



UniVerse

ODBC Guide

Version 10.2
September, 2006

IBM Corporation
555 Bailey Avenue
San Jose, CA 95141

Licensed Materials – Property of IBM

© Copyright International Business Machines Corporation 2006. All rights reserved.

AIX, DB2, DB2 Universal Database, Distributed Relational Database Architecture, NUMA-Q, OS/2, OS/390, and OS/400, IBM Informix®, C-ISAM®, Foundation.2000™, IBM Informix® 4GL, IBM Informix® DataBlade® module, Client SDK™, Cloudscape™, Cloudsync™, IBM Informix® Connect, IBM Informix® Driver for JDBC, Dynamic Connect™, IBM Informix® Dynamic Scalable Architecture™ (DSA), IBM Informix® Dynamic Server™, IBM Informix® Enterprise Gateway Manager (Enterprise Gateway Manager), IBM Informix® Extended Parallel Server™, i.Financial Services™, J/Foundation™, MaxConnect™, Object Translator™, Red Brick® Decision Server™, IBM Informix® SE, IBM Informix® SQL, InformiXML™, RedBack®, SystemBuilder™, U2™, UniData®, UniVerse®, wIntegrate® are trademarks or registered trademarks of International Business Machines Corporation.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries.

Windows, Windows NT, and Excel are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

UNIX is a registered trademark in the United States and other countries licensed exclusively through X/Open Company Limited.

Other company, product, and service names used in this publication may be trademarks or service marks of others.

This product includes cryptographic software written by Eric Young (eay@cryptosoft.com).

This product includes software written by Tim Hudson (tjh@cryptosoft.com).

Documentation Team: Claire Gustafson, Shelley Thompson

US GOVERNMENT USERS RESTRICTED RIGHTS

Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Table of Contents

Preface

Organization of This Manual	viii
Documentation Conventions.	ix
Product Support	xi
UniVerse Documentation.	xii
Related Documentation	xiv
API Documentation	xv

Chapter 1

UniVerse ODBC Overview

How UniVerse ODBC Works	1-3
The UniVerse ODBC Driver	1-3
UniVerse ODBC Client Applications	1-4
Client/Server Connections.	1-5
UniVerse ODBC Driver Conformance Levels	1-6
ODBC API Conformance Levels	1-6
ODBC SQL Grammar Conformance Levels	1-7
Tested ODBC Client Applications	1-9
UniVerse ODBC Limitations	1-10
Microsoft Access 2000.	1-10
Microsoft Query 2000	1-11
Visual Basic	1-12

Chapter 2

Administering UniVerse ODBC Accounts

Making UniVerse Files Accessible to ODBC Clients	2-3
Removing ODBC Access to UniVerse Files	2-4
UniVerse Server Administration Menu	2-5

Chapter 3

Installing and Configuring the UniVerse ODBC Driver

Installing UniVerse ODBC Driver Software	3-3
Configuring UniVerse ODBC Driver Software	3-8

Configuring UCI for UniVerse ODBC Driver	3-17
Uni Call Interface and the uci.config File.	3-17
Changing the MAXFETCHCOLS Parameter	3-17
Changing the MAXFETCHBUFF Parameter	3-18
Changing the UCI Connection Timeout	3-19
ODBC Diagnostic Tool: Dr. DeeBee Spy	3-20
Monitoring Your Configuration	3-20
The Dr. DeeBee Spy Log File	3-22

Chapter 4 Making UniVerse Data Accessible to ODBC Applications

Overview	4-3
Making UniVerse Accounts Accessible	4-3
Making Data in a UniVerse Account Accessible	4-3
Presenting UniVerse Data in ODBC Format	4-4
Accessing UniVerse Schemas and Accounts	4-6
Using ODBC Qualifiers	4-6
Accessing UniVerse Tables, Views, and Files	4-9
Updating a UniVerse Account for ODBC Access	4-9
File Privileges and Permissions	4-10
What the ODBC File Access Utility Does	4-11
Accessing Saved Select Lists	4-14
Accessing Columns and Fields	4-16
ODBC-Accessible Columns and Fields	4-16
Multivalued Columns and Fields	4-16
Modifying UniVerse Data and Data Definitions for ODBC.	4-20
Validating Tables and Files for ODBC Clients	4-20
SQL Data Types	4-22
Length of Character Data.	4-23
Empty or Unconvertible Data	4-25
Conversion Codes	4-26
Association Keys	4-27
Defining Association Keys	4-27
ODBC Name Mapping	4-29

Chapter 5 Troubleshooting

Isolating a Problem	5-4
Before Reporting a Problem	5-5
Getting Information About Your System	5-6
Manual Communications Verification	5-10
Verifying a TCP/IP Connection	5-10

Verifying Other Types of Network Connection	5-11
Connection Problems	5-11
Windows Server-Related Problems.	5-12
UNIX Server-Related Problems.	5-13
Windows and UNIX Server-Related Problems	5-14
Client-Related Problems	5-15

Appendix A ODBC Usage Notes

Data Types in SQL	A-2
Support for Core ODBC SQL Grammar	A-3
Support for Extended ODBC SQL Grammar.	A-4
SELECT...FOR UPDATE Statement	A-4
Outer Joins.	A-4
UNION Clause	A-5
ESCAPE Clause in the LIKE Predicate	A-5
Date and Time Types	A-5
Scalar Functions	A-6
UniVerse Extensions to ODBC SQL Grammar	A-10
NOWAIT Keyword	A-11
EXPLAIN Keyword.	A-11
SQL Usage Notes and Restrictions	A-11
Syntax	A-13
Parameters	A-14
Result Sets.	A-14
Behavior of SQLRowCount	A-15
Limitations.	A-15
Interoperability with Generic ODBC Applications	A-16
Executing UniVerse SQL	A-17
The {NATIVE} Syntax Extension	A-17
Retrying Statements as UniVerse SQL	A-17
SQL-Related Connection Options	A-19
Transaction Support	A-20
Autocommit Mode	A-20
Manual Mode	A-20
ODBC Table Types.	A-22

Appendix B Error Messages

Message Descriptions	B-2
Messages	B-3

Appendix C	ODBC Environment Variables	
Appendix D	Driver Process Log Files	
	Driver Process Log Filenames	D-2
	Saving and Examining the Log File	D-3
	Locating the UniVerse ODBC Driver Log File.	D-4
Appendix E	Sample Accounts	
	Locating the Sample UniVerse Accounts	E-2
	Database Definitions	E-3
	HS.SALES Account	E-3
	HS.SERVICE Account	E-4
	Sample Data	E-6
Index		

Preface

This guide describes how to install, configure, and use the UniVerse ODBC driver for ODBC clients running Windows platforms.

You should be familiar with your operating systems and UniVerse, as well as with any networking software used on your client systems.

Organization of This Manual

This manual is organized as follows:

Chapter 1, “[UniVerse ODBC Overview](#),” provides an overview of the UniVerse database server and UniVerse ODBC driver.

Chapter 2, “[Administering UniVerse ODBC Accounts](#),” describes system requirements for using UniVerse ODBC.

Chapter 3, “[Installing and Configuring the UniVerse ODBC Driver](#),” describes how to install and configure UniVerse ODBC driver software.

Chapter 4, “[Making UniVerse Data Accessible to ODBC Applications](#),” describes how to make UniVerse data accessible to ODBC applications.

Chapter 5, “[Troubleshooting](#),” describes common system malfunctions and how to diagnose them.

Appendix A, “[ODBC Usage Notes](#),” describes ODBC SQL usage.

Appendix B, “[Error Messages](#),” describes UniVerse ODBC error messages.

Appendix C, “[ODBC Environment Variables](#),” describes the UniVerse ODBC environment variables.

Appendix D, “[Driver Process Log Files](#),” describes the driver process log files.

Appendix E, “[Sample Accounts](#),” describes the sample UniVerse accounts created by the installation script.

Documentation Conventions

This manual uses the following conventions:

Convention	Usage
Ctrl+Esc	A plus sign between key names means to hold down the first key while you press the second key. For example, press Ctrl+Esc means to hold down the Ctrl key while you press the Esc key.
Alt, F	A comma between key names means to press one key after the other. For example, Alt, F means to press and release the Alt key, then press and release the F key.
(Alt, F, E)	Keystrokes in parentheses are the keyboard alternatives to the mouse.
→	A right arrow between options indicates to choose each option in sequence. For example, choose File → Exit means to choose File from the menu bar, then choose Exit from the pull-down menu.
UPPERCASE	All uppercase characters other than keys indicate directory names, filenames, and acronyms.
Bold	In syntax, bold indicates commands, function names, keywords, and options that must be entered exactly as shown. In text, bold indicates keys to press, function names, and menu selections.
<i>Italic</i>	In syntax, italic indicates information that you supply. In text, italic also indicates UNIX commands and options, file names, and paths.
Courier	Courier indicates examples of source code and system output.
Courier Bold	In examples, courier bold indicates characters the user types or keys the user presses (for example, <Enter>).

Documentation Conventions

The following are also used:

Syntax definitions and examples are indented for ease in reading.

All punctuation marks included in the syntax—for example, commas, parentheses, or quotation marks—are required unless otherwise indicated.

Syntax lines that do not fit on one line in this manual are continued on subsequent lines. The continuation lines are indented. When entering syntax, type the entire syntax entry, including the continuation lines, on the same input line.

Product Support

If you have questions about this product, first consult the documentation or online help. In some cases, such as for network or database errors, you may need to consult additional documentation or your system administrator for further assistance.

If you are unable to resolve your question, ask your IBM Customer Support contact.

When you write or call, make sure to provide the following information:

- The version number of the UniVerse ODBC driver or of UCI Config Editor.
- For network- or database-related questions, the brand and version of software you are using.
- The type of hardware you are using (client and server).
- The details of what happened and what you were doing when the problem occurred.
- The exact wording of any messages that appeared on your screen.

UniVerse Documentation

UniVerse documentation includes the following:

UniVerse Installation Guide: Contains instructions for installing UniVerse 10.2.

UniVerse BASIC: Contains comprehensive information about the UniVerse BASIC language. It is for experienced programmers.

UniVerse BASIC Commands Reference: Provides syntax, descriptions, and examples of all UniVerse BASIC commands and functions.

UniVerse BASIC Extensions: Describes the following extensions to UniVerse BASIC: UniVerse BASIC Socket API, Using CallHTTP, and Using WebSphere MQ with UniVerse.

UniVerse BASIC SQL Client Interface Guide: Describes how to use the BASIC SQL Client Interface (BCI), an interface to UniVerse and non-UniVerse databases from UniVerse BASIC. The BASIC SQL Client Interface uses ODBC-like function calls to execute SQL statements on local or remote database servers such as UniVerse, DB2, SYBASE, or INFORMIX. This book is for experienced SQL programmers.

Administering UniVerse: Describes tasks performed by UniVerse administrators, such as starting up and shutting down the system, system configuration and maintenance, system security, maintaining and transferring UniVerse accounts, maintaining peripherals, backing up and restoring files, managing file and record locks, and network services. This book includes descriptions of how to use UniAdmin on a Windows client and how to use shell commands on UNIX systems to administer UniVerse.

Using UniAdmin: Describes the UniAdmin tool, which enables you to configure UniVerse, configure and manage servers and databases, and monitor UniVerse performance and locks.

UniVerse Security Features: Describes security features in UniVerse, including configuring SSL through UniAdmin, using SSL with the CallHttp and Socket interfaces, using SSL with UniObjects for Java, and automatic date encryption.

UniVerse Transaction Logging and Recovery: Describes the UniVerse transaction logging subsystem, including both transaction and warmstart logging and recovery. This book is for system administrators.

UniVerse System Description: Provides detailed and advanced information about UniVerse features and capabilities for experienced users. This book describes how to use UniVerse commands, work in a UniVerse environment, create a UniVerse database, and maintain UniVerse files.

UniVerse User Reference: Contains reference pages for all UniVerse commands, keywords, and user records, allowing experienced users to refer to syntax details quickly.

Guide to Retrieve: Describes Retrieve, the UniVerse query language that lets users select, sort, process, and display data in UniVerse files. This book is for users who are familiar with UniVerse.

Guide to ProVerb: Describes ProVerb, a UniVerse processor used by application developers to execute prestored procedures called procs. This book describes tasks such as relational data testing, arithmetic processing, and transfers to subroutines. It also includes reference pages for all ProVerb commands.

Guide to the UniVerse Editor: Describes in detail how to use the Editor, allowing users to modify UniVerse files or programs. This book also includes reference pages for all UniVerse Editor commands.

UniVerse NLS Guide: Describes how to use and manage UniVerse's National Language Support (NLS). This book is for users, programmers, and administrators.

UniVerse SQL Administration for DBAs: Describes administrative tasks typically performed by DBAs, such as maintaining database integrity and security, and creating and modifying databases. This book is for database administrators (DBAs) who are familiar with UniVerse.

UniVerse SQL User Guide: Describes how to use SQL functionality in UniVerse applications. This book is for application developers who are familiar with UniVerse.

UniVerse SQL Reference: Contains reference pages for all SQL statements and keywords, allowing experienced SQL users to refer to syntax details quickly. It includes the complete UniVerse SQL grammar in Backus Naur Form (BNF).

Related Documentation

The following documentation is also available:

UniVerse GCI Guide: Describes how to use the General Calling Interface (GCI) to call subroutines written in C, C++, or FORTRAN from BASIC programs. This book is for experienced programmers who are familiar with UniVerse.

UV/Net II Guide: Describes UV/Net II, the UniVerse transparent database networking facility that lets users access UniVerse files on remote systems. This book is for experienced UniVerse administrators.

UniVerse Guide for Pick Users: Describes UniVerse for new UniVerse users familiar with Pick-based systems.

Moving to UniVerse from PI/open: Describes how to prepare the PI/open environment before converting PI/open applications to run under UniVerse. This book includes step-by-step procedures for converting INFO/BASIC programs, accounts, and files. This book is for experienced PI/open users and does not assume detailed knowledge of UniVerse.

API Documentation

The following books document application programming interfaces (APIs) used for developing client applications that connect to UniVerse and UniData servers.

Administrative Supplement for Client APIs: Introduces IBM's seven common APIs, and provides important information that developers using any of the common APIs will need. It includes information about UniRPC, the UCI Config Editor, the *ud_database* file, and device licensing.

UCI Developer's Guide: Describes how to use UCI (Uni Call Interface), an interface to UniVerse and UniData databases from C-based client programs. UCI uses ODBC-like function calls to execute SQL statements on local or remote UniVerse and UniData servers. This book is for experienced SQL programmers.

IBM JDBC Driver for UniData and UniVerse: Describes UniJDBC, an interface to UniData and UniVerse databases from JDBC applications. This book is for experienced programmers and application developers who are familiar with UniData and UniVerse, Java, JDBC, and who want to write JDBC applications that access these databases.

InterCall Developer's Guide: Describes how to use the InterCall API to access data on UniVerse and UniData systems from external programs. This book is for experienced programmers who are familiar with UniVerse or UniData.

UniObjects Developer's Guide: Describes UniObjects, an interface to UniVerse and UniData systems from Visual Basic. This book is for experienced programmers and application developers who are familiar with UniVerse or UniData, and with Visual Basic, and who want to write Visual Basic programs that access these databases.

UniObjects for Java Developer's Guide: Describes UniObjects for Java, an interface to UniVerse and UniData systems from Java. This book is for experienced programmers and application developers who are familiar with UniVerse or UniData, and with Java, and who want to write Java programs that access these databases.

UniObjects for .NET Developer's Guide: Describes UniObjects, an interface to UniVerse and UniData systems from .NET. This book is for experienced programmers and application developers who are familiar with UniVerse or UniData, and with .NET, and who want to write .NET programs that access these databases.

Using UniOLEDB: Describes how to use UniOLEDB, an interface to UniVerse and UniData systems for OLE DB consumers. This book is for experienced programmers and application developers who are familiar with UniVerse or UniData, and with OLE DB, and who want to write OLE DB programs that access these databases.

UniVerse ODBC Overview

How UniVerse ODBC Works	1-3
The UniVerse ODBC Driver	1-3
UniVerse ODBC Client Applications	1-4
Client/Server Connections	1-5
UniVerse ODBC Driver Conformance Levels	1-6
ODBC API Conformance Levels	1-6
ODBC SQL Grammar Conformance Levels	1-7
Tested ODBC Client Applications	1-9
UniVerse ODBC Limitations	1-10
Microsoft Access 2000	1-10
Microsoft Query 2000	1-11
Visual Basic	1-12

This chapter provides an overview of how a UniVerse ODBC works. It also describes the UniVerse ODBC driver conformance levels.

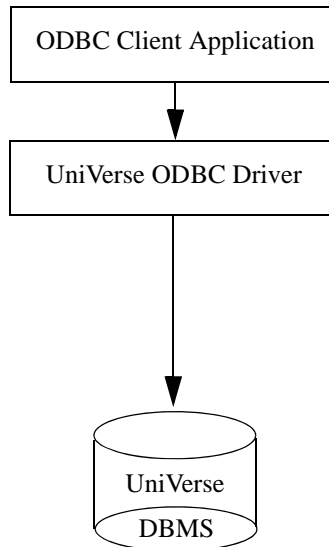
How UniVerse ODBC Works

The UniVerse ODBC driver enables ODBC applications to connect to the UniVerse database management system (DBMS).

The UniVerse ODBC driver is an implementation of the Microsoft ODBC Version 2.0 specification.

An ODBC application sends a connection request for a data source name (DSN) definition to the UniVerse ODBC driver. The driver receives the request and then establishes a connection to the UniVerse DBMS.

The following example shows an overview of the UniVerse ODBC architecture.



UniVerse ODBC Architecture

The UniVerse ODBC Driver

The UniVerse ODBC driver requests a connection to the server through the UniCall Interface (UCI) and UniRPC.

UniVerse ODBC Client Applications

A UniVerse ODBC client application, such as a custom-written ODBC application or a commercial application such as Microsoft Access or PowerBuilder, connects to the UniVerse ODBC driver, which establishes connections to remote data sources and sends database requests to one or more UniVerse database servers. Applications access ODBC data sources that are mapped to UniVerse accounts through entries in the *uci.config* file and related entries in the ODBC section of the registry. The *uci.config* file contains connection parameters necessary to route requests to the appropriate UniVerse database server.

