works in two modes, Unicode mode and nonUnicode mode. When the client character set is not a superset of the server character set or the database character set is a multibyte character set, OraOLEDB automatically enables the Unicode mode. In this mode, OraOLEDB stores the data in its cache in the UTF-16 encoding scheme. The user should ensure that the database's character set is AL32UTF8 to prevent any data loss.

If the client character set is a superset of the server's, then the provider operates in the nonUnicode mode. This mode provides slightly better performance as it does not have to deal with larger character buffers required by the UTF-16 encoding.

The detection of the client's and the server's character set is performed during logon.
Note: OraOLEDB no longer requires the client character set to be set to UTF8 to enable the Unicode mode. The provider still supports such setups but no longer requires it.

See "Data Type Mappings in Rowsets and Parameters" on page A-1 for further information.

Unicode Support Setup

To prevent any data loss, the database character set should be AL32UTF8. Other than this, there is no other setup required for Unicode support.

Database Setup

You must ensure that the Oracle Database is configured to store the data in the AL32UTF8 character set. The character set configuration is typically specified during database creation. To check the character set setting of your database, execute the following query in SQL*Plus:

```
SQL> SELECT parameter, value FROM nls_database_parameters
WHERE parameter = 'NLS_CHARACTERSET';
```

If the character set of your database is not AL32UTF8, you need to create a new database with the AL32UTF8 character set and import your data into it. See Oracle Database Administrator’s Guide for more information.

See Also: Oracle Database Globalization Support Guide for general information

Errors

OLE and COM objects report errors through the HRESULT return code of the object member functions. An OLE/COM HRESULT return code is a bit-packed structure. OLE provides macros that dereference structure members. OraOLEDB exposes IErrorLookup to retrieve information about an error.

All objects support extended error information. For this, the consumer must instantiate the OLE DB Extended Error object followed by calling the method GetErrorDescription() to get the error text.

```csharp
// Instantiate OraOLEDBErrorLookup and obtain a pointer to its IErrorLookup interface
CoCreateInstance(CLSID_OraOLEDBErrorLookup, NULL, CLSCTX_INPROC_SERVER,
                 IID_IErrorLookup, (void **)&pIErrorLookup)
//Call the method GetErrorDescription() to get the full error text
pIErrorLookup->GetErrorDescription()
```

The OraOLEDB provider returns the entire error stack in one text block.

For ADO users, the following example applies:

```vba
Dim oerr As ADODB.Error
For Each oerr In con.Errors
    MsgBox 'Error: ' & oerr.Description & vbCrLf & _
            & 'Source: ' & oerr.Source
Next
```

Note:

OraOLEDB no longer requires the client character set to be set to UTF8 to enable the Unicode mode. The provider still supports such setups but no longer requires it.
OLEDB.NET Data Provider Compatibility

The OLE DB .NET Data Provider can utilize OraOLEDB as the OLE DB Provider for accessing Oracle Database.

To make OraOLEDB compatible with OLE DB .NET Data Provider, set the connection string attribute `OLEDB.NET` to `True`.

Setting the `OLEDB.NET` attribute to `False` disables .NET compatibility.

---

**Note:** The `OLEDB.NET` connection string attribute must not be used in ADO applications.

---

Using the OLEDB.NET Attribute in a Connection String

When using OraOLEDB with the OLE DB .NET Data Provider, the `OLEDB.NET` connection attribute must be set to `True` as shown in the following examples:

// in VB.NET
Dim con As New OleDbConnection()
con.ConnectionString = "Provider=OraOLEDB.Oracle;User Id=scott;" & _
    "Password=tiger;Data Source=Oracle;OLEDB.NET=True;"
con.Open

// in C#
...
OleDbConnection con = new OleDbConnection();
con.ConnectionString = "Provider=OraOLEDB.Oracle;User Id=scott;" +
    "Password=tiger;Data Source=Oracle;OLEDB.NET=true;"
con.Open();

---

Using OraOLEDB Custom Properties

ADO allows OraOLEDB provider-specific properties to be set at the object level. The OraOLEDB-specific properties `SPPrmsLOB` and `NDatatype` can be set as connection string attributes as well as at the command-object level. The `StmtCacheSize` property can be set as a connection string attribute and the `AddToStmtCache` property can be set at the command object level. The following example shows the setting of properties at the command level:

// in VB
Dim cmd as new ADODB.Command
...
cmd.Properties("SPPrmsLOB") = True
cmd.Properties("NDatatype") = True
cmd.Properties("AddToStmtCache") = True
...

However, the OLEDB.NET Data Provider cannot expose OLE DB provider-specific properties at the object level. Therefore, the `SPPrmsLOB` and `NDatatype` properties can only be set as connection string attributes and `AddToStmtCache` property is not supported when OraOLEDB is used by OLE DB .NET Data Provider:

// in VB.NET
Dim con As New OleDbConnection()
con.ConnectionString = "Provider=OraOLEDB.Oracle;User Id=scott;" & _
   "Password=tiger;Data Source=Oracle;OLEDB.NET=True;" & _
   "SPPrmsLOB=False;NDatatype=False;"
con.Open()

Both SPPrmsLOB and NDatatype connection string attributes are set to False by default if they are not specified.

Setting either of these connection string attributes to True incurs additional processing overhead when executing commands with parameters. For this reason, before setting either attribute to True, see "OraOLEDB Custom Properties for Commands" on page 2-8.

Updating Oracle with DataTable Changes

In order for the OleDbDataAdapter.Update() method to properly update Oracle Database with changes made in DataTable, which must contain a primary key of a database table. If the database table does not contain a primary key, the ROWID must be selected explicitly when populating DataTable, so that the ROWID can be used to uniquely identify a row when updating a row in the database.

Do not select the ROWID from database tables that contains a primary key. If ROWID is selected along with a primary key, ROWID will be the only column marked as the primary key.

**See Also:** For further information on using the OLE DB .NET Data Provider
- Microsoft .NET Documentation
- Microsoft .NET Framework Class Library

Using OraOLEDB with Visual Basic

The following simple example illustrates how to use Oracle Provider for OLE DB with ADO in Visual Basic 6.0 to connect to an Oracle Database and execute PL/SQL stored procedures and functions.

Setting Up the Oracle Database

This example assumes that the Oracle Database has the demonstration table EMP under the user account scott. The scott account is included in the Oracle starter database. If the account does not exist on your database, create the account before running the sample program. If your database does not contain the emp table, then you can use the demobld.sql script to create the demonstration tables.

This example also uses exampledb as the database network alias when connecting to the Oracle Database. You must change this network alias to match your system.

**Step 1 Build the Sample Tables:**

1. Start SQL*Plus.
2. Connect as username scott with the password tiger.
3. Run the demobld.sql script:

   SQL> @ORACLE_BASE\ORACLE_HOME\sqlplus\demo\demobld.sql;
After the emp table has been created in the scott account, you need to create the PL/SQL package that contains the stored procedure and function that are run in the Visual Basic example.