When an application connects to a data source, it reads the *uci.config* file and related entries in the ODBC section of the registry on the client system to determine the host system, user name, and password to use when accessing a particular UniVerse database server.

A UniVerse ODBC application accesses UniVerse databases across various operating systems. Each configuration entry in the *uci.config* file describes the physical attributes of a database in sufficient detail to perform three tasks:

- Establish communications
- Launch a UniVerse database server process
- Route query and update requests

UCI Config Editor Tool and Microsoft ODBC Data Source Administration Tool

The UCI Config Editor tool is contained in the UniVerse ODBC driver. This tool configures your Windows client system for access to data sources on UniVerse database servers. It manages UniVerse ODBC data sources. The UCI Config Editor tool provides an easy way to enter, and edit entries in the configuration file.

Every UniVerse ODBC data source definition must reference an entry in the *uci.config* file.

Use the UCI Config Editor tool to do the following:

- Create or edit a data source. The UCI Config Editor tool can change an entry in the configuration file even if the data source is relocated or the network is altered. This shields application programs from changes in the client/server environment.

- To test UniVerse ODBC data sources, run the Microsoft ODBC Data Source Administrator tool.

For more information about the UCI Config Editor tool, see *Administrative Supplement for Client APIs*.

Client/Server Connections

Client/server interactions require either device licences or connection and user licenses.

Connection and User Licenses

If your organization does not use device licensing, each connection from an ODBC client application to the UniVerse database server requires one UniVerse license. If your application makes multiple simultaneous connections to UniVerse, each one uses one UniVerse license.

For information about device licensing, see *Administering UniVerse*.

UniVerse ODBC Driver Conformance Levels

Conformance levels help application and driver developers by establishing standards of functionality. This section describes how UniVerse ODBC conforms to the *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*. ODBC defines conformance levels for an ODBC driver in two areas:

- ODBC API
- ODBC SQL grammar, including ODBC SQL data types

The following table summarizes ODBC compliance levels for the UniVerse ODBC driver. For more information about ODBC conformance levels, see *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

Driver	ODBC API	ODBC SQL Grammar
UniVerse ODBC	Core, Level 1, and Level 2 features as described in ODBC API Conformance Levels	Minimum, core, and extended features described in ODBC SQL Grammar Conformance Levels

ODBC Conformance Levels

The UniVerse ODBC driver also provides connection and statement options to control additional option values specific to UniVerse ODBC.

ODBC API Conformance Levels

The UniVerse ODBC driver conforms to the ODBC Core API and Level 1 API functionality. In addition, the UniVerse ODBC driver supports the following Level 2 functions:

- `SQLMoreResults`¹
- `SQLNativeSql`
- `SQLNumParams`

Unsupported Level 2 Functions

The UniVerse ODBC driver does not support the following Level 2 functions:

1. `SQLMoreResults` never returns more than one result set or row count.

- **SQLBrowseConnect**
- **SQLColumnPrivileges**
- **SQLDescribeParam**
- **SQLExtendedFetch**
- **SQLForeignKeys**
- **SQLParamOptions**
- **SQLPrimaryKeys**
- **SQLProcedureColumns**
- **SQLProcedures**
- **SQLSetPos**
- **SQLSetScrollOptions**
- **SQLTablePrivileges**

ODBC SQL Grammar Conformance Levels

The UniVerse ODBC driver conforms to almost all core SQL grammar. For a list of exceptions, see [“Support for Core ODBC SQL Grammar.”](#) For detailed information about ODBC SQL grammar conformance in UniVerse ODBC, see [“Support for Extended ODBC SQL Grammar.”](#)

Supported Extended ODBC SQL Grammar

The UniVerse ODBC driver supports some of the extended ODBC SQL grammar features. These include:

- Outer joins
- UNION clause
- SELECT...FOR UPDATE statement
- Scalar functions such as SUBSTRING and CONCAT
- DATE and TIME data types
- Procedure calls

Supported SQL Data Types

UniVerse ODBC supports the following SQL data types:

- CHAR
- DATE
- DECIMAL
- DOUBLE PRECISION
- FLOAT
- INTEGER
- NUMERIC
- REAL
- SMALLINT
- TIME
- VARCHAR

Tested ODBC Client Applications

UniVerse ODBC Driver has been tested with the following Windows ODBC client applications:

- Microsoft Office 2000, including Access, Excel, and Word
- Microsoft Visual Basic Enterprise Edition Versions 6.0
- Microsoft Query 2000
- Cognos Impromptu Version 3.0
- The .NET Framework Data Provider for ODBC using C# and VB.NET CLR languages

UniVerse ODBC Limitations

The following sections describe known UniVerse ODBC limitations with the following clients:

- Microsoft Access 2000
- Microsoft Query 2000
- Visual Basic

Microsoft Access 2000

When running queries in Microsoft Access, performance can be significantly affected if the primary key of a UniVerse table or file is not accessible to UniVerse ODBC. This can occur in either of the following cases:

- If the primary key is excluded from the @SELECT phrase in the dictionary of a table or file.
- If the primary key is excluded and ID.SUP is specified in the @ phrase of a file dictionary containing no @SELECT phrase. (UniVerse ignores the @ phrase in table dictionaries.)

To get the best possible performance from Microsoft Access, we recommend that the primary key be accessible to UniVerse ODBC. For more information, see Chapter 4, [“Making UniVerse Data Accessible to ODBC Applications.”](#)

Microsoft Access has the following functional limitations imposed by its use of the Microsoft Jet database engine:

- Only the first 255 columns of an ODBC table are accessible. All other columns are hidden from the user.
- Access does not allow importing or linking ODBC tables with table or column names longer than 64 characters.

Microsoft Access converts columns defined with a UniVerse SQL data type of DECIMAL or NUMERIC and a precision greater than 15 digits to text fields.

When inserting new rows into a table whose underlying UniVerse representation is an association or an unassociated multivalued field, be sure to enter a value for the @ASSOC_ROW pseudocolumn (with name mapping on, this column is called Z_ASSOC_ROW). Failure to specify a value for @ASSOC_ROW can cause Microsoft Access to abort with an access violation error.

When creating a query in Microsoft Access, you can specify an SQL statement as the basis of the query. Due to a bug in Access, any SQL statement containing the UNION keyword fails to execute. Before submitting the query to ODBC for execution, Microsoft Access rewrites the SQL and puts parentheses around each SELECT statement, which is invalid ODBC SQL syntax. For example, if you submitted the following query to ODBC:

```
SELECT * FROM U_TABLE UNION SELECT * FROM U_TABLE
```

Microsoft Access would rewrite it as:

```
( SELECT * FROM U_TABLE ) UNION ( SELECT * FROM U_TABLE )
```

Microsoft Query 2000

Microsoft Query is accessed through “Get External Data” from Microsoft Excel.

An attempt to connect to a UniVerse account containing a table name or file name longer than 65 characters fails with a General ODBC Error. This is due to a bug in Microsoft Query, which fails to allocate enough memory to accommodate ODBC table names, which can contain up to 128 characters as defined in the ODBC 2.0 specification. Microsoft has confirmed that this is a problem with Query, but is not planning to provide a fix in any future release.

When connecting to a UniVerse account that is not a schema, only UniVerse files referenced by that account are made available by Microsoft Query. All UniVerse tables are excluded from the list of available tables, and the list of available schemas is disabled. Microsoft has confirmed that this is a problem with Query, but is not planning to provide a fix in any future release.

When connecting to a UniVerse schema, only UniVerse tables are made available by Microsoft Query. UniVerse files are excluded from the list of available tables. Due to a bug in Microsoft Query, any ODBC tables reported with an empty string as the ODBC table qualifier are ignored. Since Universe files are not part of a schema, UniVerse ODBC reports their ODBC table qualifier as the empty string, in conformance with the ODBC 2.0 specification. Microsoft has confirmed that this is a problem with Query, but is not planning to provide a fix in any future release.

Microsoft Query provides visibility only to the first 255 columns of an ODBC table. All other columns are hidden from the user.

Microsoft Query does not allow adding tables to a query that have column names longer than 64 characters. Attempts to do so fail with a “Can’t access table” message. However, such tables are accessible by building a query using an SQL statement which SELECTs the table. To execute an SQL statement, choose “Execute SQL...” from the File menu.

We suggest you use Query Designer with Microsoft Query 2000 (available in Microsoft Office 2000) rather than Query Wizard. If you use Query Wizard to create queries having selection criteria, Microsoft Query generates an SQL statement having invalid syntax, and the query fails to execute. Microsoft has confirmed that this is a problem with Query, but is not planning to provide a fix in any future release.

Visual Basic

Data Access Objects have the following functional limitations imposed by their use of the Microsoft Jet database engine:

Only the first 255 columns of an ODBC table are accessible. All other columns are hidden from the user.

ODBC Tables with table names longer than 64 characters are inaccessible.

ODBC Tables with column names longer than 64 characters are accessible for queries, but attempts to insert, update, or delete rows from the table will fail.

The above limitations do not exist with Remote Data Objects.

Administering UniVerse ODBC Accounts

Making UniVerse Files Accessible to ODBC Clients	2-3
Removing ODBC Access to UniVerse Files	2-4
UniVerse Server Administration Menu	2-5

This chapter describes the following administrative tasks:

- Making UniVerse files accessible to ODBC clients
- Removing ODBC access to UniVerse files
- Using the UniVerse Server Administration Menu

Making UniVerse Files Accessible to ODBC Clients

UniVerse tables and views are always accessible to ODBC clients. To use UniVerse files that are not tables, however, you must make them accessible—that is, visible—to ODBC clients. Complete the following steps to make such files visible to ODBC clients:

1. Log on as UniVerse Administrator.
2. Change directory to the UniVerse ODBC administration account directory (HS.ADMIN). For example, if UniVerse is installed in */usr/ibm/uv*, enter:

```
# cd /usr/ibm/uv/HS.ADMIN
```
3. Invoke UniVerse and enter **HS . ADMIN** to access the UniVerse Server Administration menu.
4. Enter **3** to choose **Activate Access to Files in an Account** from the UniVerse Server Administration menu. For information about UniVerse Server Administration menu options, see *Administering UniVerse*.
5. Do one of the following:
 - Enter the full path of the UniVerse account directory (on Windows platforms, start with the drive letter, for example **D:**).
 - Enter the UniVerse account name as listed in the UV.ACCOUNT file.
 - Press **Enter** to see a list of UniVerse accounts in which file access has already been activated.
6. Repeat steps 4 and 5 for all accounts whose files you want to make accessible to UniVerse ODBC.
7. When finished, press **Enter** to exit the UniVerse Server Administration menu.

For a detailed description of how to make UniVerse data accessible to ODBC clients, see Chapter 4, “[Making UniVerse Data Accessible to ODBC Applications.](#)”

Removing ODBC Access to UniVerse Files

To remove UniVerse ODBC access to UniVerse files that are not tables:

1. Log on as UniVerse Administrator.
2. Change directory to the UniVerse ODBC administration account directory (HS.ADMIN). For example, if UniVerse is installed in `/usr/ibm/uv`, enter:
`# cd /usr/ibm/uv/HS.ADMIN`
3. Invoke UniVerse and enter **HS . ADMIN** to display the UniVerse Server Administration menu.
4. From the UniVerse Server Administration menu, enter **4** to choose **Deactivate Access to Files in an Account**. UniVerse prompts for the name of the account whose files you want to deactivate.
5. Do one of the following:
 - a. Enter the full path of the UniVerse account (on Windows platforms, start with the drive letter, for example **D:**).
 - b. Enter the account name as listed in the UV.ACCOUNT file.
 - c. Press **Enter** to see a list of UniVerse accounts in which file access has already been activated.

When you enter an account name, the UniVerse ODBC-specific VOC entries and the file information cache are removed, and the `HS_FILE_ACCESS` file is deleted.

***Note:** This does not modify files or their dictionaries in the account, nor does it delete the account.*

6. Repeat steps 4 and 5 for all accounts whose files are accessible to UniVerse ODBC.
7. When finished, press **Enter** to exit the UniVerse Server Administration menu.



UniVerse Server Administration Menu

Use the UniVerse Server Administration menu to perform UniVerse ODBC system administration tasks. Complete the following steps to access the menu:

1. Log on as UniVerse Administrator.
2. Change directory to the UniVerse ODBC administration account directory (HS.ADMIN). For example, enter:

cd /usr/ibm/uv/HS.ADMIN (UNIX)
cd \IBM\UV\HS.ADMIN (Windows NT)
3. Invoke UniVerse and enter **HS .ADMIN**.

To exit the menu, press **Enter**.

Menu Options

The UniVerse Server Administration menu includes the options shown in the following table.

Option	Description
1. List Activated Accounts	
	List UniVerse accounts whose files are accessible to UniVerse ODBC clients.
2. Show UniVerse ODBC Config Configuration for an Account	
	Display configuration settings that clients need to access the account. This option invokes HS.SHOW.CONFIG.
3. Activate Access to Files in an Account	
	Make UniVerse files in an account accessible to ODBC clients. This option runs the UniVerse ODBC file access utility, which does the following: Creates the HS_FILE_ACCESS file Creates the server's file information cache Updates the UV.ACCOUNT file

UniVerse Server Administration Menu Options

Option	Description
4. Deactivate Access to Files in an Account	
	Remove access to UniVerse files in an account. This option deletes the server file information cache and the HS_FILE_ACCESS file, and updates the UV.ACCOUNT file.
5. Run HS.SCRUB on a File/Table	
	Detect, report, and optionally correct data and dictionary entries that can cause ODBC access problems. This option invokes HS.SCRUB.
6. Update File Information Cache in an Account	
	Rebuild the file information cache. This option invokes HS.UPDATE.FILEINFO.
UniVerse Server Administration Menu Options (Continued)	

Installing and Configuring the UniVerse ODBC Driver

Installing UniVerse ODBC Driver Software	3-3
Configuring UniVerse ODBC Driver Software	3-8
Configuring UCI for UniVerse ODBC Driver	3-17
Uni Call Interface and the uci.config File.	3-17
Changing the MAXFETCHCOLS Parameter	3-17
Changing the MAXFETCHBUFF Parameter	3-18
Changing the UCI Connection Timeout	3-19
ODBC Diagnostic Tool: Dr. DeeBee Spy	3-20
Monitoring Your Configuration	3-20
The Dr. DeeBee Spy Log File	3-22

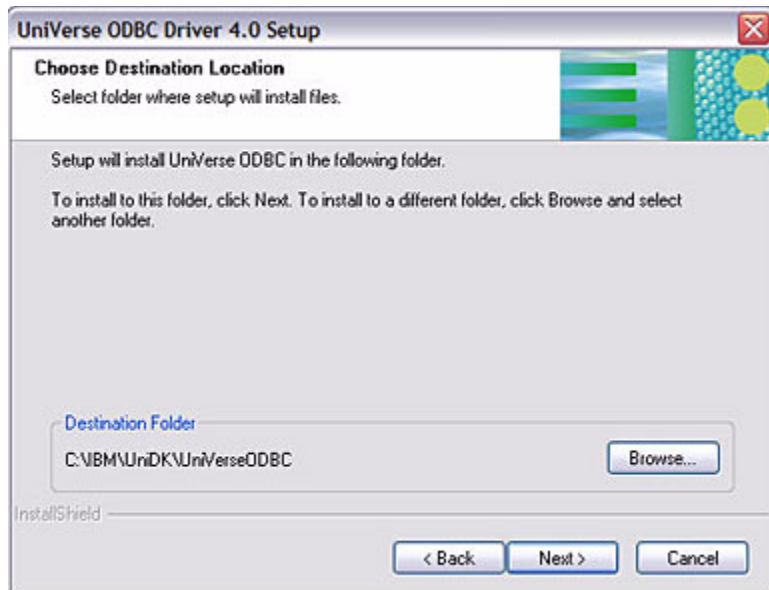
This chapter describes how to install and configure UniVerse ODBC driver software. Install the driver software on a Windows platform.

This chapter also contains information on using a third-party ODBC diagnostic tool, Dr. DeeBee Spy.

Installing UniVerse ODBC Driver Software

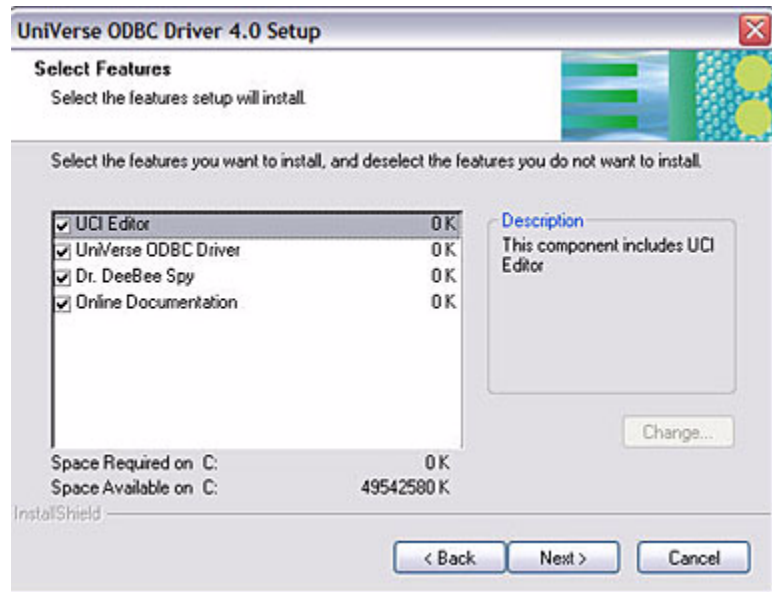
Complete the following steps to install UniVerse ODBC on Windows:

1. From the UniVerse 10.2 Client Installation screen, select UniVerse ODBC. The **UniVerse ODBC Driver 4.0** dialog box appears. Click **Next** to continue with the installation process.
2. After accepting the licensing agreement, click **Next**. The **Choose Destination Location** dialog box appears, as shown in the following example:



By default, the installation process installs UniVerse ODBC in the C:\IBM\UniDK\UniVerseODBC folder. Click **Next** to install UniVerse ODBC in the default folder, or click **Browse** to search for a different folder.