**Step 2 Create the PL/SQL package:**
1. Connect as username scott with the password tiger.
2. Create the PL/SQL packages shown in "PL/SQL Package" on page 2-12.

---

**Note:** When creating PL/SQL packages the / character is used as a terminator and must be added on a separate line following each CREATE PACKAGE...END block.

---

**Setting Up the Visual Basic Project**

After the Oracle Database setup is completed, you can create the Visual Basic 6.0 project.

1. Start Visual Basic 6.0 and create a new project.
2. Make sure that the Microsoft ActiveX Data Objects 2.1 Library and Microsoft ActiveX Data Objects Recordset 2.1 Library are included as Project References.

3. Add two command buttons to the form. One of the buttons will run the code to execute the PL/SQL procedure GetEmpRecords. The other will run the code to execute the PL/SQL function GetDept.
4. Add the following code to the Click subroutine of the button that will run the code to execute the PL/SQL procedure `GetEmpRecords`.

```vba
Dim Oracon As ADODB.Connection
Dim recset As New ADODB.Recordset
Dim cmd As New ADODB.Command
Dim param1 As New ADODB.Parameter
Dim param2 As New ADODB.Parameter
Dim objErr As ADODB.Error
Dim Message, Title, Default, EmpNoValue

Message = "Enter an employee number (5000 - 9000)"
Title = "Choose an Employee"
Default = "7654"

On Error GoTo err_test
EmpNoValue = InputBox(Message, Title, Default)
If EmpNoValue = "" Then Exit Sub
If EmpNoValue < 5000 Or EmpNoValue > 9000 Then EmpNoValue = 7654

Set Oracon = CreateObject("ADODB.Connection")
Oracon.ConnectionString = "Provider=OraOLEDB.Oracle;" & _
"Data Source=exampledb;" & _
"User ID=scott;" & _
"Password=tiger;"

Oracon.Open
Set cmd.ActiveConnection = Oracon
Set param1 = cmd.CreateParameter("param1", adSmallInt, adParamInput, ,
EmpNoValue)
cmd.Parameters.Append param1
Set param2 = cmd.CreateParameter("param2", adSmallInt, adParamOutput)
cmd.Parameters.Append param2

' Enable PLSQLRSet property
Cmd.Properties ("PLSQLRSet") = TRUE

cmd.CommandText = "{CALL Employees.GetDept(?, ?)}"
Set recset = cmd.Execute

' Disable PLSQLRSet property
Cmd.Properties ("PLSQLRSet") = FALSE
```

Features of OraOLEDB 2-29
Using OraOLEDB with Visual Basic

```vba
MsgBox "Number: " & EmpNoValue & "  Dept: " & recset.Fields("deptno").Value
Exit Sub

err_test:
    MsgBox Error$;
    For Each objErr In Oracon.Errors
        MsgBox objErr.Description
    Next
    Oracon.Errors.Clear
    Resume Next

5. Add the following code to the Click subroutine of the button that will run the code to execute the PL/SQL function GetDept.

Dim Oracon As ADODB.Connection
Dim recset As New ADODB.Recordset
Dim cmd As New ADODB.Command
Dim param1 As New ADODB.Parameter
Dim param2 As New ADODB.Parameter
Dim objErr As ADODB.Error
Dim Message, Title, Default, DeptValue
Message = "Enter a department number (10, 20, or 30)"
Title = "Choose a Department"
Default = "30"

On Error GoTo err_test
DeptValue = InputBox(Message, Title, Default)
If DeptValue = "" Then Exit Sub
If DeptValue < 10 Or DeptValue > 30 Then DeptValue = 30

Set Oracon = CreateObject("ADODB.Connection")
Oracon.ConnectionString = "Provider=OraOLEDB.Oracle;" & _
    "Data Source=exampledb;" & _
    "User ID=scott;" & _
    "Password=tiger;"
Oracon.Open
Set cmd = New ADODB.Command
Set cmd.ActiveConnection = Oracon
Set param1 = cmd.CreateParameter("param1", adSmallInt, adParamInput, ,
    DeptValue)
    cmd.Parameters.Append param1
Set param2 = cmd.CreateParameter("param2", adSmallInt, adParamOutput)
    cmd.Parameters.Append param2

' Enable PLSQLRSet property
Cmd.Properties ("PLSQLRSet") = TRUE

cmd.CommandText = "(CALL Employees.GetEmpRecords(?, ?))"
Set recset = cmd.Execute

' Disable PLSQLRSet property
Cmd.Properties ("PLSQLRSet") = FALSE

Do While Not recset.EOF
    MsgBox "Number: " & recset.Fields("empno").Value & "  Name: " &
        recset.Fields("ename").Value & "  Dept: " & recset.Fields("deptno").Value
    Loop
```

2-30 Oracle Provider for OLE DB Developer's Guide
Exit Sub

err_test:
    MsgBox Error$
    For Each objErr In Oracon.Errors
        MsgBox objErr.Description
    Next
    Oracon.Errors.Clear
    Resume Next

6. Run the project and check the results. For example, if you choose the Get Employee Records by Dept button, then you would see a dialog box requesting that you enter a department number.

![Choose a Department dialog box](image)

After you have entered a department number and clicked **OK**, another dialog box displays employee names and numbers from that department.

![OleDb1 dialog box](image)
This appendix describes OLE DB information that is specific to Oracle Provider for OLE DB. For generic OLE DB information that includes a detailed listing of all OLE DB properties and interfaces, see the Microsoft OLE DB Programmer’s Reference Guide.

This appendix contains these topics:
- Data Type Mappings in Rowsets and Parameters
- Properties Supported
- Interfaces Supported
- MetaData Columns Supported
- OraOLEDB Tracing

### Data Type Mappings in Rowsets and Parameters

This section lists the data type mapping between Oracle data types and OLE DB-defined types. Oracle Provider for OLE DB represents Oracle data types by using certain OLE DB-defined data types in the rowset as well as in parameters. OLE DB-defined types are also mapped to an Oracle data type when creating tables.

Each Oracle data type is mapped to a specific OLE DB data type, as shown in Table A–1. This correspondence is used when data type information is retrieved from an Oracle Database.