3. The **Select Components** dialog box appears, as shown in the following example:

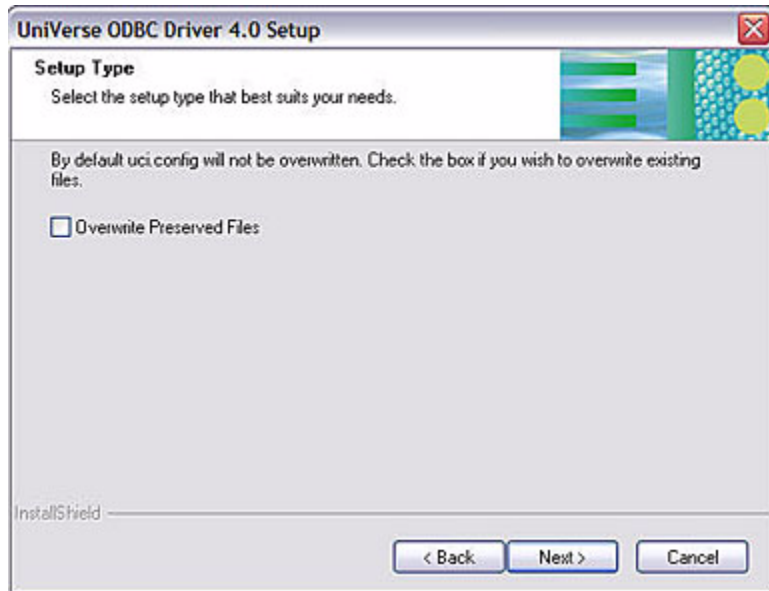


You can choose to install any or all of the following components:

- UniVerse ODBC Driver
- Dr. DeeBee Spy – monitors the activity between an ODBC enabled application and an ODBC driver, and creates a log of the calls.
- Online Documentation
- UCI Editor

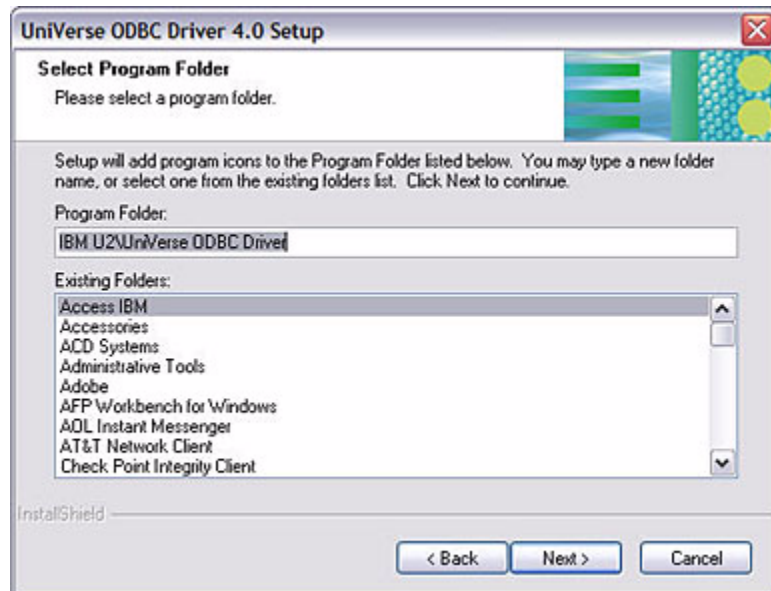
Select the component(s) you want to install, and then click **Next**.

4. The **Setup Type** dialog box appears, as shown in the following example:



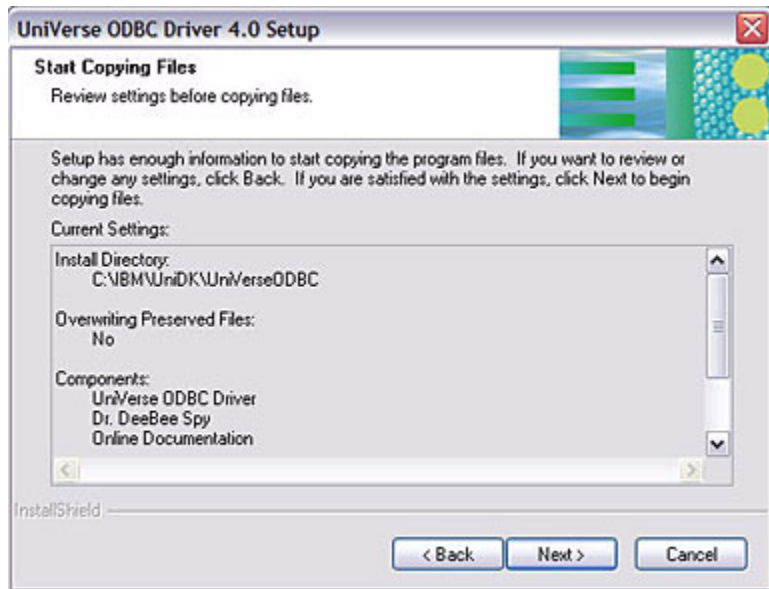
By default, the installation process does not overwrite the uci.config file, if it exists. If you want to overwrite this file, select the **Overwrite Preserved Files** check box. Click **Next**.

5. The **Select Program Folder** dialog box appears, as shown in the following example:



By default, the installation process adds the IBM U2\UniVerse ODBC Driver program icon to the Program Folders list. Click **Next** to accept this default, or select a different folder from the **Existing Folders** list and then click **Next**.

6. The **Start Copying Files** dialog box appears, as shown in the following example:



Review the information listed in the **Start Copying Files** dialog box. If the information is correct, click **Next** to begin copying files. If the information is not correct, click **Back** to make changes.

7. Click **Finish** to complete the installation process.

Converting DSNs

If the DSN conversion process fails, or if you need to rerun the process, execute the following command:

```
ConvertDsn.exe c:\ibm\uvodbc\config.hsc  
c:\ibm\uv\uvdk\config\uci.config
```

If you want to reverse the conversion process, execute the following command:

```
ConvertDsn.exe -u c:\ibm\uvodbc\config.hsc c:\ibm\uv\uvdk\config\uci.config
```

Configuring UniVerse ODBC Driver Software

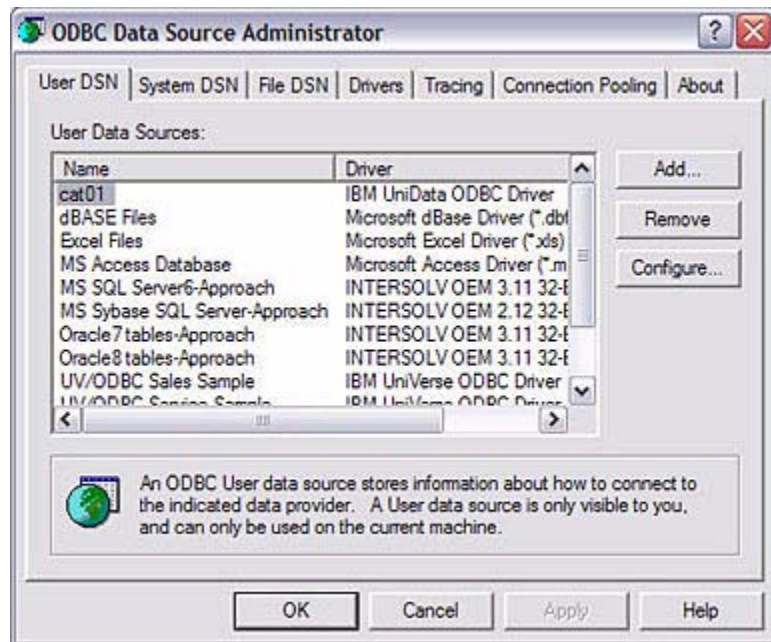
You must run the ODBC Data Source Administrator tool to define a configuration. A UniVerse ODBC configuration defines the following:

- Server identification (system name and system type)
- Server authorization (user name and password)
- UniVerse account to access using UniVerse ODBC

UniVerse no longer stores configuration information in the config.hsc file. Instead, configuration information is stored in the registry and in the uci.config file.

Complete the following steps to configure a UniVerse data source:

1. To access the Microsoft ODBC Data Source Administrator tool, choose **Start -> Programs -> IBM U2 UniVerse ODBC Driver -> ODBC Admins**. The **ODBC Data Source Administrator** dialog box appears, as shown in the following example:



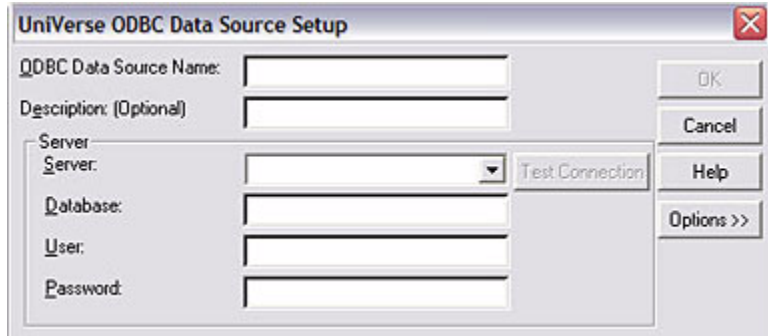
Click **Add** to add a new data source.

2. The **Create New Data Source** dialog box appears, as shown in the following example:



Scroll the **Name** list and select **IBM UniVerse ODBC Driver**. Click **Finish**.

3. The **UniVerse ODBC Data Source Setup** dialog box appears, as shown in the following example:



- In the **ODBC Data Source Name** box, enter the data source name. ODBC applications use this name to connect to the data source.
- In the **Description** box, you can enter a description of the data source if desired.
- In the **Server** box, enter the name of the server to which you want to connect.
- In the **Database** box, enter the full path to the database to which you want to connect: for example, **c:\ibm\uv\HS.SALES**.
- In the **User** box, enter the user name.
- In the **Password** box, enter the password for the User.

Click **Test Connection** to ensure UniVerse ODBC accepts the connection to the server and database you defined.

4. Click **Options** to define optional configuration information. A dialog box similar to the following example appears:

Strict Statement Compliance Options

You can select the following Strict Statement Compliance Options:

- **Enforce ODBC Dates/Times** – If you select the **Enforce ODBC Dates/Times** check box, UniVerse ODBC checks all date and time literals for strict adherence to ODBC specifications. You must specify years with four digits, and months, days, hours, minutes, and seconds with two digits.

If you do not select the **Enforce ODBC Dates/Times** check box, UniVerse ODBC accepts two-digit years, and single-digit months, days, hours, minutes, and seconds, and passes these shorter values to UniVerse.

- **Enforce ODBC Function Argument Types** – If you select the **Enforce ODBC Function Argument Types** check box, UniVerse ODBC type-checks arguments to scalar functions, where possible.

If you do not select the **Enforce ODBC Function Argument Types** check box, UniVerse ODBC passes all arguments to scalar functions to UniVerse without performing any type checking.

- **Reject non-ODBC SQL Syntax** – If you select the **Reject non-ODBC SQL Syntax** check box, UniVerse ODBC reports an error if it encounters an SQL construct that is not in the ODBC SQL grammar. Select this option if you are writing an application that needs to be portable to other ODBC drivers or databases, or both.

If you do not select the **Reject non-ODBC SQL Syntax** check box, UniVerse ODBC accepts certain non-ODBC SQL constructs.

- **Reject Constructs Unsupported by UniVerse** – If you select the **Reject Constructs Unsupported by UniVerse** check box, UniVerse ODBC returns an error if it encounters an SQL construct that is part of the ODBC SQL grammar but which UniVerse does not support.

If you do not select the **Reject Constructs Unsupported by UniVerse** check box, UniVerse ODBC passes unsupported constructs to UniVerse, which will then issue its own error message.

UniVerse ODBC Parameters

UniVerse ODBC parameters can no longer be specified on the server.

- **UniVerse Name Mapping** – Name mapping controls whether UniVerse ODBC maps the names of tables, columns, and indexes to ODBC-compliant equivalents. If you select **Yes**, name mapping is enabled. If you select **No**, name mapping is disabled.

***Note:** The `HS_NAME_MAPPING` environment variable has no effect at UniVerse 10.2.*

- **Retry as UniVerse SQL** – When you send an SQL statement to UniVerse ODBC, it tries to interpret the statement as if it were written in the ODBC dialect of SQL. If you do not select **Retry as UniVerse SQL**, UniVerse ODBC considers only mapped names to be valid in this dialect. If the interpretation fails, UniVerse ODBC acts according to the **Retry as UniVerse SQL** option.

If you select **Yes**, UniVerse ODBC assumes that the statement was written in the UniVerse dialect of SQL, and passes it through to UniVerse without modification. It does not apply name mapping nor file access control to the statement.

***Note:** The `HS_RETRY_AS_UV_SQL` environment variable has no effect at UniVerse 10.2.*



Performance Options

You can select any of the following performance options:

- **Prefetch Size** – This field specifies the number of rows of a result set that are fetched (or prefetched) from the database when one or more unfetched result set rows are requested. Each block of result set rows fetched from the database when result set rows are needed is called a result set chunk. Each result set chunk contains Size rows of the result set. Valid values for this field range from 1 to 32,767.
- **Threshold** – This field specifies the number of result set rows that must be consumed (via calls to SQLFetch) from the current result set chunk before the next result set chunk is prefetched from the server. The Threshold must be less than or equal to the Size. If a threshold that is greater than the size is entered, UniVerse ODBC internally adjusts the threshold to be equal to the size when it reads in the values specified. Valid values for this field range from 0 to 32,767.
- **Max LONG Type Length** – This field specifies the maximum number of bytes of data per data item that are fetched from the database for any SQL_LONGVARCHAR or SQL_LONGVARBINARY type columns. This option is ignored as UniVerse ODBC currently does not support the SQL_LONGVARCHAR and SQL_LONGVARBINARY data types.
- **Fast Connect (Old OTL)** – This option applies only to UniVerse files that are not tables. When you select this option, the ODBC table list is not refreshed in memory, and the File Information Cache is read from disk and used instead. This option may result in faster connect times, but possibly in a less up-to-date list of available tables.

Note: The HS_USE_FILEINFO environment variable has no effect at UniVerse 10.2.

- **Refresh OTL on Connect** – This option applies only to UniVerse files that are not tables. When you select this option, the ODBC table list is queried. This operation may impact initial connect time. The File Information Cache is not updated by the operation. Update the cache by running HS.UPDATE.FILEINFO in the UniVerse account you are accessing on the server system.

NLS Options

You can define any of the following NLS options:

- **NLS Mapname** – The name of the map that defines how a character set is mapped between the internal and external character sets.
- **Locale Name** – The name of a specific set of conventions in various categories that represent the language, character set, and data formatting conventions used by a group of people.

Miscellaneous Options

- **Create Debug Log** – Select the **Create Debug Log** check box if you want to create a debug log.
- **Config File Name** – Displays the full path of the uci.config file, defined in the Registry in read-only mode.

Configuring UCI for UniVerse ODBC Driver

After you install and configure UniVerse ODBC driver software, the next task is to configure the Uni Call Interface (UCI) on the UniVerse ODBC client.

Uni Call Interface and the uci.config File

The UniVerse ODBC driver uses UCI to access UniVerse data. The UCI interface no longer uses the uvodbc.config file to obtain information about data access, or the config.hsc file. It now uses the uci.config file to communicate to the UniVerse database server.

The following is an example of a uci.config file:

```
[ODBC DATA SOURCES]
<localuv>
DBMSTYPE = UNIVERSE
network = TCP/IP
service = uvserver
host = localhost

<localud>
DBMSTYPE = UNIDATA
network = TCP/IP
service = udserver
host = localhost
```

To access your data from UniVerse ODBC, you may need to increase the values of two configuration parameters for the UNIVERSE DBMS type. These parameters are defined in the uci.config file. You may also need to increase the UCI connection timeout.

For details about configuring UCI, see *UCI Developer's Guide*.

Changing the MAXFETCHCOLS Parameter

The default MAXFETCHCOLS setting is 400 column values. Increase the value of the MAXFETCHCOLS configuration parameter if either of the following conditions applies:

- Any of your UniVerse files, tables, or views have more than 400 fields defined in the dictionary (field definitions include D-, A-, I-, and S-descriptors)
- Any queries executed from ODBC client applications through UniVerse ODBC contain more than 400 columns in the result set

Changing the MAXFETCHBUFF Parameter

The default MAXFETCHBUFF setting is 8192 bytes. Increase the value of the MAXFETCHBUFF configuration parameter if either of the following conditions applies:

- The record length of any table or file exceeds 8192 bytes
- Any queries executed through UniVerse ODBC yield a result set with rows longer than 8192 bytes

Example

The following UCI configuration file definition sets MAXFETCHCOLS to 1000 and MAXFETCHBUFF to 32000 for all UNIVERSE data sources:

```
<localuv>
DBMSTYPE = UNIVERSE
network = TCP/IP
service = uvserver
host = localhost

[UNIVERSE]
MAXFETCHCOLS = 1000
MAXFETCHBUFF = 32000
```

If any query you try to execute has more than MAXFETCHCOLS result columns or has a result row length greater than MAXFETCHBUFF bytes, the query fails with a UCI error code of 930122:

```
[IBM][SQL Client][UNIVERSE]UniVerse/SQL: Row length
exceeds buffer size
```

Changing the UCI Connection Timeout

The six-minute default inactivity timeout value for UCI connections can be too short. If users leave client ODBC connections open but inactive for longer than six minutes, they can receive UniVerse error code 81015. To increase this timeout value, log on as a UniVerse Administrator and edit the *unirpcservices* file in the *unishared/unirpc* directory. In the line starting with *uvserver*, change the rightmost number to 864,000 (10 times the number of seconds in a 24-hour day). The line should appear as follows:

```
uvserver <uvhome>/bin/uvsrvd * TCP/IP 864000
```

<uvhome> is the path of the UV account directory (for example, */usr/ibm/uv* on UNIX systems or *\IBM\UV* on Windows platforms).

ODBC Diagnostic Tool: Dr. DeeBee Spy

UniVerse ODBC is delivered with a third-party ODBC diagnostic tool called Dr. DeeBee Spy. Dr. DeeBee Spy monitors activity between ODBC applications and drivers and keeps a log of the activity. Use this log to track down problems with ODBC applications or drivers.

To use Dr. DeeBee Spy, double-click the **Dr. DeeBee Spy** icon, pick the data source to monitor, then run an ODBC-enabled application to monitor. For more information on using Dr. DeeBee Spy, see [Monitoring Your Configuration](#).

Install Dr. DeeBee Spy in the Custom installation section of the UniVerse ODBC driver installation process.

Monitoring Your Configuration

Dr. DeeBee Spy monitors activity between ODBC applications and drivers and keeps a log of the activity. Use this log to track down problems with ODBC applications or drivers.

Two common issues that Dr. DeeBee Spy can address are as follows:

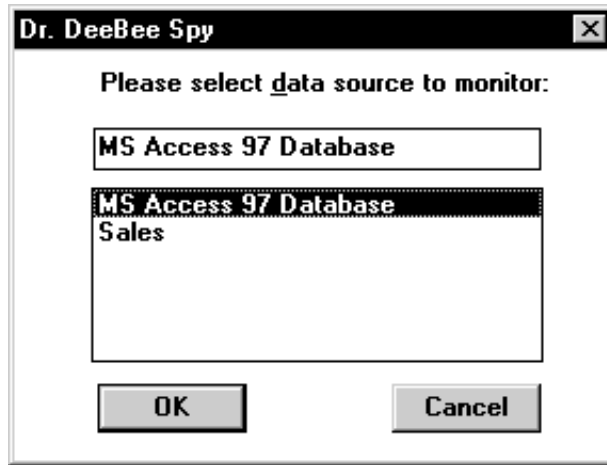
- Many front-end applications report ODBC problems very briefly (for example, “ODBC call failed”). Near the bottom of the DRDEEBEE.LOG file you should see a logged call to **SQLError** that shows the complete error message that the UniVerse database server reported to the front-end application.
- Some queries appear to run inefficiently and you may be unsure how changes made to the front-end application affect ODBC SQL. Search the DRDEEBEE.LOG file for calls to **SQLPrepare** and **SQLExecute** to see the exact SQL statements that your front end is sending.

Dr. DeeBee Spy is compatible with both ODBC 1.0 and 2.0.

To use Dr. DeeBee Spy:

1. Access **Dr. DeeBee Spy** by choosing **Start ->Programs -> IBM U2 -> UniVerse ODBC Driver -> Dr. DeeBee Spy**.
2. A log file called DRDEEBEE.LOG may exist in the WINDOWS directory. Choose to overwrite it or append it.

3. Dr. DeeBee Spy asks for the name of the data source to monitor. Select a data source, then click **OK**.



Dr. DeeBee Spy is initialized and then is minimized to an icon. Dr. DeeBee Spy records all ODBC calls until it is closed.

4. Start the ODBC-enabled application.
5. Use the ODBC-enabled application to access information through your selected data source.
6. This can be done with an automated test script or manually. Dr. DeeBee Spy displays and records all requests to the ODBC driver.
7. Close Dr. DeeBee Spy.
8. Open the DRDEEBEE.LOG file from the WINDOWS directory to examine the ODBC calls.

The Dr. DeeBee Spy Log File

The general structure of information captured to the DRDEEBEE.LOG file is as follows:

```
    ODBC Function
      argument value
      .
      .
      .
    RETCODE
```

The left-justified text is the ODBC function called by the application. The total number of indented lines in the captured information matches the total number of arguments for that function.

For example:

```
    SQLAllocEnv
      0x01000000
      SQL_SUCCESS
    SQLAllocConnect
      0x01000000
      0x01010000
      SQL_SUCCESS
```

In the first function (`SQLAllocEnv`), the driver generated the environment handle (*phenv*, 0x01000000) and the return value (`SQL_SUCCESS`). In the second function (`SQLAllocConnect`), the application generated the environment handle (*henv*, 0x01000000), and the driver generated the connection handle (*phenv*, 0x01010000) and the return value (`SQL_SUCCESS`).

Dr. DeeBee Spy resides between the Driver Manager and the driver. Therefore, it captures calls to ODBC functions made by the application and calls made solely by the Driver Manager. After the Driver Manager loads the driver, for example, it calls **SQLGetInfo** to determine which version of ODBC the driver supports.

Other methods of monitoring configurations and diagnosing problems include the **SQLTrace** feature of the ODBC Data Source Administrator and tracing features built in to a front-end application.

Making UniVerse Data Accessible to ODBC Applications

Overview	4-3
Making UniVerse Accounts Accessible	4-3
Making Data in a UniVerse Account Accessible	4-3
Presenting UniVerse Data in ODBC Format	4-4
Accessing UniVerse Schemas and Accounts	4-6
Using ODBC Qualifiers	4-6
Accessing UniVerse Tables, Views, and Files	4-9
Updating a UniVerse Account for ODBC Access	4-9
File Privileges and Permissions	4-10
What the ODBC File Access Utility Does	4-11
Accessing Saved Select Lists	4-14
Accessing Columns and Fields	4-16
ODBC-Accessible Columns and Fields	4-16
Multivalued Columns and Fields	4-16
Modifying UniVerse Data and Data Definitions for ODBC	4-20
Validating Tables and Files for ODBC Clients	4-20
SQL Data Types	4-22
Length of Character Data	4-23
Empty or Unconvertible Data	4-25
Conversion Codes	4-26
Association Keys.	4-27
Defining Association Keys	4-27
ODBC Name Mapping	4-29

This chapter describes how to make data in UniVerse tables and files accessible to ODBC client programs. You need to do three things:

1. Make UniVerse accounts accessible to ODBC applications.
2. Specify what data in those accounts ODBC applications can access.
3. Present the data in ODBC format.

Overview

This section gives a general overview of what you need to do to make UniVerse data accessible to ODBC applications. Subsequent sections describe each major step in detail.

Making UniVerse Accounts Accessible

ODBC applications connect to ODBC data sources. A UniVerse ODBC data source is defined as one UniVerse schema or account, although there are ways to make more than one schema or account accessible through a single UniVerse ODBC connection. For details about how to make UniVerse schemas and accounts accessible, see [Accessing UniVerse Schemas and Accounts](#).

Making Data in a UniVerse Account Accessible

The following UniVerse data can be made accessible to ODBC applications:

- Tables, views, and UniVerse files
- Columns and fields in tables, views, and UniVerse files
- Saved select lists associated with tables, views, and UniVerse files

Tables, Views, and UniVerse Files

UniVerse tables and views are always accessible to ODBC applications, but UniVerse files that are not tables are not. To make UniVerse files accessible to ODBC applications, you must run the ODBC file access utility in the account. Among other things, this utility creates the HS_FILE_ACCESS file, which lists all UniVerse files referenced by F- and Q-pointers in the VOC file. You can edit this file to define exactly which files should be ODBC-accessible. You probably also need to run the HS.UPDATE.FILEINFO program from time to time, which updates the account's file information cache.

You can also make saved select lists that are associated with a table, view, or file visible to ODBC applications.

For detailed information about making tables, views, files, and select lists accessible, see [Accessing UniVerse Tables, Views, and Files](#).

Columns and Fields

You can make particular columns and fields accessible to ODBC applications. You can do this by adding or editing an @SELECT phrase in the dictionary of a table, view, or file. If a UniVerse file contains multivalued data, you may also need to edit other entries in the dictionary that control the behavior of multivalued columns and fields. For detailed information about making columns and fields accessible, see [Accessing Columns and Fields](#).

Presenting UniVerse Data in ODBC Format

UniVerse data is organized differently from the way ODBC applications expect it to be organized. Two areas where UniVerse data differs from standard ODBC data are:

- Data and data types
- Multivalued data

Data and Data Types

Data in UniVerse files has no data type. ODBC applications, on the other hand, expect all data to be one of several data types. In addition, UniVerse data can be of variable length, whereas ODBC expects data to have either a fixed or a maximum length. To make UniVerse data look more like what an ODBC application expects, you may want to do one or more of the following:

- Run the HS.SCRUB utility in the account
- Define SQL data types for data in UniVerse files
- Fix data values that cause SQL and ODBC problems
- Modify certain UniVerse conversion codes

Details about these can be found in [Modifying UniVerse Data and Data Definitions for ODBC](#).

Multivalued Data

ODBC applications expect data to be organized relationally in first normal form (1NF). Although some UniVerse tables, views, and files may be in first normal form, with only one value in each column of each row, many UniVerse tables, views, and files have columns that store multiple values in the columns of a row.

UniVerse ODBC always presents multivalued data to ODBC applications in first normal form. UniVerse ODBC automatically normalizes UniVerse tables and files in order to present their data to ODBC applications in ODBC format. For detailed information about how UniVerse normalizes UniVerse data files, see [Multivalued Columns and Fields](#) and [Association Keys](#).

Accessing UniVerse Schemas and Accounts

UniVerse databases can be organized into:

- UniVerse SQL schemas
- UniVerse accounts that are not schemas

A UniVerse ODBC application connects to a UniVerse schema or account and can access the tables, views, and files there. The schema or account must be defined as a data source on the client machine using:

- The Windows ODBC Administrator
- The UCI Config Editor tool

See Chapter 3, “[Installing and Configuring the UniVerse ODBC Driver](#),” for how to configure UniVerse ODBC data sources.

To access other schemas and accounts, you can do three things:

- Use Q-pointers or remote F-pointers. A remote file with a Q-pointer or F-pointer in the VOC of the local account appears as a local file. In UniVerse ODBC, Q-pointers and remote F-pointers work only with files, not with tables or views. For information about how to use Q-pointers, see *UniVerse System Description*.
- Use ODBC qualifiers. ODBC qualifiers do not require you to set up Q-pointers, but they work only on SQL tables and views.
- Use multiple UniVerse ODBC connections, each to a different account.

Note: *You cannot access files through UV/Net using UniVerse ODBC.*



Using ODBC Qualifiers

A qualifier is the ODBC equivalent of the name of a UniVerse schema or account. In ODBC SQL you can refer to a remote table as *qualifier.tablename*. An ODBC connection has a *current qualifier* which implicitly qualifies unqualified table names.

For ODBC tables derived from UniVerse tables and views, UniVerse ODBC reports the UniVerse schema name as the ODBC qualifier.

For ODBC tables derived from UniVerse files, UniVerse ODBC reports the empty string as the qualifier. To refer explicitly to a UniVerse file in an SQL statement, qualify the ODBC table name with the empty string qualifier. For example:

```
SELECT * FROM "" .MYFILE;
```

UniVerse ODBC uses two qualifiers:

- The *current* ODBC qualifier
- The *local* ODBC qualifier

Both the current and the local qualifiers are initially set to:

- The name of the schema to which you are connected
- An empty string if the UniVerse account is not a schema

You can change the current qualifier to the name of another schema (or to an empty string) using the `SQL_CURRENT_QUALIFIER` option of the **SQLSetConnectOption** function. The local qualifier is the name of the UniVerse account to which the client is connected and remains constant during the ODBC connection.

Resolving Unqualified ODBC Table Names

UniVerse ODBC uses the following rules when resolving an unqualified ODBC table name in an SQL statement:

- If the table name occurs in a place in an SQL statement where UniVerse SQL does not allow a qualified table name (such as `DROP TABLE`), UniVerse ODBC resolves the table name using the local qualifier. An error occurs if the SQL statement is a DDL statement and the local account is not a schema, because UniVerse SQL does not allow DDL statements to be executed in accounts that are not schemas.
- If the table name occurs in a place in an SQL statement where UniVerse SQL allows a qualified table name, UniVerse ODBC first looks for a file defined in the local VOC file that matches the ODBC table name.
- If no match is found, and if the current ODBC qualifier is the name of a UniVerse schema (that is, it is not set to the empty string), UniVerse ODBC looks for an SQL table or view in that schema that matches the ODBC table name.

Accessing UniVerse Tables, Views, and Files

UniVerse schemas can contain:

- SQL tables and views
- Files that are not tables or views

UniVerse accounts that are not schemas can contain only UniVerse files that are not tables or views.

Tables and views are always accessible to ODBC applications, subject to SQL privileges and operating system permissions. To make files that are not tables accessible to ODBC applications, you must run the ODBC file access utility. To do this, choose the **Activate Access to Files in an Account** option from the UniVerse ODBC System Administration menu.

When you run the ODBC file access utility, it creates the HS_FILE_ACCESS file, which lists all UniVerse files referenced by F- and Q-pointers in the VOC file. This makes all UniVerse files in the account, except system files such as &DEVICE&, DICT.DICT, APP.PROGS, and so forth, accessible to ODBC applications. You can edit the HS_FILE_ACCESS file to define exactly which files should be ODBC-accessible (see [File Privileges and Permissions](#)).

For more information about how UniVerse ODBC modifies an account to make it ODBC-accessible, see [What the ODBC File Access Utility Does](#).

By default, UniVerse ODBC applies name mapping to the names of tables, views, and files to make them ODBC-compliant. For more information about name mapping, see [ODBC Name Mapping](#).

Updating a UniVerse Account for ODBC Access

If you make changes to UniVerse files in the account after running the ODBC file access utility and you want the new information to be accessible to UniVerse ODBC, you should update the UniVerse ODBC file information cache. The following changes to UniVerse files require you to update UniVerse ODBC file access in the account:

- Adding, changing, or deleting F- or Q-pointers in the VOC file
- Creating or deleting UniVerse files

- Defining, changing, or deleting association definitions in file dictionaries
- Adding or deleting unassociated multivalued fields to or from a file

To update the UniVerse ODBC file information cache for an account, choose the **Update File Information Cache in an Account** option from the UniVerse Server Administration menu. You can also use the HS.UPDATE.FILEINFO command to update UniVerse ODBC file access in an account.

File Privileges and Permissions

UniVerse SQL privileges control access to the tables and views in a schema or an account. However, they do not provide any kind of access control over UniVerse files.

Operating system permissions on the files underlying UniVerse tables, views, and files also control access to those tables, views, and files.

UniVerse ODBC uses the HS_FILE_ACCESS file, created by the ODBC file access utility, to control access to UniVerse files. This file controls UniVerse ODBC access to UniVerse files defined by F- or Q-pointers in the VOC file.

The IDs of records in the HS_FILE_ACCESS file are the names of the files whose access you want to control. Each record has one field, ACCESS, which contains one of the following values:

- READ_WRITE
- READ
- NONE

These values define the type of access to the file referenced by the record ID.

The HS_FILE_ACCESS file contains a record called HS_DEFAULT that controls default access to all files in the account. When you first run the ODBC file access utility, it sets the HS_DEFAULT record to READ_WRITE and sets the records for UniVerse system files (APP.PROGS, BASIC.HELP, ERRMSG, UV.ACCOUNT, DICT.DICT, NEWACC, and so forth) to NONE (no access). To restrict UniVerse ODBC to read-only access for all files in the account, change the ACCESS field in the HS_DEFAULT record to READ.

If an account does not have an HS_FILE_ACCESS file, UniVerse ODBC denies all access to UniVerse files in the account.



Note: You can circumvent the access control provided by the `HS_FILE_ACCESS` file by using the native `SQL` syntax extension to the `ODBC SQL` grammar or by reparsing. These mechanisms allow `SQL` statements and UniVerse commands to be passed directly to UniVerse, bypassing UniVerse ODBC. For more information see *Executing UniVerse SQL*.

What the ODBC File Access Utility Does

When you run the ODBC file access utility in an account, it does the following:

- Creates the `HS_FILE_ACCESS` file
- Writes an `@EMPTY.NULL` X-descriptor in all UniVerse file dictionaries
- Writes S or M in field 5 of A- and S-descriptors
- Creates Q-pointers for multiple data files
- Creates the file information cache
- Updates the `UV.ACCOUNT` file

File Information Cache

When you run the ODBC file access utility, UniVerse ODBC scans the dictionaries of all nonsystem files in the account and stores the information in a file information cache. The UniVerse ODBC driver uses this cache to provide ODBC applications with a list of accessible ODBC tables without having to construct this information at run time, thus improving performance.

Warning: *The file information cache is for UniVerse ODBC internal use only and should not be modified. Any changes to the cache cause unpredictable behavior and can make all UniVerse files in the account inaccessible to UniVerse ODBC.*

To control how the server uses the file information cache, use the **Fast Connect (Old OTL)** or the **Refresh OTL on Connect** option on the UniVerse ODBC Data Source Setup menu of the UniVerse ODBC Administrator.

- Choose **Fast Connect (Old OTL)** (the default setting) if you want the server always to use the file information cache to construct the list of available ODBC tables. If the server cannot access the cache, it constructs the ODBC table information from scratch at connection time.



- Choose **Refresh OTL on Connect** if you want the server to construct the ODBC table list from scratch at connection time, ignoring the cache. The advantage of this mode is that the list of accessible ODBC tables is more current. The disadvantage is that initial connection time can be significantly increased for an account containing many files. Responses to subsequent requests for dictionary information are faster.

Upgrading Accounts from Previous UniVerse ODBC Releases

When you use the **Activate Access to Files in an Account** option, it upgrades the account from previous (non-FP) UniVerse ODBC releases. All configuration information is preserved so that both the previous and upgraded UniVerse database servers are operational.

The following tables illustrate how accounts and files from previous UniVerse ODBC releases are updated. The next table lists the VOC entries affected by the upgrade procedure.

VOC Entry	Description	Upgrade Processes
HYPERFILES	Files accessible to UniVerse ODBC.	For each file listed, an HS_FILE_ACCESS entry is created with access set to READ_WRITE. The default file access is set to NONE.
HYPERLISTFILT	Matches patterns to filter which select lists are visible to UniVerse ODBC.	Ignored.
HYPER.ODBC.TEXT.MULT	Multiplication factor for calculating the ODBC display width on text fields.	Ignored.
Nonserver VOC Entries		

The following table lists the dictionary entries affected by the upgrade procedure.