<table>
<thead>
<tr>
<th>Oracle Data Type</th>
<th>OLE DB Data Type - Regular (NonUnicode) Mode</th>
<th>OLE DB Data Type - Unicode Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>BFILE</td>
<td>DBTYPE_BYTES</td>
<td>DBTYPE_BYTES</td>
</tr>
<tr>
<td>BINARY_FLOAT</td>
<td>DBTYPE_R4</td>
<td>DBTYPE_R4</td>
</tr>
<tr>
<td>BINARY_DOUBLE</td>
<td>DBTYPE_R8</td>
<td>DBTYPE_R8</td>
</tr>
<tr>
<td>BLOB</td>
<td>DBTYPE_BYTES</td>
<td>DBTYPE_BYTES</td>
</tr>
<tr>
<td>CHAR</td>
<td>DBTYPE_STR</td>
<td>DBTYPE_WSTR</td>
</tr>
<tr>
<td>CLOB</td>
<td>DBTYPE_STR</td>
<td>DBTYPE_WSTR</td>
</tr>
<tr>
<td>DATE</td>
<td>DBTYPE_DBTIMESTAMP</td>
<td>DBTYPE_DBTIMESTAMP</td>
</tr>
<tr>
<td>FLOAT</td>
<td>DBTYPE_R8</td>
<td>DBTYPE_R8</td>
</tr>
<tr>
<td>INTERVAL DAY TO SECOND</td>
<td>DBTYPE_STR</td>
<td>DBTYPE_WSTR</td>
</tr>
<tr>
<td>INTERVAL YEAR TO MONTH</td>
<td>DBTYPE_STR</td>
<td>DBTYPE_WSTR</td>
</tr>
</tbody>
</table>
Properties Supported

This section lists the properties supported by Oracle Provider for OLE DB. The read/write status and initial values are noted.

- Data Source Properties
- DataSourceInfo Properties
- Initialization and Authorization Properties
- Session Properties
- Rowset Properties

Data Source Properties

Table A–2 lists data source properties.

### Table A–2  DBPROPSET_DATASOURCE Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Status</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBPROP_CURRENTCATALOG</td>
<td>READ-ONLY</td>
<td>NULL</td>
</tr>
</tbody>
</table>

DataSourceInfo Properties

Table A–3 lists DataSourceInfo properties.

Table A–1 (Cont.) Data Type Mappings

<table>
<thead>
<tr>
<th>Oracle Data Type</th>
<th>OLE DB Data Type - Regular (NonUnicode) Mode</th>
<th>OLE DB Data Type - Unicode Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>LONG</td>
<td>DBTYPE_STR</td>
<td>DBTYPE_WSTR</td>
</tr>
<tr>
<td>LONG RAW</td>
<td>DBTYPE_BYTES</td>
<td>DBTYPE_BYTES</td>
</tr>
<tr>
<td>NCHAR</td>
<td>DBTYPE_STR</td>
<td>DBTYPE_WSTR</td>
</tr>
<tr>
<td>NCLOB</td>
<td>DBTYPE_STR</td>
<td>DBTYPE_WSTR</td>
</tr>
<tr>
<td>NUMBER</td>
<td>DBTYPE_VARNUMERIC</td>
<td>DBTYPE_VARNUMERIC</td>
</tr>
<tr>
<td>NUMBER (p, s)</td>
<td>DBTYPE_NUMERIC</td>
<td>DBTYPE_NUMERIC</td>
</tr>
<tr>
<td>NVARCHAR2</td>
<td>DBTYPE_STR</td>
<td>DBTYPE_WSTR</td>
</tr>
<tr>
<td>RAW</td>
<td>DBTYPE_BYTES</td>
<td>DBTYPE_BYTES</td>
</tr>
<tr>
<td>ROWID</td>
<td>DBTYPE_STR</td>
<td>DBTYPE_STR</td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td>DBTYPE_DBTIMESTAMP</td>
<td>DBTYPE_DBTIMESTAMP</td>
</tr>
<tr>
<td>TIMESTAMP WITH TIME ZONE</td>
<td>DBTYPE_DBTIMESTAMP</td>
<td>DBTYPE_DBTIMESTAMP</td>
</tr>
<tr>
<td>TIMESTAMP WITH LOCAL TIME ZONE</td>
<td>DBTYPE_DBTIMESTAMP</td>
<td>DBTYPE_DBTIMESTAMP</td>
</tr>
<tr>
<td>VARCHAR</td>
<td>DBTYPE_STR</td>
<td>DBTYPE_WSTR</td>
</tr>
<tr>
<td>Property</td>
<td>Status</td>
<td>Initial Value</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>------------</td>
<td>----------------------------------------------------------------</td>
</tr>
<tr>
<td>DBPROP_ACTIVESESSIONS</td>
<td>READ-ONLY</td>
<td>0, Unlimited sessions</td>
</tr>
<tr>
<td>DBPROP_ASYNCTXNABORT</td>
<td>READ-ONLY</td>
<td>VARIANT_FALSE</td>
</tr>
<tr>
<td>DBPROP_ASYNCTXNCOMMIT</td>
<td>READ-ONLY</td>
<td>VARIANT_FALSE</td>
</tr>
<tr>
<td>DBPROP_BYREFACCESSORS</td>
<td>READ-ONLY</td>
<td>VARIANT_TRUE</td>
</tr>
<tr>
<td>DBPROP_CATALOGLOCATION</td>
<td>READ-ONLY</td>
<td>DBPROPVAL_CL_END</td>
</tr>
<tr>
<td>DBPROP_CATALOGTERM</td>
<td>READ-ONLY</td>
<td>&quot;Database link&quot;</td>
</tr>
<tr>
<td>DBPROP_DATASOURCENAME</td>
<td>READ-ONLY</td>
<td>&quot; &quot;, set at run time</td>
</tr>
<tr>
<td>DBPROP_DATASOURCEREADONLY</td>
<td>READ-ONLY</td>
<td>VARIANT_FALSE</td>
</tr>
<tr>
<td>DBPROP_DBMSNAME</td>
<td>READ-ONLY</td>
<td>&quot; &quot;, set at run time</td>
</tr>
<tr>
<td>DBPROP_DBMSVER</td>
<td>READ-ONLY</td>
<td>set at run time</td>
</tr>
<tr>
<td>DBPROP_DSOTHREADMODEL</td>
<td>READ/WRITE</td>
<td>DBPROPVAL_RT_FREETHREAD</td>
</tr>
<tr>
<td>DBPROP_GROUPBY</td>
<td>READ-ONLY</td>
<td>DBPROPVAL_GB_CONTAINS_SELECT</td>
</tr>
<tr>
<td>DBPROP_HETEROGENEOUSTABLES</td>
<td>READ-ONLY</td>
<td>DBPROPVAL_HT_DIFFERENT_CATALOGS</td>
</tr>
<tr>
<td>DBPROP_IDENTIFIERCASE</td>
<td>READ-ONLY</td>
<td>DBPROPVAL_IC_UPPER</td>
</tr>
<tr>
<td>DBPROP_MAXINDEXSIZE</td>
<td>READ-ONLY</td>
<td>0, limit unknown - depends on block size</td>
</tr>
<tr>
<td>DBPROP_MAXOPENCHAPTERS</td>
<td>READ-ONLY</td>
<td>0, not supported</td>
</tr>
<tr>
<td>DBPROP_MAXORSINFILTER</td>
<td>READ-ONLY</td>
<td>0, not supported</td>
</tr>
<tr>
<td>DBPROP_MAXROWSIZE</td>
<td>READ-ONLY</td>
<td>0, no limit</td>
</tr>
<tr>
<td>DBPROP_MAXROWSIZEINCLUDESBLOB</td>
<td>READ-ONLY</td>
<td>VARIANT_FALSE</td>
</tr>
<tr>
<td>DBPROP_MAXSORTCOLUMNS</td>
<td>READ-ONLY</td>
<td>0, not supported</td>
</tr>
<tr>
<td>DBPROP_MAXTABLESINSELECT</td>
<td>READ-ONLY</td>
<td>0, no limit</td>
</tr>
<tr>
<td>DBPROP_MULTIPLEPARAMSETS</td>
<td>READ-ONLY</td>
<td>VARIANT_TRUE</td>
</tr>
<tr>
<td>DBPROP_MULTIPLERESULTS</td>
<td>READ-ONLY</td>
<td>DBPROPVAL_MR_SUPPORTED</td>
</tr>
<tr>
<td>DBPROP_MULTIPLESTORAGEOBJECTS</td>
<td>READ-ONLY</td>
<td>VARIANT_FALSE</td>
</tr>
<tr>
<td>DBPROP_MULTITABLEUPDATE</td>
<td>READ-ONLY</td>
<td>VARIANT_FALSE</td>
</tr>
<tr>
<td>DBPROP_NULLCOLLATION</td>
<td>READ-ONLY</td>
<td>DBPROPVAL_NC_HIGH</td>
</tr>
<tr>
<td>DBPROP_OLEOBJECTS</td>
<td>READ-ONLY</td>
<td>DBPROPVAL_OO_BLOB</td>
</tr>
<tr>
<td>DBPROP_ORDERBYCOLUMNSENSELECT</td>
<td>READ-ONLY</td>
<td>VARIANT_FALSE</td>
</tr>
<tr>
<td>DBPROP_OUTPUTPARAMETERAVAILABILITY</td>
<td>READ-ONLY</td>
<td>DBPROPVAL_OA_ATEXECUTE</td>
</tr>
<tr>
<td>DBPROP_PERSISTENTIDTYPE</td>
<td>READ-ONLY</td>
<td>DBPROPVAL_PT_NAME</td>
</tr>
<tr>
<td>DBPROP_PREPAREABORTBEHAVIOR</td>
<td>READ-ONLY</td>
<td>DBPROPVAL_CB_PRESERVE</td>
</tr>
<tr>
<td>DBPROP_PREPARECOMMITBEHAVIOR</td>
<td>READ-ONLY</td>
<td>DBPROPVAL_CB_PRESERVE</td>
</tr>
</tbody>
</table>
Initialization and Authorization Properties

Table A–4 lists initialization and authorization properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Status</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBPROP_AUTH_PERSIST_SENSITIVE_AUTHINFO</td>
<td>READ-ONLY</td>
<td>VARIANT_FALSE</td>
</tr>
<tr>
<td>DBPROP_AUTH_USERID</td>
<td>READ/WRITE</td>
<td>User ID</td>
</tr>
<tr>
<td>DBPROP_INIT_DATASOURCE</td>
<td>READ/WRITE</td>
<td>Connect string</td>
</tr>
<tr>
<td>DBPROP_INIT_HWND</td>
<td>READ/WRITE</td>
<td>Window handle for prompt</td>
</tr>
<tr>
<td>DBPROP_INIT_LCID</td>
<td>READ/WRITE</td>
<td>LCID of system</td>
</tr>
<tr>
<td>DBPROP_INIT_OLEDBSERVICES</td>
<td>READ/WRITE</td>
<td>DBPROPVAL_OS_ENABLEALL</td>
</tr>
<tr>
<td>DBPROP_INIT_PERSISTSENSITIVE</td>
<td>READ/ONLY</td>
<td>DBPROPVAL_OS_ENABLEALL</td>
</tr>
</tbody>
</table>

Session Properties

Table A–5 lists session properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Status</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBPROP_ROWSETCONVERSIONSONCOMMAND</td>
<td>READ-ONLY</td>
<td>VARIANT_TRUE</td>
</tr>
<tr>
<td>DBPROP_SCHEMAUSAGE</td>
<td>READ-ONLY</td>
<td>DBPROPVAL_SU_DML_STATEMENTS</td>
</tr>
<tr>
<td>DBPROP_SERVERNAME</td>
<td>READ-ONLY</td>
<td>&quot; &quot;, set at run time</td>
</tr>
<tr>
<td>DBPROP_SORTONINDEX</td>
<td>READ-ONLY</td>
<td>VARIANT_FALSE</td>
</tr>
<tr>
<td>DBPROP_SQLSUPPORT</td>
<td>READ-ONLY</td>
<td>DBPROPVAL_SQL_ODBC_MINIMUM</td>
</tr>
<tr>
<td>DBPROP_STRUCTUREDSTORAGE</td>
<td>READ-ONLY</td>
<td>DBPROPVAL_SS_ISEQUENTIAL_STREAM</td>
</tr>
<tr>
<td>DBPROP_SUBQUERIES</td>
<td>READ-ONLY</td>
<td>DBPROPVAL SQ_CORRELATEDSUBQUERIES</td>
</tr>
<tr>
<td>DBPROP_SUPPORTEDTXNDDL</td>
<td>READ-ONLY</td>
<td>DBPROPVAL TC DDL COMMIT</td>
</tr>
<tr>
<td>DBPROP_SUPPORTEDTXNISOLEVELS</td>
<td>READ-ONLY</td>
<td>DBPROPVAL TI CURSORSTABILITY</td>
</tr>
<tr>
<td>DBPROP_SUPPORTEDTXNISORETAIN</td>
<td>READ-ONLY</td>
<td>DBPROPVAL TR DONTCARE</td>
</tr>
<tr>
<td>DBPROP_TABLETERM</td>
<td>READ-ONLY</td>
<td>&quot;Table&quot;</td>
</tr>
<tr>
<td>DBPROP_USERNAME</td>
<td>READ-ONLY</td>
<td>&quot; &quot;, set at run time</td>
</tr>
</tbody>
</table>
### Table A–5  DBPROPSET_SESSION Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Status</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBPROP_SESS_AUTOCOMMITISOLEVELS</td>
<td>READ-ONLY</td>
<td>DBPROPVAL_TI_CURSORSTABILITY</td>
</tr>
</tbody>
</table>

### Rowset Properties

Table A–6 lists rowset properties.