DICT Entry	Description	Upgrade Processes
@HYPERFIELDS	Fields accessible to UniVerse ODBC.	Copied to @SELECT entry.
@HYPER.SEARCH.FIELDS	Fields searchable by UniVerse ODBC.	Ignored. All UniVerse ODBC-accessible fields are searchable.
@HYPER.UPDATE.FIELDS	Fields updatable by UniVerse ODBC.	Ignored. All UniVerse ODBC-accessible noncomputed (stored) fields are updatable.
HYPER.ODBC.ALIAS	Map of file, association, and field names to aliases.	Ignored. Name mapping is done automatically by UniVerse ODBC without provision for user-specified aliases.
HYPER.ODBC.KEYS. <i>name</i>	Key fields in an association or unassociated multivalued field.	Copied to the @ASSOC_KEY. <i>mvname</i> entry.
HYPER.ODBC.LISTS	Select list names associated with this file.	Copied to HS.ODBC.LISTS.
HYPER.ODBC.TEXT.MULT	Multiplication factor for calculating ODBC display width on text fields.	Ignored.

NonsERVER DICT Entries

Accessing Saved Select Lists

UniVerse ODBC makes saved select lists accessible to ODBC applications. An ODBC application sees a select list as a virtual table with a name in the following format:

filename_listname

To make a saved select list accessible to UniVerse ODBC, the following conditions must be met:

- The select list must be saved in the &SELECTLISTS& file of the local account.
- The name of the select list must be included in an X-descriptor called HS.ODBC.LISTS in the dictionary of a table, view, or file.
 - A table or view associated with the select list must reside in the local schema.
 - A file associated with the select list can be local or remote—that is, it can be referenced by an F- or a Q-pointer in the VOC file.

The HS.ODBC.LISTS X-descriptor has the following format:

```

HS.ODBC.LISTS
0001 X
0002 LIST1 LIST2 LIST3 ...

```

Choose your lists carefully. If one table, view, or file has n associations, unassociated multivalued fields, or both, and if it also has m select lists, ODBC applications see $(n + 1) * (m + 1)$ virtual tables for that one table, view, or file. For example, if the file MYTABLE has the associations MYASSOC1 and MYASSOC2 and the unassociated multivalued field MV1, and if the HS.ODBC.LISTS entry has the lists MYLIST1 and MYLIST2 in field 2, the following ODBC tables would appear in the result set of **SQLTables**:

```

MYTABLE
MYTABLE_MYLIST1
MYTABLE_MYLIST2
MYTABLE_MYASSOC1
MYTABLE_MYASSOC1_MYLIST1
MYTABLE_MYASSOC1_MYLIST2
MYTABLE_MYASSOC2
MYTABLE_MYASSOC2_MYLIST1
MYTABLE_MYASSOC2_MYLIST2
MYTABLE_MV1
MYTABLE_MV1_MYLIST1
MYTABLE_MV1_MYLIST2

```

By default, UniVerse ODBC applies name mapping to the names of select lists to make them ODBC-compliant. For information about name mapping, see [ODBC Name Mapping](#).

Accessing Columns and Fields

This section describes what columns and fields are accessible to ODBC, and how to make multivalued columns and fields accessible.

ODBC-Accessible Columns and Fields

For any table, view, or file, the columns and fields accessible to ODBC are those returned by the following SELECT statement, executed in NF² mode:

```
SELECT * FROM tablename
```

For tables and views, accessible columns are those listed in the @SELECT phrase, if it exists. If there is no @SELECT phrase, the columns are those listed in the table's SICA, as defined by CREATE TABLE or CREATE VIEW, and as modified by ALTER TABLE.

For files that are not tables, accessible fields are those listed in the @SELECT phrase, if it exists. If there is no @SELECT phrase, the fields are those listed in the @ phrase. If neither the @SELECT nor the @ phrase exist, only the record ID is accessible.

Note: An @ phrase in the dictionary of a table or view is ignored.

For more information about the @SELECT and @ phrases, see *UniVerse SQL Administration for DBAs*.

By default UniVerse ODBC applies name mapping to the names of columns and fields to make them ODBC-compliant. For information about name mapping, see [ODBC Name Mapping](#).

Multivalued Columns and Fields

UniVerse tables and files can contain any mix of the following:

- Singlevalued columns or fields
- Multivalued columns or fields not associated with any other columns or fields
- Multivalued columns or fields associated with other multivalued columns or fields



UniVerse ODBC shows UniVerse tables, views, and files in *first normal* form. That is, a table, view, or file containing one or more multivalued columns or fields is treated as a set of ODBC tables comprising:

- One table that includes all singlevalued columns or fields
- One table for each association of multivalued columns or fields
- One table for each unassociated multivalued column or field

To make multiple values in a column or field accessible to ODBC applications, the column or field must be defined as multivalued in one of the following ways:

- By the CREATE TABLE, ALTER TABLE, or CREATE VIEW statement that defined the column
- By an M in the SM field of the dictionary definition (field 6 in D- and I-descriptors, field 5 in A- and S-descriptors)
- By a C or D in field 4 of the dictionary definition in A- and S-descriptors (Pick-style associations)

***Note:** If UniVerse ODBC finds multivalued data in a field defined as singlevalued, only the first value in the field is accessible to ODBC.*

If an A- or S-descriptor does not have an M or an S in field 5, the UniVerse ODBC file access utility samples the data in the field, and if it finds multivalued data and can write to the file dictionary, it defines the field as multivalued.

Certain ODBC tables are not visible to the **SQLTables** function, even though they are accessible through SQL statements. Some ODBC applications provide limited or no access to ODBC tables that are not visible to **SQLTables**. **SQLTables** can see the following:

- In UniVerse tables and views, all associations and unassociated multivalued columns defined by the CREATE TABLE, ALTER TABLE, and CREATE VIEW statements
- In UniVerse files, all associations and unassociated multivalued columns
- In UniVerse files, all Pick associations whose controlling field is visible to **SQLTables**

Example

Consider a UniVerse file MYFILE comprising the following fields:

Field Name	Location	Single or Multivalued	Associated
@ID	0	S	No
CUSTOMER	1	S	No
ADDRESS	2	M	No
QTY	3	M	Yes
DESCRIPTION	4	M	Yes

The dictionary contains a phrase defining the association of fields 3 and 4:

```
LINEITEMS
0001 PH
0002 QTY DESCRIPTION
```

The field descriptors for QTY and DESCRIPTION specify LINEITEMS in field 7.

The dictionary also contains the following @ phrase:

```
@
0001 PH
0002 CUSTOMER ADDRESS QTY DESCRIPTION
```

All fields of this file are accessible to ODBC applications. The @ID field is not included in the @ phrase because it is always accessible unless suppressed by the ID.SUP keyword.

UniVerse ODBC presents this file as three tables.

ODBC Table	ODBC Columns
MYFILE	@ID CUSTOMER
MYFILE_ADDRESS	@ID CUSTOMER ADDRESS @ASSOC_ROW ¹
MYFILE_LINEITEMS	@ID QTY DESCRIPTION @ASSOC_ROW ¹

1. System-generated. @ASSOC_ROW appears automatically for any association or unassociated multivalued column or field, unless the dictionary contains an @ASSOC_KEY.*myname* X-descriptor that defines an association key. See [Association Keys](#).

If name mapping is on, @ID appears as Z_ID and @ASSOC_ROW appears as Z_ASSOC_ROW. For information about name mapping, see [ODBC Name Mapping](#).

Modifying UniVerse Data and Data Definitions for ODBC

This section describes things you may need to do to make particular kinds of data accessible to ODBC applications:

- Fix data in data files and dictionaries that cause SQL and ODBC problems
- Define SQL data types for data in UniVerse files
- Define the length of character string data in UniVerse files
- Modify certain UniVerse conversion codes

Validating Tables and Files for ODBC Clients

Use the HS.SCRUB utility to scan data in a table or UniVerse file and fix data file and dictionary problems that cause SQL and ODBC difficulties. The HS.SCRUB utility does the following:

- Reports table and UniVerse file anomalies
- Saves a select list of record IDs of problem records
- Adjusts dictionary entries to accommodate bad data¹
- Adds an @EMPTY.NULL record to the dictionary
- Adds an @SELECT record to the dictionary
- Fixes the data in the file, optionally saving a copy of the original file

@EMPTY.NULL

HS.SCRUB adds an @EMPTY.NULL record to the table or file dictionary if all the following conditions are met:

- The dictionary does not already include an @EMPTY.NULL record
- The data contains empty values
- Modification of the dictionary is enabled (FIX, AUTOFIX, AUTOFIX DICT)

1. Such as nonnumeric values in numeric, date, and time fields, which can lead to adjusting the column type to CHARACTER.

For information about empty-null mapping, see [Empty or Unconvertible Data](#).

@SELECT

HS.SCRUB adds an @SELECT record to the file dictionary (but not to a table dictionary) if both of the following conditions are met:

- The dictionary does not already include an @ or @SELECT record
- Modification of the dictionary is enabled (FIX, AUTOFIX, AUTOFIX DICT)

For information about @SELECT records, see *UniVerse SQL Administration for DBAs*.

Running HS.SCRUB

To run the HS.SCRUB utility, enter **5** to choose **Run HS.SCRUB on a File/Table** from the UniVerse ODBC System Administration menu, or use the HS.SCRUB command.

If you use the **Run HS.SCRUB on a File/Table** option, UniVerse prompts to enter either the full path of the UniVerse account (for Windows platforms, start with the drive letter, for example D:) or the account name as listed in the UV.ACCOUNT file. Press **Enter** to see a list of UniVerse accounts in which file access has already been activated.

Next UniVerse prompts to enter the name of the table or file you want to analyze or change. When you enter a file name, UniVerse prompts to enter the mode of operation. Enter one of the following at the Mode prompt:

- Press **Enter** to generate a report without modifying the table or file.
- FIX
- AUTOFIX
- AUTOFIX DICT
- AUTOFIX DATA

See the next section for a description of these modes.

Using the HS.SCRUB Command

The syntax of the HS.SCRUB command is as follows:

HS.SCRUB *filename* [FIX | AUTOFIX [DICT | DATA]]

filename is the UniVerse file or table to be analyzed.

FIX puts HS.SCRUB in interactive mode, in which you are prompted to resolve any anomalies following the analysis.

AUTOFIX puts HS.SCRUB in automatic mode; anomalies are corrected with the default action that would have been presented to the user. If you don't specify DICT or DATA with the AUTOFIX option, HS.SCRUB resolves anomalies in both the data file or table and its dictionary.

DICT indicates that HS.SCRUB resolves only those anomalies associated with the file's or table's dictionary.

DATA indicates that HS.SCRUB resolves only those anomalies associated with the file's or table's data.

When neither FIX nor AUTOFIX is specified, HS.SCRUB only reports anomalies. No data or dictionary items are modified.

SQL Data Types

To fine-tune or define data type, length, precision, and scale values for fields and I-descriptors in files, you must edit the DATATYPE field of the corresponding dictionary entry (field 8 in D- and I-descriptors, field 6 in A- and S-descriptors). This is especially important for character data, in which the display width defined in the dictionary may be much larger or smaller than the largest data values in the file. The HS.SCRUB utility can automatically make these adjustments based on the data found.

For UniVerse files, UniVerse determines a field’s SQL data type by examining its conversion code and format specifications. The UniVerse database server reports this SQL data type to ODBC applications. If the UniVerse-generated SQL data type is inappropriate for the actual data in the field, you can specify the correct SQL data type in the field’s dictionary entry. For some data types, the SQL data type syntax is different between dictionary specifications and UniVerse SQL statements (for example, CREATE TABLE) as noted in the following table. Square brackets indicate optional parameters.

Dictionary Syntax	UniVerse SQL Syntax	Notes
CHAR [ACTER] [,n]	CHAR [ACTER] [(n)]	<i>n</i> = number of characters
DEC [IMAL] [,p [,s]]	DEC [IMAL] [(p [,s])]	<i>p</i> = precision <i>s</i> = scale
FLOAT [,p]	FLOAT [(p)]	<i>p</i> = precision
NUMERIC [,p [,s]]	NUMERIC [(p [,s])]	<i>p</i> = precision <i>s</i> = scale
VARCHAR [, n]	VARCHAR [(n)]	<i>n</i> = number of characters

SQL Data Type Syntax

Note the syntactic differences regarding the use of parentheses and commas.

The DATE, DOUBLE PRECISION, INT[EGER], REAL, SMALLINT, and TIME data type syntax is identical for dictionaries and UniVerse SQL.

You can specify the SQL data type for any field, real or virtual, in a UniVerse file. You need not specify the SQL data type for any column of a table or view defined by the CREATE TABLE or CREATE VIEW statement, but you may want to specify the SQL data type for other columns in the table or view (such as I-descriptors) that are not defined in the SICA. You cannot modify the SQL data type for columns defined in the SICA, and UniVerse ignores the dictionary definitions for these columns. For more information about the SICA and UniVerse SQL tables and views, see *UniVerse SQL Administration for DBAs* and *UniVerse SQL User Guide*.

Length of Character Data

In ODBC, every character column has either a fixed length or a maximum length. This column length is called its *precision*.

The precision of a character column is determined from one of the following:

- For a column defined by a CREATE TABLE or ALTER TABLE statement, the precision is defined by the column definition, which is stored in the table's SICA.
- For a column in a view, the precision is defined by the CREATE VIEW statement or by the precision of the column specified by the SELECT statement that creates the view.
- For a column not defined by the CREATE TABLE or ALTER table statement (such as an I-descriptor), or for a field in a UniVerse file, the precision is defined by the data type specified in the DATATYPE field of the field's dictionary definition. If the DATATYPE is not defined, the FORMAT field of the dictionary defines the precision.

The actual number of characters in a UniVerse character column can be greater than its precision. UniVerse ODBC retrieves such extra characters, up to a limit. The number of bytes of character data that UniVerse ODBC retrieves from a character column is the smallest of:

- The number of bytes of data the column actually contains.
- 255, or four times the column's precision, whichever is greater.
- The value of the ODBC statement option SQL_MAX_LENGTH, if it has been set.
- The number of bytes of data that UCI can fetch. This is controlled by the ODBC configuration parameters MAXFETCHBUFF and MAXFETCHCOLS, defined in the *uci.config* file.

If your fetch buffer is not big enough to hold all the character data that UniVerse ODBC retrieves, UniVerse ODBC fills your buffer and generates a truncation warning.

The actual number of characters in a UniVerse character column can be less than its precision: unlike some DBMSs, UniVerse does not automatically pad CHAR(*n*) columns on the right with spaces. If you insert the value "abc " (with two trailing spaces) into a CHAR(10) column, the column contains only five characters, not 10.

Your application and data source must agree on a consistent way to treat trailing spaces in a CHAR(*n*) column. Generally it is better to treat CHAR(*n*) columns as if they were VARCHAR columns with no space padding.

Empty or Unconvertible Data

UniVerse ODBC provides mechanisms for handling empty values (empty strings) and unconvertible data (dirty data).

Empty-Null Mapping

UniVerse files use empty strings in much the same way tables use null values. Unfortunately, empty strings in numeric or date columns cause data conversion errors in ODBC, making these columns almost inaccessible to many ODBC applications. To make files with empty values accessible to ODBC, UniVerse ODBC provides empty-null mapping, converting empty values in UniVerse files to null values in ODBC application buffers, and vice versa.

To turn on empty-null mapping for a UniVerse table or file, add an X-descriptor named @EMPTY.NULL to the dictionary. To turn off empty-null mapping, delete the @EMPTY.NULL entry from the dictionary.

Note: The **Activate Access to Files in an Account** option enables empty-null mapping in all files in the account by creating @EMPTY.NULL dictionary entries.

If empty-null mapping is not enabled, UniVerse ODBC does the following:

- Returns empty strings from empty character columns
- Translates empty strings in numeric or date columns to null values and returns them with a warning (SQL_SUCCESS_WITH_INFO)

If empty-null mapping is enabled, UniVerse ODBC assumes the user wants all empty strings in all columns to be returned as null values and therefore does not generate a warning.

Dirty Data

Dirty data such as nonnumeric or out-of-range values in a numeric column is unconvertible. Dirty data is tolerated without modifying the data or causing a run-time error to ODBC applications. When UniVerse ODBC encounters a dirty data value, it returns it as the null value along with a warning (SQL_SUCCESS_WITH_INFO).



Conversion Codes

When you insert or update a value in a column whose definition specifies a conversion code, UniVerse inserts the value without converting it, whether or not the value satisfies the conversion code. Subsequent SELECT statements, however, may not find such a value because UniVerse applies the conversion code to it and returns an empty string or some other unexpected value. This can occur with the following conversion codes among others:

- G (Group Extraction)
- L (Length)
- P (Pattern Match)
- R (Range)
- S (Substitution)

To avoid this problem, restrict your conversion codes to those that are essential. To enforce patterns on input values in tables, use a CHECK constraint.

Association Keys

UniVerse tables and views have primary keys. UniVerse files have record IDs. Primary keys and record IDs are unique identifiers for each row (record) of data in a table or file.

Since UniVerse ODBC shows associations and unassociated multivalued fields as ODBC tables, they too require a set of unique identifiers that serve as primary keys. These keys are called association keys.

You can define one or more association columns as the association key, but you do not have to. If you do not, UniVerse SQL generates a virtual column called @ASSOC_ROW containing unique values that, combined with the primary keys or record IDs of the base table, become the association keys for the ODBC table generated from the association.

The @ASSOC_ROW column can have either *stable* or *unstable* characteristics. A stable key allows an association to be managed as a static array of values without compaction (such as an association row representing a quarter, month, or day). An unstable key allows an association to be managed as a dynamic array with compaction (for example, names of multiple customer contacts and phone numbers).

***Note:** Even if you specify @ASSOC_ROW in the @SELECT or @ phrase, it appears as a column or field only in an ODBC table generated from the base table, view, or file. It does not appear as a column in the base table.*



Defining Association Keys

If a table has any associations or unassociated multivalued columns (such as I-descriptors) that were not defined by CREATE TABLE or ALTER TABLE, you can define association keys for them in two ways:

- Using the ASSOC clause of the CREATE TABLE or ALTER TABLE statement
- By adding an X-descriptor called @ASSOC_KEY.mvname to the table or file dictionary

For detailed information about defining association keys, see *UniVerse SQL Administration for DBAs* and *UniVerse SQL Reference*.