### Table A–6  DBPROPSET_ROWSET Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Status</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBPROP_ABORTPRESERVE</td>
<td>READ/WRITE</td>
<td>VARIANT_TRUE</td>
</tr>
<tr>
<td>DBPROP_ACCESSORORDER</td>
<td>READ-ONLY</td>
<td>DBPROP_AO_RANDOM</td>
</tr>
<tr>
<td>DBPROP_APPENDONLY</td>
<td>READ-ONLY</td>
<td>VARIANT_FALSE</td>
</tr>
<tr>
<td>DBPROP_BLOCKINGSTORAGEOBJECTS</td>
<td>READ-ONLY</td>
<td>VARIANT_FALSE</td>
</tr>
<tr>
<td>DBPROP_BOOKMARKINFO</td>
<td>READ-ONLY</td>
<td>0</td>
</tr>
<tr>
<td>DBPROP_BOOKMARKS</td>
<td>READ/WRITE</td>
<td>VARIANT_TRUE</td>
</tr>
<tr>
<td>DBPROP_BOOKMARKSKIPPED</td>
<td>READ/WRITE</td>
<td>DBPROP_BMK_NUMERIC</td>
</tr>
<tr>
<td>DBPROP_BOOKMARKTYPE</td>
<td>READ-ONLY</td>
<td>DBPROP_BMK_NUMERIC</td>
</tr>
<tr>
<td>DBPROP_CACHEDEFERRED</td>
<td>READ-ONLY</td>
<td>VARIANT_FALSE</td>
</tr>
<tr>
<td>DBPROP_CANFETCHBACKWARDS</td>
<td>READ/WRITE</td>
<td>VARIANT_FALSE</td>
</tr>
<tr>
<td>DBPROP_CANHOLDROWS</td>
<td>READ/WRITE</td>
<td>VARIANT_FALSE</td>
</tr>
<tr>
<td>DBPROP_CANSCROLLBACKWARDS</td>
<td>READ/WRITE</td>
<td>VARIANT_FALSE</td>
</tr>
<tr>
<td>DBPROP_CHANGEINSERTEDROWS</td>
<td>READ-ONLY</td>
<td>VARIANT_FALSE</td>
</tr>
<tr>
<td>DBPROP_CLIENTCURSOR</td>
<td>READ/WRITE</td>
<td>VARIANT_TRUE</td>
</tr>
<tr>
<td>DBPROP_COLUMNRESTRICT</td>
<td>READ-ONLY</td>
<td>VARIANT_TRUE</td>
</tr>
<tr>
<td>DBPROP_COMMANDTIMEOUT</td>
<td>READ/WRITE</td>
<td>0</td>
</tr>
<tr>
<td>DBPROP_COMMITPRESERVE</td>
<td>READ/WRITE</td>
<td>VARIANT_TRUE</td>
</tr>
<tr>
<td>DBPROP_DEFERRED</td>
<td>READ-ONLY</td>
<td>VARIANT_TRUE</td>
</tr>
<tr>
<td>DBPROP_DELAYSTORAGEOBJECTS</td>
<td>READ-ONLY</td>
<td>VARIANT_TRUE, no delayed update</td>
</tr>
<tr>
<td>DBPROP_FINDCOMPAREOPS</td>
<td>READ-ONLY</td>
<td>DBPROPVAL_CO_EQUALITY</td>
</tr>
<tr>
<td>DBPROP_HIDDENCOLUMNS</td>
<td>READ-ONLY</td>
<td>0</td>
</tr>
<tr>
<td>DBPROP_IACCESSOR</td>
<td>READ-ONLY</td>
<td>VARIANT_TRUE</td>
</tr>
<tr>
<td>DBPROP_ICOLUMNSINFO</td>
<td>READ-ONLY</td>
<td>VARIANT_TRUE</td>
</tr>
<tr>
<td>DBPROP_ICOLUMNSROWSET</td>
<td>READ/WRITE</td>
<td>VARIANT_TRUE</td>
</tr>
<tr>
<td>DBPROP_ICONNECTIONPOINTCONTAINER</td>
<td>READ-ONLY</td>
<td>VARIANT_TRUE</td>
</tr>
<tr>
<td>DBPROP_ICONVERTTYPE</td>
<td>READ-ONLY</td>
<td>VARIANT_TRUE</td>
</tr>
<tr>
<td>DBPROP_IMMOMOBILEROWS</td>
<td>READ-ONLY</td>
<td>VARIANT_TRUE</td>
</tr>
</tbody>
</table>
### Table A-6 (Cont.) DBPROPSET_ROWSET Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Status</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBPROP_IMULTIPLIERESULTS</td>
<td>READ/WRITE</td>
<td>VARIANT_TRUE</td>
</tr>
<tr>
<td>DBPROP_IROWSET</td>
<td>READ-ONLY</td>
<td>VARIANT_TRUE</td>
</tr>
<tr>
<td>DBPROP_IROWSETCHANGE</td>
<td>READ/WRITE</td>
<td>VARIANT_FALSE</td>
</tr>
<tr>
<td>DBPROP_IROWSETFIND</td>
<td>READ/WRITE</td>
<td>VARIANT_FALSE</td>
</tr>
<tr>
<td>DBPROP_IROWSETIDENTITY</td>
<td>READ-ONLY</td>
<td>VARIANT_TRUE</td>
</tr>
<tr>
<td>DBPROP_IROWSETINFO</td>
<td>READ-ONLY</td>
<td>VARIANT_TRUE</td>
</tr>
<tr>
<td>DBPROP_IROWSETLOCATE</td>
<td>READ/WRITE</td>
<td>VARIANT_FALSE</td>
</tr>
<tr>
<td>DBPROP_IROWSETREFRESH</td>
<td>READ/WRITE</td>
<td>VARIANT_FALSE</td>
</tr>
<tr>
<td>DBPROP_IROWSETSCROLL</td>
<td>READ/WRITE</td>
<td>VARIANT_FALSE</td>
</tr>
<tr>
<td>DBPROP_IROWSETUPDATE</td>
<td>READ/WRITE</td>
<td>VARIANT_FALSE</td>
</tr>
<tr>
<td>DBPROP_ISEQUENTIALSTREAM</td>
<td>READ/WRITE</td>
<td>VARIANT_TRUE</td>
</tr>
<tr>
<td>DBPROP_ISUPPORTERRORINFO</td>
<td>READ/WRITE</td>
<td>VARIANT_TRUE</td>
</tr>
<tr>
<td>DBPROP_LITERALBOOKMARKS</td>
<td>READ-ONLY</td>
<td>VARIANT_FALSE</td>
</tr>
<tr>
<td>DBPROP_LITERALIDENTITY</td>
<td>READ-ONLY</td>
<td>VARIANT_FALSE</td>
</tr>
<tr>
<td>DBPROP_LOCKMODE</td>
<td>READ-ONLY</td>
<td>DBPROPVAL_LM_NONE</td>
</tr>
<tr>
<td>DBPROP_MAXOPENROWS</td>
<td>READ/WRITE</td>
<td>0, No limit</td>
</tr>
<tr>
<td>DBPROP_MAXPENDINGROWS</td>
<td>READ-ONLY</td>
<td>0, No limit</td>
</tr>
<tr>
<td>DBPROP_MAXROWS</td>
<td>READ/WRITE</td>
<td>0</td>
</tr>
<tr>
<td>DBPROP_MAXROWSIZE</td>
<td>READ-ONLY</td>
<td>0</td>
</tr>
<tr>
<td>DBPROP_MAXROWSIZEINCLUDESBLOB</td>
<td>READ-ONLY</td>
<td>VARIANT_FALSE</td>
</tr>
<tr>
<td>DBPROP_NOTIFICATIONGRANULARITY</td>
<td>READ/WRITE</td>
<td>DBPROPVAL_NT_MULTIPLEROWS</td>
</tr>
<tr>
<td>DBPROP_NOTIFICATIONPHASES</td>
<td>READ/WRITE</td>
<td>DBPROPVAL_NP_OKTODO</td>
</tr>
<tr>
<td>DBPROP_NOTIFYCOLUMNSET</td>
<td>READ/WRITE</td>
<td>DBPROPVAL_NP_OKTODO</td>
</tr>
<tr>
<td>DBPROP_NOTIFYROWDELETE</td>
<td>READ/WRITE</td>
<td>DBPROPVAL_NP_OKTODO</td>
</tr>
<tr>
<td>DBPROP_NOTIFYROWFIRSTCHANGE</td>
<td>READ/WRITE</td>
<td>DBPROPVAL_NP_OKTODO</td>
</tr>
<tr>
<td>DBPROP_NOTIFYROWINSERT</td>
<td>READ/WRITE</td>
<td>DBPROPVAL_NP_OKTODO</td>
</tr>
</tbody>
</table>
## Properties Supported