Note: *If you use the @ASSOC_KEY.mvname dictionary entry to define an association key, UniVerse enforces uniqueness on the combination of this key with the record ID for SQL INSERT and UPDATE statements. This means that attempts to update an association row fail if other rows contain the same record ID and association key values, even if you are not changing the key value as part of the update. To avoid this problem, make sure that any column or field you define as an association key has unique values within each record in the UniVerse file.*

ODBC Name Mapping

ODBC requires that table and column names begin with a letter, followed by either letters, digits, or underscores (see *Microsoft ODBC 2.0 Programmer's Reference*, Appendix C, "SQL Grammar").

Most UniVerse databases have file names, column names, and field names that do not conform to ODBC naming rules. Periods (.) are the most commonly used illegal characters. Several key ODBC client applications, including PowerBuilder and all Microsoft products (such as Access, Visual Basic, and Visual C++) that use the Jet database engine, reject periods in identifiers.

To make UniVerse data accessible to UniVerse ODBC, offending file names, column names, and field names are translated to conform to ODBC naming rules. The key features of name mapping are as follows:

- By default, name mapping is in effect to ensure that any UniVerse files, columns, and fields are ODBC-accessible.
- Name mapping is controlled on the client. For more information about the **UniVerse Name Mapping** option, see "[UniVerse ODBC Parameters](#)" on page 3-13.
- Once a UniVerse ODBC connection maps a UniVerse file, column, or field name to comply with ODBC, that mapping remains for the duration of the connection, even if the object is deleted and re-created. This prevents UniVerse ODBC from unexpectedly renaming an object.
- When name mapping is off, you can use non-ODBC compliant identifiers as long as they are delimited by double quotation marks in SQL statements.

The following algorithm translates names to conform to the ODBC 2.0 specification for user-defined names:

- Z_ is prefixed to names beginning with a digit.
- Z is prefixed to names beginning with an underscore or an ODBC-illegal character such as @.
- ODBC-illegal characters are converted to underscores throughout the name.
- Names are truncated to 120 characters.

If these transformations cause two or more names to conflict within a name space, UniVerse appends unique numbers in the following format to the names to distinguish them:

_n

The sequence number *n* is a decimal integer, up to seven digits long.

Name mapping is not applied in the following cases:

- When SQL statements or UniVerse commands are executed by the UniVerse ODBC native syntax extension, as described in [“The {NATIVE} Syntax Extension.”](#)
- When retrying as UniVerse SQL. See [“Retrying Statements as UniVerse SQL.”](#)

Troubleshooting

Isolating a Problem	5-4
Before Reporting a Problem	5-5
Getting Information About Your System	5-6
Manual Communications Verification	5-10
Verifying a TCP/IP Connection	5-10
Verifying Other Types of Network Connection	5-11
Connection Problems	5-11
Windows Server-Related Problems	5-12
UNIX Server-Related Problems	5-13
Windows and UNIX Server-Related Problems	5-14
Client-Related Problems	5-15

Beta Beta

This chapter describes how to isolate common operational problems and how to collect the pertinent information. It also contains checklists for troubleshooting UniVerse ODBC.

Isolating a Problem

Most installation and operation difficulties with UniVerse ODBC fall into one of the following categories:

- Windows operating system configuration
- PC hardware or software configuration
- Windows security, permissions, quotas, or disk space
- Nonfunctioning communications paths

If a client encounters a problem accessing the UniVerse database server, first try to access the server from a similarly configured client system that is working. Many problems are communications-related, so isolating them to the client or server is important.

In the DOS Command Prompt window, enter the Ping command to verify the network connection. In UniVerse ODBC Data Source Setup (called by the Microsoft ODBC Data Source Administrator tool), use the Test button to verify database access.

If Ping fails, the problem is communications-related, and you should examine whether other applications external to UniVerse ODBC (or even external to Windows) are visible to the server. Terminal emulators and FTP utilities should be checked to see if they fail similarly. For more information, see [Manual Communications Verification](#).

If Ping succeeds but Test fails, the problem is most likely on the server or in the configuration entry. Examine the configuration carefully. Correct any problems you find, then retry the test.

Before Reporting a Problem

Before you report a UniVerse ODBC problem, you need relevant information about your system. This includes the information provided in [Product Support](#) in the Preface.

Getting Information About Your System

This section describes how to find the information you may need for problem diagnosis or assistance. The following tables itemize the UNIX and Windows server information you may need. The Action column provides detailed procedures for getting the information.

Item	Action
UniVerse version number	Enter .L RELLEVEL at the UniVerse prompt.
Is the UniRPC daemon running?	Enter ps -ef grep unirpcd or ps -aux grep unirpcd , depending on your system.
TCP/IP host name of the UNIX server	Enter hostname at the UNIX prompt.
Contents of the UV account directory, and permissions on them	Change directory to the UV account directory and enter ls -l > logfile to write the permissions settings to a file.
Contents of the UV account directory, and permissions on them	Change directory to the UV account directory and enter ls -l > logfile to write the permissions settings to a file.
Contents of the <i>unishared</i> directory, and permissions on them	Change directory to the <i>unishared</i> directory and enter ls -l > logfile to write the permissions settings to a file.
Contents of the <i>unirpc</i> directory, and permissions on them	Change directory to the <i>unirpc</i> directory and enter ls -l > logfile to write the permissions settings to a file.
Disk usage of the UV account directory	Enter du UV.account.dir . <i>UV.account.dir</i> is the directory path where UniVerse is installed.
Available disk space on the file system where UniVerse is installed	Enter df filesystem on that file system. <i>filesystem</i> is the name of the UNIX file system that you are checking.

Server Information for UNIX

Item	Action
Windows version number	Enter WINMSD in the Run dialog box. Click the Version tab.
UniVerse version number	Enter L RELLEVEL at the UniVerse prompt.
Is UniVerse running?	Try to use <i>telnet</i> to reach the Windows host name. From the Windows Control Panel, choose Services . Verify that UniVerse is in the list of installed services and the status is Started.
Is the UniRPC service running?	Choose Services from the Windows Control Panel. Verify that UniVerse UniRPC is in the list of installed services and the status is Started.
TCP/IP host name of the Windows NT server	Enter hostname at the MS-DOS prompt.
Contents of the UV account directory, and permissions on them	Use the Windows Explorer.
Contents of the <i>unishared</i> directory, and permissions on them	Use the Windows Explorer.
Contents of the <i>unirpc</i> directory, and permissions on them	Use the Windows Explorer.
Available disk space on the file system containing UniVerse	Use the Windows Explorer.
Disk usage of the UV account directory	Use the Windows Explorer.

Server Information for Windows NT

The following table itemizes the Windows client information you may need. The Action column provides detailed procedures for obtaining each item of information.

Item	Action
UniVerse ODBC driver version number	Right-click UVODBC.DLL to check the version number.
Configuration file entry on which the problem occurred	Check the UCI.CONFIG file.
PC processor type and clock speed	Choose System from the Control Panel, then click the General tab.
PC main and extended memory size	Choose System from the Control Panel, then click the Performance tab.
Windows version	Choose System from the Control Panel, then click the General tab.
Type of Ethernet card installed	Look at the vendor documentation.
Pathname of the UniVerse ODBC driver installation directory	Use the Windows Explorer (usually <drive>:\IBM\UVODBC).
PATH environment variable value	Choose System from the Control Panel, then click the Environment tab.

Client Information for Windows NT, Windows 2000, or Windows XP

For diagnostic purposes it may also be necessary to get copies of the following files at the time of the failure.

For UNIX only:

- *.profile*, *.login*, and *.cshrc* (or other environment files) files for the user
- */etc/services* file
- */etc/inetd.conf* file
- */etc/rc** files
- *ps -aux* or *ps -ef* output file

For Windows Platforms only:

- \WINNT\SYSTEM32\DRIVERS\etc\HOSTS file
- \WINNT\SYSTEM32\DRIVERS\etc\LMHOSTS file
- \WINNT\SYSTEM32\DRIVERS\etc\SERVICES file
- \WINNT\SYSTEM32\DRIVERS\etc\NETWORKS file
- C:\WINNT\VSL.INI file
- C:\WINNT\WIN.INI file
- The client TCP/IP software HOSTS file (if used)
- The contents of the UniVerse ODBC driver installation directory (usually C:\IBM\UVODBC), with file sizes and dates

Manual Communications Verification

The UCI Config Editor and Microsoft ODBC Data Source Administrator tool are usually all that is needed to configure and validate UniVerse ODBC driver access to a UniVerse database server. However, if you are having difficulties, you can use manual procedures to further diagnose communications difficulties, as described in the following sections.

Verifying a TCP/IP Connection

TCP/IP products typically provide several utilities to verify connectivity between the UniVerse ODBC driver and the UniVerse database server. Consult the documentation for your TCP/IP software for detailed instructions for performing the following checks.

To verify a TCP/IP connection, use *ping* or *telnet*.

Ping

Provide the host name. The *ping* program looks up the host name in the local hosts file or in a name server and sends one or a series of low-level messages to the host, which echoes them back.

If using *ping* with the host name does not work, you can use *ping* with the IP address of the host. If the IP address works but the host name does not, either the local hosts file or the name server in use does not know the host by the name you are using. Check the spelling and capitalization (host names are case-sensitive).

If using *ping* with the IP address does not work to a system that is known to be available, you may have the wrong IP address, or there may be gateways in the Ethernet path between your client and the UniVerse database server that are not passing your requests. In this case, you need to get help from your system administrator.

If *ping* works, you have verified that the TCP/IP software has been installed and configured correctly, and the host name you used is valid (you still do not know if it is the host you want). Do not proceed until *ping* works.

Telnet

Provide the host name, then log on with a user name and password to an interactive host terminal session. For information about how to use the *telnet* utility, see your TCP/IP software manual.

When you can log on successfully to the correct host through *telnet*, you have verified that your local hosts file or name server has the correct IP address for that host name, and that the user name and password you used are valid on that host.

Verifying Other Types of Network Connection

The procedure for verifying other types of communications link (including asynchronous communications) depends on the type of hardware and software you have.

Connection Problems

If your client application does not show the detailed error message, you may need to create either a client log file or a `ubodbc.log` file in the working directory to verify the error.

- To create a client log file, use Dr. DeeBee Spy on the ODBC data source. For more information on Dr. DeeBee Spy, see [ODBC Diagnostic Tool: Dr. DeeBee Spy](#).
- To create a `uvodbc.log` file, use the Microsoft ODBC Data Source Administrator tool. Click the Option button in the UVODBC Data Source Setup dialog box and select the checkbox for enabling a log file.

For more information on log files, see Appendix D, “[Driver Process Log Files](#).”

Windows Server-Related Problems

- Is the correct version of the Windows operating system installed?
- Is the UniRPC service running?
- Is there enough memory and paging space?
- Is there correct access to the UV account directory for all UniVerse ODBC user IDs?
- Are the permissions set correctly on UniVerse accounts for intended users (this may require read/write access to the VOC file)?
- Is the UniVerse Resource service installed and started?
- Does other TCP-based software use too many TCP connections?

UNIX Server-Related Problems

- Is the correct version of the UNIX operating system installed?
- Is the UniRPC daemon (*unirpcd*) running?
- Is there enough memory and paging space?
- Is there correct access to the UV account directory for all UniVerse ODBC user IDs?
- Are the correct permissions set on UniVerse accounts for intended users? (This can require write access to the VOC file.)
- Does other TCP-based software use too many TCP connections?
- Make sure that *.cshrc* does not change terminal parameters (*.login* and *.profile* do not need to be run), and does not echo anything to standard output.
- Are there entries in */etc/hosts* for all UniVerse ODBC client machines?

Windows and UNIX Server-Related Problems

- Does the UniVerse System Administration menu option **Activate Access to Files in an Account** run successfully on all UniVerse accounts to be accessed (as evidenced by the HS.INSTALL.LOG file in the UniVerse account)?
- Are all functions used by virtual fields or fields with conversion or correlative codes cataloged and listed in the account's VOC?
- Are secondary indexes defined on likely query fields where possible?
- Are the file user permissions in the HS_FILE_ACCESS file in this account set correctly, as is the user-specific record based on the login used in the client's configuration file?
- Does the record ID field appear in the dictionary of an updatable file?

Client-Related Problems

- Does the system have the required configuration of memory, processor type, and operating system?
- Is the supported Ethernet card installed correctly?
- Is the client TCP/IP software installed correctly, and is it verified to communicate with this host?
- If your client TCP/IP software is configured to use a HOSTS file, is the HOSTS file accessible by the path (PATH environment variable on Windows)?
- Do the host names in the configuration file `uci.config` match those in the HOSTS file (case must match) or the name server?
- Do the HOSTS file entries for the host names used contain the correct IP addresses?
- Does the HOSTS file specify the IP address of the first Ethernet controller when multiple controllers are present?
- Was the UniVerse ODBC driver installation completed successfully?
- Make sure that other Windows applications do not take up too much memory.
- Are the host names in the configuration file `uci.config` spelled and capitalized correctly?
- Are you using the correct user names and passwords?
- Are the database parameters in the configuration file `uci.config` absolute paths?
- Is the server parameter in the configuration file `uci.config` an absolute path, and does it correspond to the UV account directory?

ODBC Usage Notes

A

This appendix describes the UniVerse implementation of ODBC, including:

- [Data Types in SQL](#)
- [Support for core and extended ODBC SQL grammar](#)
- [UniVerse Extensions to ODBC SQL Grammar](#)
- [Procedures](#)
- [Executing UniVerse SQL](#)
- [SQL-related connection options](#)
- [Transaction support](#)
- [ODBC table types](#)

Data Types in SQL

UniVerse ODBC supports the following SQL data types. Use the UniVerse SQL data types in CREATE TABLE and ALTER TABLE statements, and the corresponding ODBC SQL data types in calls to the ODBC API.

UniVerse SQL Type	ODBC SQL Type
CHARACTER [(n)]	SQL_CHAR
DATE	SQL_DATE
DECIMAL [(p[,s])]	SQL_DECIMAL
DOUBLE PRECISION	SQL_DOUBLE
FLOAT [(p)]	SQL_FLOAT
INTEGER	SQL_INTEGER
NUMERIC [(p[,s])]	SQL_NUMERIC
REAL	SQL_REAL
SMALLINT	SQL_SMALLINT
TIME	SQL_TIME
VARCHAR [(n)]	SQL_VARCHAR ¹

1. The 254-character limit for strict VARCHAR definition is relaxed to allow longer data values, although some ODBC application tools may assume this limit and truncate longer values.

For information about specifying UniVerse SQL data types in UniVerse files, see [SQL Data Types](#).

Support for Core ODBC SQL Grammar

In response to **SQLGetInfo**, UniVerse ODBC says that it supports the core ODBC SQL grammar. More accurately, it supports *most* of the core grammar. The parts it does not support are:

- Table name qualifiers in DDL. For example:

```
DROP TABLE MY_SCHEMA.MY_TABLE;
```

- The CASCADE and RESTRICT qualifiers of the REVOKE statement. For example:

```
REVOKE UPDATE, DELETE ON MY_TABLE FROM PUBLIC CASCADE;
```

- LIKE USER. For example:

```
SELECT * FROM MY_TABLE WHERE NAME LIKE USER;
```

- Qualified index names in CREATE INDEX, and unqualified names in DROP INDEX. For example:

```
CREATE INDEX MY_SCHEMA.MY_INDEX ON MY_TABLE (MY_COLUMN);  
DROP INDEX MY_INDEX;
```

Support for Extended ODBC SQL Grammar

UniVerse ODBC supports some of the extended ODBC SQL grammar.

SELECT...FOR UPDATE Statement

UniVerse ODBC supports the SELECT...FOR UPDATE version of the SELECT statement. The syntax is as follows:

```
SELECT [ ALL | DISTINCT ] column_specifications
      FROM table_specification
      [ WHERE clause ]
      FOR UPDATE [ OF column [, column] ... ]
```

SELECT...FOR UPDATE locks selected rows with exclusive record or file locks, letting users update or delete the selected rows without having to wait for other users' or processes' locks to be released. For example:

```
SELECT ORDERS.CUSTNAME, CUSTOMER.NAME
      FROM ORDERS, CUSTOMER
      WHERE ORDERS.CUSTNO = CUSTOMER.CUSTNO
      FOR UPDATE OF CUSTOMER.NAME;
```

Outer Joins

UniVerse ODBC supports left outer joins in statements such as:

```
SELECT CODE AS STATE, COMPANY
      FROM OJ_STATES
      LEFT OUTER JOIN
      CUSTOMER
      ON STATES.CODE = CUSTOMER.STATE;
```

However, UniVerse ODBC does not support nested left outer joins, such as:

```
SELECT CODE AS STATE, COMPANY, PRODID
      FROM OJ_STATES
      LEFT OUTER JOIN
      CUSTOMER
      LEFT OUTER JOIN
      CUSTOMER_ORDERS
      ON CUSTOMER.CUSTID = CUSTOMER_ORDERS.CUSTID
      ON STATES.CODE = CUSTOMER.STATE;
```

In UniVerse SQL, the table expression on the left side of a left outer join can itself be a left outer join, but the table expression on the right side cannot. In ODBC SQL, the reverse is true: only the table expression on the right side can be a left outer join. Therefore, in UniVerse ODBC SQL, neither the left nor the right table expression in a left outer join can be a left outer join.

UNION Clause

UniVerse ODBC supports the UNION clause as given in the ODBC grammar. For example:

```
SELECT SCHEMA_NAME FROM CATALOG.UV_SCHEMA
UNION
SELECT TABLE_NAME FROM CATALOG.UV_TABLES;
```

ESCAPE Clause in the LIKE Predicate

UniVerse ODBC supports the ESCAPE clause in the LIKE predicate, which lets you use % and _ as literal characters in a search pattern. For example:

```
SELECT * FROM PROJECTS
WHERE STATUS LIKE '%80\% complete%' ESCAPE '\';
```

Date and Time Types

UniVerse ODBC supports DATE and TIME in CREATE TABLE and ALTER TABLE statements. It also supports date and time literals. For example:

```
CREATE TABLE HALS (BIRTHDAY DATE);
INSERT INTO HALS VALUES ({d '1997-01-12'});
```

Scalar Functions

The following table lists UniVerse ODBC support of scalar functions.