### Table A–6 (Cont.) DBPROPSET_ROWSET Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Status</th>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBPROP_NOTIFYROWRESYNCH</td>
<td>READ/WRITE</td>
<td>DBPROPVAL_np_OkTodo</td>
</tr>
<tr>
<td>DBPROP_NOTIFYROWSETRELEASE</td>
<td>READ/WRITE</td>
<td>DBPROPVAL_np_OkTodo</td>
</tr>
<tr>
<td>DBPROP_NOTIFYROWSETFETCPOSITIONCHANGE</td>
<td>READ/WRITE</td>
<td>DBPROPVAL_np_OkTodo</td>
</tr>
<tr>
<td>DBPROP_NOTIFYROWUNDOCHANGE</td>
<td>READ/WRITE</td>
<td>DBPROPVAL_np_OkTodo</td>
</tr>
<tr>
<td>DBPROP_NOTIFYROWUNDODELETE</td>
<td>READ/WRITE</td>
<td>DBPROPVAL_np_OkTodo</td>
</tr>
<tr>
<td>DBPROP_NOTIFYROWUNDOINSERT</td>
<td>READ/WRITE</td>
<td>DBPROPVAL_np_OkTodo</td>
</tr>
<tr>
<td>DBPROP_NOTIFYROWUNDOUPDATE</td>
<td>READ/WRITE</td>
<td>DBPROPVAL_np_OkTodo</td>
</tr>
<tr>
<td>DBPROP_ORDEREDBOOKMARKS</td>
<td>READ-ONLY</td>
<td>VARIANT_TRUE</td>
</tr>
<tr>
<td>DBPROP_OTHERINSERT</td>
<td>READ-ONLY</td>
<td>VARIANT_FALSE</td>
</tr>
<tr>
<td>DBPROP_OOTHERUPDATEDELETE</td>
<td>READ-ONLY</td>
<td>VARIANT_FALSE</td>
</tr>
<tr>
<td>DBPROP_OWNINSERT</td>
<td>READ-ONLY</td>
<td>VARIANT_TRUE</td>
</tr>
<tr>
<td>DBPROP_OWNUPDATEDELETE</td>
<td>READ-ONLY</td>
<td>VARIANT_TRUE</td>
</tr>
<tr>
<td>DBPROP_QUICKRESTART</td>
<td>READ/WRITE</td>
<td>VARIANT_FALSE</td>
</tr>
<tr>
<td>DBPROP_REENTRANTEVENTS</td>
<td>READ-ONLY</td>
<td>VARIANT_FALSE</td>
</tr>
<tr>
<td>DBPROP_REMOVEDDELETED</td>
<td>READ-ONLY</td>
<td>VARIANT_TRUE</td>
</tr>
<tr>
<td>DBPROP_REPORTMULTIPLECHANGES</td>
<td>READ-ONLY</td>
<td>VARIANT_FALSE</td>
</tr>
<tr>
<td>DBPROP_RETURNPENDINGINSERTS</td>
<td>READ/_WRITE</td>
<td>VARIANT_TRUE</td>
</tr>
<tr>
<td>DBPROP_ROWRESTRICT</td>
<td>READ/WRITE</td>
<td>VARIANT_FALSE</td>
</tr>
<tr>
<td>DBPROP_ROWTHREADMODEL</td>
<td>READ-ONLY</td>
<td>DBPROPVAL_RT_FREETHREAD</td>
</tr>
<tr>
<td>DBPROP_SERVERCURSOR</td>
<td>READ/WRITE</td>
<td>VARIANT_FALSE</td>
</tr>
<tr>
<td>DBPROP_SERVERDATAONINSERT</td>
<td>READ/WRITE</td>
<td>VARIANT_TRUE</td>
</tr>
<tr>
<td>DBPROP_STRONGIDENTITY</td>
<td>READ/WRITE</td>
<td>VARIANT_TRUE</td>
</tr>
</tbody>
</table>
Rowset Property Implications

Oracle Provider for OLE DB sets other necessary properties if a particular property is set to VARIANT_TRUE.

- If DBPROP_IROWSETLOCATE is set to VARIANT_TRUE, then the following properties are also set to VARIANT_TRUE:
  - DBPROP_IROWSETIDENTITY
  - DBPROP_CANHOLDROWS
  - DBPROP_BOOKMARKS
  - DBPROP_CANFETCHBACKWARDS
  - DBPROP_CANS CROLLBACKWARDS

- If DBPROP_IROWSETSCROLL is set to VARIANT_TRUE, then the following properties are also set to VARIANT_TRUE:
  - DBPROP_IROWSETIDENTITY
  - DBPROP_IROWSETLOCATE
  - DBPROP_CANHOLDROWS
  - DBPROP_BOOKMARKS
  - DBPROP_CANFETCHBACKWARDS
  - DBPROP_CANS CROLLBACKWARDS

- If DBPROP_IROWSETUPDATE is set to VARIANT_TRUE, then the DBPROP_IROWSETCHANGE property is also set to VARIANT_TRUE.

Interfaces Supported

This section identifies the OLE DB interfaces that are supported by Oracle Provider for OLE DB.