	Function	Supported
String	ASCII	No
	CHAR	No
	CONCAT	Yes
	DIFFERENCE	No
	INSERT ¹	Yes
	LCASE	Yes
	LEFT	Yes
	LENGTH	Yes
	LOCATE	No
	LTRIM	Yes
	REPEAT	No
	REPLACE	No
	RIGHT ¹	Yes
	RTRIM	Yes
	SOUNDEX	No
	SPACE	No
	SUBSTRING	Yes
	UCASE	Yes
Numeric	ABS	No
	ACOS	No

UniVerse ODBC Support of Scalar Functions

	Function	Supported
	ASIN	No
	ATAN	No
	ATAN2	No
	CEILING	No
	COS	No
	COT	No
	DEGREES	Yes
	EXP	No
	FLOOR	No
	LOG	No
	LOG10	No
	MOD	No
	PI	Yes
	POWER	No
	RADIANS	Yes
	RAND	No
	ROUND	No
	SIGN	No
	SIN	No
	SQRT	No
	TAN	No
	TRUNCATE	No
Time and Date	CURDATE	No
	CURTIME	No

UniVerse ODBC Support of Scalar Functions (Continued)

	Function	Supported
	DAYNAME	No
	DAYOFMONTH	No
	DAYOFWEEK	No
	DAYOFYEAR	No
	HOUR	No
	MINUTE	No
	MONTH	No
	MONTHNAME	No
	NOW	No
	QUARTER	No
	SECOND	No
	TIMESTAMPADD	No
	TIMESTAMPDIFF	No
	WEEK	No
	YEAR	No
System	DATABASE	Yes
	IFNULL	No
	USER	Yes
Conversion	CONVERT ²	Yes

UniVerse ODBC Support of Scalar Functions (Continued)

1. INSERT and RIGHT may evaluate their arguments more than once, consequently you cannot use a parameter marker as an argument to these functions, and you must watch out for side effects if you use an I-descriptor.
2. UniVerse ODBC supports the same conversions that UniVerse SQL CAST supports.

An example of a call to a scalar function is as follows:

```
SELECT * FROM UV_SCHEMA
WHERE SCHEMA_NAME = {fn DATABASE()};
```

ODBC implies that no scalar functions can take an arithmetic expression or a parameter marker as an argument. UniVerse ODBC tolerates violations of this rule, but in the future it may optionally catch them and give an error message.

UniVerse Extensions to ODBC SQL Grammar

UniVerse ODBC supports the following driver-specific extensions to the SQL grammar.

The following extensions are always available:

- Mixed-case keywords
- Multiline statements
- End-of-line comments beginning with two hyphens (--)
- The {NATIVE} statement

The following extensions are available only when the **Reject non-ODBC SQL Syntax** option is off (the default setting):

- The ORDER BY clause in SELECT...FOR UPDATE statements
- The COUNT (*expression*) clause
- The EXPLAIN and NOWAIT keywords
- The concatenation operator (||)
- The following scalar functions:
 - CHAR[ACTER]_LENGTH
 - CONCAT
 - LOWER
 - UPPER
- The following values in the VALUES clause of an INSERT statement:
 - Literals
 - The null value
 - USER
 - ODBC scalar functions
 - The UniVerse scalar functions CHAR_LENGTH, CONCAT, LOWER, and UPPER
 - Combinations of the above with arithmetic operators, the concatenation operator, or ODBC scalar functions
- SOME instead of ANY or ALL before a subquery

NOWAIT Keyword

UniVerse ODBC supports the NOWAIT keyword at the end of the following statements:

- SELECT (including SELECT...FOR UPDATE)
- INSERT
- UPDATE
- DELETE

NOWAIT prevents users from having to wait if another user or process has a file or record lock that prevents the database operation from proceeding. For example:

```
INSERT INTO SALESMEN
  SELECT * FROM EMPS
  WHERE DEPTCODE = 'SALES'
NOWAIT;
```

EXPLAIN Keyword

UniVerse ODBC supports the EXPLAIN keyword for use in SELECT statements. EXPLAIN returns information about how the statement will be processed, letting users decide if they want to rewrite the query more efficiently. The SELECT statement is not executed.

SQL Usage Notes and Restrictions

The following issues also affect SQL usage in UniVerse ODBC:

- [INSERT statements](#)
- [Parameter markers](#)
- [DDL](#)
- [ODBC-compliant names](#)
- [Column aliases](#)

INSERT Statements

UniVerse handles SQL INSERT statements that do not include an explicit column list as documented in *UniVerse SQL Reference*. For files, the values listed in the VALUES clause should match the fields defined in the @INSERT phrase. For tables, the values should match the columns defined in the SICA.

Parameter Markers

ODBC SQL has certain restrictions on the use of parameter markers. UniVerse ODBC tolerates some violations of these restrictions, but in the future it may optionally catch them and give an error message.

No DDL on Files

SQL DDL statements that create, drop, alter, grant, or revoke privileges on tables are permitted on tables and views only. To use DDL statements on a file, the file must be converted to a table.

Tolerance of ODBC-Compliant Names

When name mapping is on (the default), and the **Retry as UniVerse SQL** option is off, UniVerse ODBC returns an error if an SQL statement includes an ODBC qualifier name, table name, or column name that is not ODBC-compliant. However, UniVerse ODBC does not check the ODBC compliance of index names, correlation names, or column aliases.

Column Aliases

The UniVerse ODBC interpretation of column aliases is closer to that of SQL-92 than to that of UniVerse. This interpretation is better suited for the default behavior of ODBC. UniVerse ODBC permits references to column aliases only in ORDER BY clauses. As of Release 9, UniVerse permits references to column aliases in other places.

Procedures

All UniVerse procedures are available through UniVerse ODBC as ODBC procedures. These include, but are not limited to, UniVerse BASIC programs and paragraphs. For details about UniVerse procedures, see *UCI Developer's Guide* or *UniVerse BASIC SQL Client Interface Guide*. For details about ODBC procedures, see *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

To invoke procedures using UniVerse ODBC, you need to write custom SQL statements, and, depending on your ODBC application, you may need to write custom ODBC calls.

Syntax

UniVerse ODBC supports the following ODBC syntax for procedure calls:

`{ CALL procedure [(parameter1, ... parametern)] }`

The braces are part of the syntax and must be specified. You can specify parameters as bound parameters (?) or as literals.

You can specify parameters only when calling a UniVerse BASIC subroutine. When calling other UniVerse procedures, enclose the entire command in double quotation marks. For example:

`{CALL "LIST CUSTOMER WITH CUSTID = 2"}`

If *procedure* contains periods or is otherwise not ODBC-compliant, you must enclose it in double quotation marks. For example:

`{CALL "MY.BASIC.SUBROUTINE" (?, ?, ?)}`

You might like to call MY.BASIC.PROGRAM as MY_BASIC_PROGRAM, but this is not possible because UniVerse ODBC does not apply its name-mapping feature to procedure names.

Parameters

You can bind a procedure parameter using any SQL type that UniVerse ODBC supports (see [Data Types in SQL](#)). However, the UniVerse engine itself treats all procedure parameters as character strings. Therefore, binding such a parameter using an SQL type other than `SQL_CHAR` or `SQL_VARCHAR` makes UniVerse ODBC check that the value of the parameter is compatible with the SQL type, and generate an error or warning if it is not.

You can bind any procedure parameter as input, input/output, or output. If you bind a procedure parameter as input/output or output and successfully execute the procedure, UniVerse ODBC always returns some value for the parameter.

In UniVerse ODBC, the values of output parameters are available immediately after the return of **SQLExecute** or **SQLExecDirect**. But this is not true of all ODBC drivers; according to *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*, a driver is not guaranteed to return the output parameters of a procedure until after all of the procedure's results have been fetched. For maximum interoperability, you should not attempt to read output parameter values until then.

If an error occurs during the process of returning a value for an output parameter, UniVerse ODBC may return a null value for the parameter.

Result Sets

A procedure may or may not return a result set.

The number and types of columns in the result set of a procedure are not available until the procedure is executed. **SQLPrepare** immediately followed by **SQLNumResultCols**, **SQLDescribeCol**, or **SQLColAttributes** gives an error.

The number and types of columns in the result set of a procedure may change from execution to execution.

Behavior of SQLRowCount

SQLRowCount returns a value of `-1` after a procedure that returns a result set.

After a procedure which does not return a result set, **SQLRowCount** returns what the UCI **SQLRowCount** function returns. This might not be what you expect; see *UCI Developer's Guide* for more information.

Limitations

UniVerse ODBC does not support the following ODBC procedure functionality:

- Return values from procedures. For example, the following statement generates an error:

```
{ ? = CALL MYPROC ( ?, ? ) }
```

The ? = is used for procedures with return values. UniVerse procedures can return data through any parameter, but they do not have return values as such, so UniVerse ODBC does not support this feature.

- Procedure names qualified with a qualifier or an owner. For example, the following statement generates an error:

```
{ CALL MYSCHEMA.MYPROC ( ?, ? ) }
```

This syntax is legal in ODBC data sources in which procedures can be qualified with a qualifier or owner name, or both. It is illegal in UniVerse ODBC because UniVerse does not support qualified procedure names. See *UCI Developer's Guide* or *UniVerse BASIC SQL Client Interface Guide* for more information.

If your procedure name contains periods, it must be quoted, as noted earlier.

- Catalog information.

ODBC defines two functions that return catalog information about procedures. **SQLProcedures** returns a list of procedures.

SQLProcedureColumns returns a list of parameters and of columns in the result sets of procedures. In UniVerse ODBC these functions always return empty result sets.

Consequently, to call a procedure, you must already know its name and the number and types of its parameters. Each time you execute a procedure, if the execution produces a result set, you can find out about the number and types of columns in the result set.

- Default values of procedure parameters.

Interoperability with Generic ODBC Applications

The UniVerse ODBC implementation of procedures differs from that of other ODBC drivers in these respects:

- **SQLProcedures** and **SQLProcedureColumns** always return empty result sets.

- You cannot call **SQLPrepare**, and then call **SQLNumResultCols**, **SQLDescribeCol**, or **SQLColAttributes** without first calling **SQLExecute**.
- You cannot prepare a procedure, then execute it multiple times, with the guarantee that the number and types of columns in its result set will be the same after each execution.

A generic ODBC application may not expect this behavior. Therefore, to use UniVerse ODBC procedures, you probably need to write custom SQL calls, and depending on your ODBC application, you may need to write custom ODBC calls.

Executing UniVerse SQL

You can pass an ODBC application's SQL syntax directly to UniVerse. This bypasses UniVerse ODBC's SQL processor, which performs operations such as parsing, name mapping, and validating SQL syntax, and directly accesses UniVerse's SQL interpreter. You can do this using one of the following methods:

- Using the UniVerse ODBC-specific { NATIVE } syntax extension
- Configuring the ODBC data source to retry statements as UniVerse SQL
- Configuring the ODBC data source not to require strict ODBC syntax

The {NATIVE} Syntax Extension

To let you use native SQL syntax that conflicts with ODBC-standard SQL, UniVerse ODBC supports an SQL extension that allows the direct pass-through of native statements (SQL or other database-specific commands) to the database without interpretation. The syntax for this extension is:

`{ NATIVE statement }`

The braces are part of the syntax and must be specified. *statement* is the single- or double-quoted string to pass to the underlying database without translation. If you use single quotation marks to delimit *statement* and want to include a literal single quotation mark in *statement*, double the single quotation mark. Similarly, if you use double quotation marks to delimit *statement* and you want to include a literal double quotation mark in *statement*, double the double quotation mark.

UniVerse ODBC does not support name mapping or file access control in this extension.

Retrying Statements as UniVerse SQL

In an ODBC connection, retrying statements as UniVerse SQL can be on or off. It is off by default.

When retrying statements as UniVerse SQL is on, if UniVerse ODBC fails to interpret an SQL statement as a valid ODBC SQL statement, it retries the statement; that is, it tries to interpret the statement as though it were written in UniVerse SQL. If your ODBC application was developed to use the special features of UniVerse SQL, you may want to turn this option on by setting the HS_RETRY_AS_UV_SQL environment variable to YES or ON, or by clicking **Yes** at the **Retry as UniVerse SQL** in the UniVerse ODBC Data Source Setup dialog box.

When this option is off, UniVerse ODBC always tries to interpret SQL as ODBC SQL (unless a statement conforms to UniVerse ODBC's native SQL extension, in which case, UniVerse ODBC always passes the statement through). If you want your ODBC application to be portable across multiple databases, do not activate this option. In addition, you can set one or more of the options that enforce ODBC syntax. By doing so you ensure that the application's SQL statements are portable across databases and provide uniform error messages when SQL does not meet portability standards.

Retrying statements as UniVerse SQL takes effect only when name mapping is off. This option bypasses the translation of the client's SQL statement from ODBC SQL to UniVerse SQL, in which ODBC names are converted to UniVerse equivalents. If you want to use UniVerse extensions directly through this option, you also must use UniVerse native file, table, and column names in your SQL statements.

UniVerse ODBC does not support file access control in retried statements.

Retrying statements as UniVerse SQL is useful mainly to support custom ODBC applications that "know" they are accessing UniVerse databases. You should turn this option off when you use it with commercial ODBC-aware applications.

SQL-Related Connection Options

The client can use the UniVerse ODBC Data Source Setup dialog box to set many SQL-related UniVerse ODBC connection options. For more information, see [Configuring UniVerse ODBC Driver Software](#).

Transaction Support

UniVerse ODBC supports two modes of transaction behavior:

- Autocommit mode (the default when the server starts)
- Manual mode

Autocommit Mode

In autocommit mode, UniVerse ODBC treats every SQL statement received from a client as a separate transaction within the data session. It commits every statement that could have modified the database.

DDL statements such as CREATE TABLE are allowed only when the driver is in autocommit mode.

Multiple Statement Handles

Whenever a transaction is committed or rolled back, all open statement handles on that connection are closed. If your application is using autocommit mode and it executes an INSERT, UPDATE, or DELETE SQL statement on one ODBC statement handle, all other statement handles (such as for SELECT statements) are closed and cannot be used for additional **SQLFetch** operations. If you want to use **SQLFetch** to fetch rows one at a time from one SELECT statement, and you want to execute INSERT, UPDATE, or DELETE statements on another ODBC statement handle (this might be typical for a records-maintenance application), use manual mode.

For more information on cursor commit behavior and transaction modes in ODBC, see *Microsoft ODBC 2.0 Programmer's Reference and SDK Guide*.

Manual Mode

In manual mode the server runs the SQL statements as they come in from the client, but it defers committing changes to the database until it receives an explicit commit or rollback command from the client.

ODBC client applications request transaction commits or rollbacks internally by calling the ODBC **SQLTransact** function on the UniVerse ODBC driver. Each ODBC client application has its own user interface to control transactions.

ODBC Table Types

An ODBC table's type is presented in the TABLE_TYPE column of the **SQLTables** result set. UniVerse ODBC maps UniVerse file structures to ODBC table types as follows:

UniVerse File Type	ODBC Table Type
Table	TABLE
View	VIEW
File	TABLE

ODBC Table Types

Error Messages

B

This appendix describes the UniVerse ODBC error reporting system. It also explains how to obtain information about an error.

Message Descriptions

Each message description includes the Error ID, Severity, and Facilities codes. Because some errors originate in other products such as your database or TCP/IP product, it is not possible to list every error message. Instead, the word *STRING* refers to a portion of the message that varies depending on the server platform or database. In these cases, you may need to refer to your product's documentation for details.

Errors are returned as a set of codes to an application program or as text to a user. In either case, an error consists of four components:

Component	Description
Message	A brief description of the error, displayed on the screen.
Error ID	The numeric code that identifies the specific error or warning.
Severity	<p>The error's level of severity. Informational or warning messages do not prevent further processing. Errors, severe errors, or fatal errors usually do. The severity levels are as follows:</p> <p>SUCCESS – Successful completion.</p> <p>INFO – Successful completion with extra information.</p> <p>WARNING – Completion with information such as truncation or roundoff errors.</p> <p>ERROR – Failure to complete operation. UniVerse ODBC is still functioning, but an exceptional situation is reported (such as a locked record or invalid parameters).</p> <p>SEVERE – Failure to complete operation and UniVerse ODBC system failure. Try to save what you can and discontinue.</p> <p>FATAL – Failure of the UniVerse ODBC system. This is most commonly caused by the lack of virtual memory and by communications failures. Continued use of the UniVerse ODBC connection is likely to fail.</p>
Facility	The source of the error, usually the software module from which the error originated. This information is useful when calling technical support.

Error Components

Messages

A CREATE VIEW statement's name list did not correspond to the query defining the view.

Error ID: 10, Severity: ERROR, Facility: DBCAPERR

ADdIng a column to a table failed because there is already a column with the specified name.

Error ID: 30, Severity: ERROR, Facility: DBCAPERR

An INSERT statement contained the wrong number of values for the target table.

An INSERT statement contained the wrong number of values for the target table. Verify that the number of parameter markers and values supplied are consistent with the table's definition.

Error ID: 9, Severity: ERROR, Facility: DBCAPERR

An invalid HSTMT was encountered.

The HSTMT associated with this SQL statement was either invalid or equal to SQL_NULL_HSTMT when it should have been valid.

Error ID: 61, Severity: ERROR, Facility: DBCAPERR

An invalid parameter type has been specified.

Error ID: 38, Severity: ERROR, Facility: DBCAPERR

An operation in progress could not be canceled.

Error ID: 24, Severity: ERROR, Facility: DBCAPERR

A positioned UPDATE or DELETE failed to modify any rows.

An UPDATE or DELETE WHERE CURRENT OF *cursorname* did not modify any rows in the SELECT FOR UPDATE statement on *cursorname*.

Error ID: 7, Severity: ERROR, Facility: DBCAPERR

A positioned UPDATE or DELETE modified more than 1 row.

An UPDATE or DELETE WHERE CURRENT OF *cursorname* affected more rows than the last row fetched by *cursorname*.

Error ID: 8, Severity: ERROR, Facility: DBCAPERR

Argument to Scalar Function not compatible with parameter type.