- Data Source
- Session
- Command
- Rowset
- Multiple Results
- Transaction Options
- Custom Error Object
Data Source

CoType TDataSource {
  interface IDbCreateSession;
  interface IDbInitialize;
  interface IDbProperties;
  interface IPersist;
  interface IDbInfo;
  interface ISupportErrorInfo;
}

Session

CoType TSession {
  interface IGetDataSource;
  interface IOpenRowset;
  interface ISessionProperties;
  interface IDbCreateCommand;
  interface IdbschemaRowset;
  interface ISupportErrorInfo;
  interface ITransactionJoin;
  interface ITransactionLocal;
  interface ITransaction;
}

Command

CoType TCommand {
  interface IAccessor;
  interface IColumnsInfo;
  interface ICommand;
  interface ICommandProperties;
  interface ICommandText;
  interface IConvertType;
  interface IColumnsRowset;
  interface ICommandPrepare;
  interface ICommandWithParameters;
  interface ISupportErrorInfo;
}

Rowset

CoType TRowset {
  interface IAccessor;
  interface IColumnsInfo;
  interface IConvertType;
  interface IRowset;
  interface IRowsetInfo;
  interface IColumnsRowset;
  interface IConnectionPointContainer;
  interface IRowsetChange;
  interface IRowsetFind;
  interface IRowsetIdentity;
  interface IRowsetLocate;
  interface IRowsetRefresh;
  interface IRowsetScroll;
}
interface IRowsetUpdate;
interface ISupportErrorInfo;
}

Multiple Results

CoType TMultipleResults {
    interface IMultipleResults;
    interface ISupportErrorInfo;
}

Transaction Options

CoType TTransactionOptions {
    interface ITransactionOptions;
    interface ISupportErrorInfo;
}

Custom Error Object

CoType TCustomErrorObject {
    interface IErrorLookup;
}

MetaData Columns Supported

DBTYPE_BASECOLUMNNAME, DBTYPE_BASETABLENAME, and DBTYPE_BASESCHEMANAME metadata columns are not populated for read-only recordsets. OraOLEDB creates a read-only recordset for server cursor for SQL queries with DISTINCT or UNIQUE keywords. OraOLEDB also creates a read-only recordset for server cursor for JOIN queries.

The following metadata columns are supported by the column rowset of OraOLEDB:

- DBCOLUMN_IDNAME
- DBCOLUMN_PROPID
- DBCOLUMN_NAME
- DBCOLUMN_NUMBER
- DBCOLUMN_TYPE
- DBCOLUMN_TYPEINFO
- DBCOLUMN_COLUMNSIZE
- DBCOLUMN_PRECISION
- DBCOLUMN_SCALE
- DBCOLUMN_FLAGS
- DBCOLUMN_BASECATALOGNAME
- DBCOLUMN_BASECOLUMNNAME
- DBCOLUMN_BASESCHEMANAME
OraOLEDB Tracing

OraOLEDB provides the ability to trace the interface calls for debugging purposes. This feature has been provided to assist Oracle Support Services in debugging customer issues.

The provider can be configured to record the following information:

- For OLE DB Interface method entry and exit:
  - Parameter values supplied (entry)
  - Return value; HRESULT (exit)
  - Thread ID (entry and exit)
- For Distributed transaction enlistment and delistment:
  - Session object information
  - Transaction ID

**Note:** To record global transaction enlistment and delistment information, the TraceLevel value must be set to session object. See "TraceLevel" on page A-12.

Registry Setting for Tracing Calls

To trace the interface calls, you must configure the following registry values for HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE\KEY_HOMENAME\OLEDB:\n
- **TraceFileName**
  
  Valid Value: Any valid path and file name

  *TraceFileName* specifies the file name that is to be used for logging trace information. If TraceOption is set to 0, the name is used as is. However, if TraceOption is 1, then the thread ID is appended to the file name provided. See "TraceOption" for more information.

- **TraceCategory**
  
  Valid Values:
  - 0 = None
  - 1 = OLEDB Interface method entry
  - 2 = OLEDB Interface method exit
  - 4 = Distributed Transaction Enlistment and Delistment
TraceCategory specifies the information that is to be traced. Combinations of different tracing categories can be made by adding the valid values. For example, set TraceCategory to 3 to trace all OLE DB interface method entries and exits.

- **TraceLevel**
  
  **Valid Values:**
  
  - 0 = None
  - 1 = Data Source object
  - 2 = Session object
  - 4 = Command object
  - 8 = Rowset object
  - 16 = Error object
  - 64 = Multiple Results Object

  TraceLevel specifies the OLE DB objects to be traced. Because tracing all the entry and exit calls for all the OLE DB objects can be excessive, TraceLevel is provided to limit tracing to a single or multiple OLE DB objects. To obtain tracing on multiple objects, add the valid values. For example, if TraceLevel is set to 12 and TraceCategory is set to 3, the trace file will only contain method entry and exit for Command and Rowset objects.

  The TraceLevel value must be set to session object (2) to trace global transaction enlistment and delistment information.

- **TraceOption**
  
  **Valid Values:**
  
  - 0 = Single trace file
  - 1 = Multiple trace files

  TraceOption specifies whether to log trace information in single or multiple files for each Thread ID. If a single trace file is specified, the file name specified in TraceFileName is used. If multiple trace file is requested, a Thread ID is appended to the file name provided to create a trace file for each thread.
Component Object Model (COM)
A binary standard that enables objects to interact with other objects, regardless of
the programming language that each object was written in.

consumer
A consumer is any application or tool that calls to a data source or the interfaces of
provider to access data. See provider.

Oracle Net Services
The Oracle client/server communication software that offers transparent operation
to Oracle tools or databases over any type of network protocol and operating
system.

PL/SQL
Procedural language extension to SQL provided by Oracle.

provider
A provider is an interface or set of components that provides data to a consumer. As
the term is used with Oracle Provider for OLE DB, a data provider is a set of COM
components that transfer data from a data source to a consumer, by placing the data
in a tabular format when called for. See consumer.

stored procedure
A stored procedure is a PL/SQL block that are stored in an Oracle Database and can
be called by name from an application.


Index

A
ActiveX Data Objects, vii
ADO, 2-2, 2-6, 2-13
ADO Applications with OLE DB Services, 2-2
ADO.NET, 2-26
attributes
  connection string, 2-3

C
C#
  connection string, 2-26
  example, 2-26
C++/COM, 2-2
C++/COM Applications with OLE DB Services, 2-2
CacheType
  connection string attribute for rowsets, 2-3, 2-18
  caching, 2-15
  Cancel method, 2-15
  case of object names, 2-20
ChunkSize
  connection string attribute for rowsets, 2-3, 2-18
  class ID
    CLSID_OraOLEDB, 2-1
    CLSCTX_INPROC_SERVER macro, 2-1
    CLSID_MSDAINITIALIZE class, 2-2
CoCreateInstance
  creating an instance of the data source object, 2-1
  columns
    metadata, A-10
COM
  Component Object Model, vii
  commands, 2-8
  parameters, 2-8
  CommandTimeout property, 2-15
  compatibility with OLE DB Services, 2-2
Component Certifications
  My Oracle Support, 1-4
Component Object Model (COM), vii
  connecting
    Oracle databases supported, 2-2
    to a specific database, 2-2
    to an Oracle database, 2-3
    to an Oracle database using ADO, 2-6
  connection string attributes, 2-3
    defaults, 2-4
    registry, 2-4
    rowsets, 2-18
  consumers
    OLE DB, 1-2
    creating
      an instance of the data source object, 2-1
      rowsets, 2-16
    Cursor Stability, 2-8
    CursorType
      tips for ADO programmers, 2-19
      custom error objects
        interfaces supported, A-10

D
data source
  creating an instance of, 2-1
  distributed transactions, 2-5
  objects, 2-1
  properties, A-2
data source info
  properties, A-2
DataTable, 2-27
datatypes
  mappings between Oracle datatypes and OLE DB types, A-1
  mappings in rowsets and parameters, A-1
  OLE DB, A-1
  Oracle, A-1
date formats
  NLS_DATE_FORMAT, 2-19
  settings, 2-19
DBNotificationPort, 2-4, 2-5
DBNotifications, 2-4, 2-5
DBPROP.AUTH_PASSWORD property
  setting, 2-2
DBPROP.AUTH_USERNAME property
  enabling operating system authentication, 2-6
  setting, 2-2
DBPROP_INIT_DATASOURCE property
  setting, 2-2
DBPROP_INIT_OLEDBSERVICES property, 2-2
DBPROP_INIT_PROMPT property
  setting, 2-2
DBPROP_INIT_PROVIDERSTRING property
  enabling operating system authentication, 2-6
  setting, 2-4
DBPROP_IROWSETUPDATE property
  setting of other properties, A-8
DBPROP_SERVERDATAONINSERT property, 2-17
DBPROPSET_DBINIT property set
  setting properties, 2-2, 2-4
debugging, A-11
  DeferUpdChk, 2-4, 2-18
  connection string attribute to indicate whether to defer updateability, 2-18
demobld.sql, 2-27
design
  OLE DB, 1-1
DistribTX
  connection string attribute for commands, 2-3
Distributed Transactions, 2-5
distributed transactions, 2-8

EnableCmdTimeout registry value, 2-16
Enhanced Failover Capability, 2-5
enlistment, 2-5
distributed transactions, 2-5
echoes
  HRESULT, 2-25
  OLE and COM, 2-25
eamples, 2-26
  connecting to an Oracle database using ADO, 2-6
  stored procedure returning multiple rowsets, 2-12
  using OraOLEDB with Visual Basic, 2-27

FetchSize
  connection string attribute for rowsets, 2-3, 2-18
features
Oracle Provider for OLE DB, 2-1
files
installed on system for Oracle Provider for OLE DB, 1-3
Oracle Provider for OLE DB, 1-3
global transactions, 2-8

HRESULT
  error return code, 2-25

initialization and authorization properties, A-4
installation, 1-3
  files for Oracle Provider for OLE DB, 1-3

interface call traces, A-11
interfaces
custom error objects, A-10
rowsets, A-9
sessions, A-9
  supported by Oracle Provider for OLE DB, A-8
  transaction options, A-10

KEY_HOMENAME, 2-4

LOB support, 2-23
  ISequentialStream interface, 2-23
LockType
tips for ADO programmers, 2-19

MDAC, 1-3
metadata, 2-15
metadata caching, 2-15
metadata columns
  supported by Oracle Provider for OLE DB, A-10
MetaDataCacheSize, 2-4
Microsoft Data Access Components, 1-3
Microsoft Distributed Transaction Coordinator, 2-8
Microsoft Transaction Server, 2-8
MTS, see Microsoft Transaction Server
My Oracle Support, 1-4

NDataType, 2-4
NDatatype, 2-9
.NET, 2-26

object names
case, 2-20
OLE DB
consumers, 1-2
datatypes, A-1
design, 1-1
Microsoft web site, 1-3
providers, 1-2
OLE DB .NET Data Provider, 2-3
compatibility, 2-26
OLE DB Services
  ADO Applications with, 2-2
  C++/COM Applications with, 2-2
  compatibility with, 2-2
  OleDbDataAdapter.Update(), 2-27
  OLEDB.NET, 2-3, 2-26
  operating system authentication, 2-5
  DBPROP_INIT_PROVIDERSTRING, 2-5
Oracle
datatypes, A-1
Oracle Provider for OLE DB
   class ID, 2-1
   features, 2-1
   intended audience, vii
   provider-specific information, A-1
   system requirements, 1-3
Oracle Services for Microsoft Transaction Server, 1-3, 2-8
OracleMetaLink, 1-4
OraOLEDB see Oracle Provider for OLE DB
OraOLEDB sessions, 2-7
OSAuthent
   connection string attribute for data source, 2-3
   enabling operating system authentication, 2-6
P
password expiration
   connection string attribute, 2-6
   PwdChgDlg, 2-6
performance, 2-15
PLSQLRSet, 2-9
   connection string attribute for commands, 2-4
   properties
      data source, A-2
      data source info, A-2
      initialization and authorization, A-4
      rowset, A-5
      rowset implications, A-8
      sessions, A-4
   supported by Oracle Provider for OLE DB, A-2
providers
   OLE DB, 1-2
   PwdChgDlg
      connection string attribute for commands, 2-4
      connection string attribute for data source, 2-6
R
registry
   default attribute values, 2-4
   returning rowsets
      stored procedures and functions, 2-12
ROWID, 2-27
   rowsets, 2-16
   creating, 2-16
   creating with ICommand, 2-16
   creating with IOpenRowset, 2-16
   date formats, 2-19
   interfaces supported, A-9
   properties, A-5
   property implications, A-8
   returning with procedures and functions, 2-12
   schema, 2-19
   searching with IRowsetFind, 2-18
   updatability, 2-17
S
   schema rowsets, 2-19

Server Data on Insert property, 2-17
   sessions
      interfaces supported, A-9
      objects, 2-7
      properties, A-4
   SPPrmDefVal, 2-4
   SPPrmsLOB, 2-4, 2-10
   StmtCacheSize, 2-4
   stored procedures and functions
      executing, 2-8
      returning rowsets, 2-12
   system requirements
      Oracle Provider for OLE DB, 1-3
T
tips
   for ADO programmers, 2-19
   TraceCategory, A-11
   TraceFileName, A-11
   TraceLevel, A-12
   tracing, A-11
   transaction options
      interfaces supported, A-10
   transactions
      distributed, 2-8
      global, 2-8
      isolation levels, 2-8
      local, 2-7
      types supported, 2-7
U
   UCS-2 character set, 2-24
   Unicode, 2-24
   UseSessionFormat
      connection string attribute for commands, 2-4
V
   VB.NET
      connection string, 2-26
      examples, 2-26
   VCharNull, 2-4