A literal value supplied to a scalar function is not compatible with the required parameter type.

Error ID: 42, Severity: ERROR, Facility: FPSRVERR

Argument value out of range.

An argument value was outside the allowable range for this request.

Error ID: 45, Severity: ERROR, Facility: FPSRVERR

Attempt to connect to a database which uses string and/or identifier delimiters which UV/ODBC does not support.

UniVerse ODBC supports only those databases in which the delimiter character for SQL string literals is single quotation marks, and the delimiter character for SQL identifiers is double quotation marks. The user attempted to connect to a database that did not have this property.

Error ID: 66, Severity: ERROR, Facility: DBCAPERR

Bad socket number.

This error is an internal communications error. Disconnect and reconnect. If that fails, reboot the PC.

Error ID: 101, Severity: SEVERE, Facility: LINKERR

CREATE INDEX failed because the specified index already exists.

CREATE INDEX failed because the specified index already exists.

Error ID: 28, Severity: ERROR, Facility: DBCAPERR

CREATE TABLE/VIEW failed because a table or view with that name already exists.

CREATE TABLE or CREATE VIEW failed because a table or view with that name already exists.

Error ID: 26, Severity: ERROR, Facility: DBCAPERR

Cannot interpret input data. Versions incompatible.

Attempt to read a file or connect to a server failed because of incompatible software versions.

Error ID: 15, Severity: FATAL, Facility: OIOERR

Cannot read a directory as a file.

Attempt to read data from a directory failed. Check your file pathname.

Error ID: 11, Severity: ERROR, Facility: OIOERR

Client timed out waiting for initial server response.

The UniVerse ODBC driver timed out waiting for the initial response from the UniVerse database server. If the client's login timeout interval is too short, you can increase it by setting the SQL_LOGIN_TIMEOUT connection option to a higher value.

Error ID: 5, Severity: ERROR, Facility: OCLIERR

Column number out of bounds.

A numeric reference to a column failed because the column number was out of bounds.

Error ID: 32, Severity: ERROR, Facility: DBCAPERR

Communications driver not loaded.

TCP/IP network driver software has not been loaded or is not running. Check to see that it has been installed correctly.

Error ID: 3, Severity: FATAL, Facility: OIOERR

Communications error. STRING

A communications error has occurred. The connection has been closed. Retry the connection. If this message persists, contact IBM Customer Support.

Error ID: 123, Severity: SEVERE, Facility: LINKERR

Connection closed from server.

The UniVerse database server process has terminated abnormally. This could be caused by the host computer being rebooted, or it could indicate a problem in the server. Check that the server is running. Disconnect and reconnect. If that fails, reboot the PC.

Error ID: 109, Severity: SEVERE, Facility: LINKERR

Connection has been aborted.

This error can result from a variety of situations, such as the host is shut down, the host operator force logged out the server process, the host inactivity time-out expired and logged out the server process, TCP/IP has failed, or a fatal error occurred to the server process. Retry the connection. If this fails, reboot your PC and try again. If this fails, check to see if the host and TCP/IP are running.

Error ID: 102, Severity: SEVERE, Facility: LINKERR

Connection has closed.

The UniVerse ODBC driver is having trouble initiating communication to the server. This is often caused by invalid or no parameters in the configuration file uci.config.

This error can also result from a variety of situations, such as the host is shut down, the host operator force logged out the server process, the host inactivity time-out expired and logged out the server process, TCP/IP has failed, or a fatal error occurred to the server process. Retry the connection. If this fails, reboot your PC and try again. If this fails, check to see if the host and TCP/IP are running.

Error ID: 103, Severity: SEVERE, Facility: LINKERR

Connection is not open.

The UniVerse ODBC driver is having trouble initiating communication to the server. This is often caused by a host being down or by invalid link parameters in the configuration file.

Use UCI Config Editor to check the spelling of the host name.

Retry the connection. If this fails, reboot your PC and try again. If this fails, check to see if the host and TCP/IP are running.

Error ID: 104, Severity: SEVERE, Facility: LINKERR

Connection or statement option was set to a value other than that specified.

The driver does not support the requested value of the specified option and substituted a similar value.

Error ID: 6, Severity: WARNING, Facility: DBCAPERR

Connection timed out.

This reports an attempt to connect to a host that did not accept the connection within the time limit set in the local PC TCP software (usually several minutes). This can happen if the host is not running TCP/IP, or is not running at all. It can also occur if the system is very heavily loaded and other UniVerse ODBC users are also trying to connect at the same time. Check the host system and try again.

This error can also occur during the course of normal communications if the host or client system becomes very heavily loaded or hangs. In this case, check both the host and the client. If the problem persists, reboot both systems and try again.

Error ID: 107, Severity: SEVERE, Facility: LINKERR

Conversion method unknown.

The type of an argument supplied to a conversion function is not compatible with the type expected by that function. If the ODBC data source is configured with **Enforce ODBC Function Argument Types** set, arguments to conversion functions must be compatible with expected data types.

Error ID: 44, Severity: ERROR, Facility: FPSRVERR

Conversion of a value to a numeric type caused an underflow.

A text- or number-to-number conversion failed because the number was too small or too far negative to be represented in the specified format.

Error ID: 26, Severity: ERROR, Facility: DATUMERR

Could not login to database *STRING* as user *STRING*

A database session could not be established because the database name, user name, or password was invalid.

Error ID: 3, Severity: FATAL, Facility: DBCAPERR

Could not start server. *STRING*

The server process could not be started for the indicated reason. Correct the problem and try again.

Error ID: 124, Severity: SEVERE, Facility: LINKERR

Could not make name unique: more than 10 million names with the same stem.

Could not find a `_nnnnnnn` suffix for a name, which would make it unique within a space of names because 9,999,999 suffixes have already been given out for that name. This is an *extremely rare* error. Contact IBM Customer Support.

Error ID: 45, Severity: ERROR, Facility: DATUMERR

Cursor could not honor a FETCH_SAME_AS_LAST request.

The database interface could not fetch the same number of rows as the last request because there was no last request.

Error ID: 11, Severity: ERROR, Facility: FPSRVERR

Cursor not found with given name.

The indicated cursor name does not correspond to any allocated cursor.

Error ID: 5, Severity: ERROR, Facility: FPSRVERR

Cursor number not currently allocated.

The indicated cursor number is not allocated.

Error ID: 6, Severity: ERROR, Facility: FPSRVERR

DDL statements are illegal within a Transaction.

DDL statements are illegal within a transaction.

Error ID: 62, Severity: ERROR, Facility: DBCAPERR

Database capsule creation failed.

Error ID: 9, Severity: FATAL, Facility: FPSRVERR

Database error.

The database encountered an error while performing the operation. See your database manual for an explanation of the message and external error code.

Error ID: 1, Severity: ERROR, Facility: DBCAPERR

Database name/uid/password parameters must be textual.

User name and password strings must be alphanumeric. If your user name or password could be interpreted as a number, currency, date, or time, enclose it in double quotation marks.

Error ID: 13, Severity: SEVERE, Facility: LINKERR

Database reported a miscellaneous warning.

The database reported a warning to UniVerse ODBC but did not specify the nature of the problem.

Error ID: 37, Severity: WARNING, Facility: DBCAPERR

Database table integrity constraint violation.

An operation was attempted that would have violated integrity constraints on a database table.

Error ID: 17, Severity: ERROR, Facility: DBCAPERR

Database warning.

The database generated warnings while performing the operation. See your database manual for an explanation of the message and external error code.

Error ID: 2, Severity: WARNING, Facility: DBCAPERR

Data conversion error: STRING

The data in this column is invalid for the defined SQL data type. The most common reason is nonnumeric data in numeric or date/time columns.

Error ID: 67, Severity: WARNING, Facility: DBCAPERR

Data source does not support default values of parameters of procedures.

Error ID: 62, Severity: ERROR, Facility: FPSRVERR

Data source does not support table name qualifiers.

Error ID: 37, Severity: ERROR, Facility: FPSRVERR

Data source not capable.

A construct was passed to the data source for which no support is provided.

Error ID: 36, Severity: ERROR, Facility: FPSRVERR

Data source not capable due to Dynamic Parameter.

This data source does not support a dynamic parameter value in this position.

Error ID: 43, Severity: ERROR, Facility: FPSRVERR

Data truncated.

All of the data for the specified column, *icol*, could not be retrieved in a single call to the function.

Error ID: 28, Severity: WARNING, Facility: OIOERR

Data truncated entering or leaving the database.

Either more data was provided than the defined column width, or the column contained more data than the output buffer could hold.

Error ID: 4, Severity: WARNING, Facility: DBCAPERR

Data type conflict.

The data type of a bound statement parameter or literal conflicted with the corresponding table column.

Error ID: 13, Severity: ERROR, Facility: DBCAPERR

Date/Time/Timestamp literal too long.

Error ID: 22, Severity: ERROR, Facility: FPSRVERR

Date/Time overflow.

A date/time parameter or literal in an SQL statement overflowed its allowed range.

Error ID: 14, Severity: ERROR, Facility: DBCAPERR

Date/Time overflow.

A date/time parameter or literal in an SQL statement overflowed its allowed range.

Error ID: 70, Severity: WARNING, Facility: DBCAPERR

Date or Time data truncated entering or leaving the database.

Error ID: 51, Severity: WARNING, Facility: DBCAPERR

Datetime field overflow.

The data for the column was not a valid date, time, or timestamp value.

Error ID: 30, Severity: ERROR, Facility: OIOERR

Disconnection failed because a COMMIT or ROLLBACK was in progress.

The disconnection failed because a COMMIT or ROLLBACK was in progress.

Error ID: 19, Severity: ERROR, Facility: DBCAPERR

Divide by zero error.

A divide-by-zero exception occurred during the execution of an SQL statement.

Error ID: 15, Severity: ERROR, Facility: DBCAPERR

Duplicate cursor name.

A cursor name assignment failed because the name already refers to another cursor.

Error ID: 22, Severity: ERROR, Facility: DBCAPERR

Empty string found in date/time column.

An empty string was found in a column defined by UniVerse SQL as SQL_DATE or SQL_TIME.

Error ID: 68, Severity: WARNING, Facility: DBCAPERR

End of data encountered.

The database indicated that the end of the data associated with this request has been reached.

Error ID: 10, Severity: INFORMATION, Facility: FPSRVERR

Error checking "STRING": STRING

Some problem occurred when UniVerse ODBC tried to check the status of a file. Check the external error number against your operating system's documentation to determine the cause.

Error ID: 6, Severity: ERROR, Facility: OCLIERR

Error fetching BASIC result set: [STRING] STRING

UniVerse ODBC encountered an error fetching results from one of the cataloged BASIC helper programs it uses to obtain UniVerse data dictionary information.

Error ID: 57, Severity: ERROR, Facility: FPSRVERR

Error opening "STRING": STRING

Some problem occurred when UniVerse ODBC tried to open a file, as reported by the stream. Check the external error number against your operating system's documentation to determine the cause.

Error ID: 2, Severity: ERROR, Facility: OCLIERR

Error writing "STRING": STRING

Some problem occurred when UniVerse ODBC tried writing to a file, as reported by the stream. This error usually occurs when the system is running out of disk space. Check the external error number against your operating system documentation to determine the cause.

Error ID: 3, Severity: ERROR, Facility: OCLIERR

Escape character not length 1.

The LIKE operator escape character must be a single character.

Error ID: 38, Severity: ERROR, Facility: FPSRVERR

Failure reading data from file.

A premature end-of-file or some other problem occurred while an application was reading streamed data from a file. Your file may have been truncated because of the lack of disk space, or somehow corrupted. Reconcile the external error number against your operating system's error number list for more details.

Error ID: 1, Severity: FATAL, Facility: XIOERR

Failure writing data to file.

Some problem occurred while an application was writing streamed data to a file. Your file may have been truncated because of the lack of disk space, or your disk may be corrupted. Reconcile the external error number against your operating system's error number list for more details.

Error ID: 2, Severity: FATAL, Facility: XIOERR

File write error, disk probably full.

A client was unable to write data to a disk file. Check free space on your disk.

Error ID: 19, Severity: ERROR, Facility: OIOERR

Floating-point value was truncated to integer.

A floating-point value was truncated during conversion to integer.

Error ID: 43, Severity: WARNING, Facility: DATUMERR

Identifier too big.

An identifier component (qualifier, owner name, table name, or column name) is longer than 128 characters.

Error ID: 27, Severity: ERROR, Facility: FPSRVERR

Incorrectly formatted date/time.

You typed a date/time value that could not be interpreted. Use the ANSI-standard *YYYY-MM-DD HH:MM:SS HH:MM* format. Note, however, that you cannot include any elements that are outside the precision of the date/time column (that is, you cannot include the time if the column is defined as date-only). Also note that the time zone is optional.

Error ID: 13, Severity: ERROR, Facility: DATUMERR

Incorrectly formatted integer.

You typed a value that could not be interpreted as an integer. Integers consist only of digits, with an optional + or – sign.

Error ID: 22, Severity: ERROR, Facility: DATUMERR

Incorrectly formatted interval.

You typed an interval value that could not be interpreted. Use the ANSI-standard *YYYY-MM* or *DD HH:MM:SS* format. Note, however, that you cannot include any elements that are outside the precision of the interval column.

Error ID: 20, Severity: ERROR, Facility: DATUMERR

Incorrectly formatted number.

You typed a value that could not be interpreted as a number. Numbers consist of an optional plus (+) or minus (–) sign, integer part, decimal point, fractional part, and exponent (including “e” for exponent, optional sign, and integer exponent value). For example, 1.2 and –3.4e56

Error ID: 23, Severity: ERROR, Facility: DATUMERR

Index names must be unqualified in CREATE INDEX and qualified in DROP INDEX.

In UniVerse SQL and UniVerse ODBC SQL, index names must be unqualified in CREATE INDEX statements and qualified in DROP INDEX statements.

Error ID: 61, Severity: ERROR, Facility: FPSRVERR

Input/output parameter bound to non-procedure statement.

Input/output parameters provide only output values when used with procedure calls. Other statements treat I/O parameters as input-only.

Error ID: 45, Severity: WARNING, Facility: DBCAPERR

Internal error. Please report details to your UV/ODBC Support contact: Assorted Internal Errors.

You have encountered an unexpected internal error. Record what you were doing and the complete error information (such as the Error ID, Extern ID, Severity, and Facility). Then report this information to IBM Customer Support.

Invalid Cursor State.

A cursor unprepared by an autocommit caused by a previously executed statement.

Error ID: 8, Severity: ERROR, Facility: OCLIERR

Invalid Date literal.

A date literal is in an invalid format. If the ODBC data source is configured with **Enforce ODBC Dates/Times** checked, all four digits of year and two digits of month and day must be specified.

Error ID: 23, Severity: ERROR, Facility: FPSRVERR

Invalid STRING argument to STRING

An invalid argument was passed to a request.

Error ID: 3, Severity: ERROR, Facility: FPSRVERR

Invalid Time literal.

A time literal is in an invalid format. If the ODBC data source is configured with **Enforce ODBC Dates/Times** checked, both digits of hour, minute, and second must be specified.

Error ID: 39, Severity: ERROR, Facility: FPSRVERR

Invalid Timestamp literal.

A timestamp literal is in an invalid format. If the ODBC data source is configured with **Enforce ODBC Dates/Times** checked, all four digits of year and two digits of month, day, hour, minute, and second must be specified.

Error ID: 40, Severity: ERROR, Facility: FPSRVERR

Invalid connection or statement option value.

The client tried to set a connection or statement option to an invalid value.

Error ID: 43, Severity: ERROR, Facility: DBCAPERR

Invalid cursor position.

Error ID: 40, Severity: ERROR, Facility: DBCAPERR

Invalid datatype.

Error ID: 35, Severity: ERROR, Facility: FPSRVERR

Invalid entry in character mapping file STRING line NUMBER.

The indicated line in the character mapping file contains an error. Each entry line in a character map file has the following format: *local* = *remote*. *local* and *remote* are 8-bit character code values in the range 1–255, and the codes for ASCII a–z, A–Z, 0–9, and !%*.<=>_ cannot be mapped. Check the indicated line in the indicated file for extraneous characters, invalid values, or conflicts with other lines. The system has ignored this line in setting up the character map.

Error ID: 11, Severity: INFORMATION, Facility: XIOERR

Invalid monetary unit STRING

The monetary unit specified is not a recognized monetary unit. Consult documentation on the money data type for allowable monetary units.

Error ID: 39, Severity: ERROR, Facility: DATUMERR

Invalid number.

A string is not in a recognizable numeric format and cannot be converted to a number.

Error ID: 17, Severity: ERROR, Facility: FPSRVERR

Invalid or unrecognized character.

Error ID: 24, Severity: ERROR, Facility: FPSRVERR

Invalid procedure created.

The procedure was created but errors were encountered. Errors could include invalid parameters, compilation errors, and so on.

Error ID: 48, Severity: WARNING, Facility: FPSRVERR

Invalid string or buffer length.

Error ID: 60, Severity: ERROR, Facility: DBCAPERR

Licensing Error: MESSAGE.

This error occurred while trying to establish a license for UniVerse ODBC. For more information about UniVerse licensing, see *Administering UniVerse* or contact IBM Customer Support.

Error ID: 75, Severity: ERROR, Facility: DBCAPERR

Licensing error: STRING

This error occurred while trying to establish a license for UniVerse ODBC. For more information regarding UniVerse licensing, see *Administering UniVerse* or contact IBM Customer Support.

Error: 75, Severity: ERROR, Facility: DBCAPERR

Line NUMBER, column NUMBER: STRING

Error ID 33 should not appear; message text was used when constructing other errors.

Error ID: 33, Severity: INFORMATION, Facility: FPSRVERR

Line NUMBER, column NUMBER (around "STRING"): STRING

Error ID 34 should not appear; message text was used when constructing other errors.

Error ID: 34, Severity: INFORMATION, Facility: FPSRVERR

No cursor of the specified name.

A reference to a cursor by name failed because there is no cursor with the specified name.

Error ID: 20, Severity: ERROR, Facility: DBCAPERR

No data to read from file or pipe.

A nonblocking read on a pipe or stream found no data available.

Error ID: 12, Severity: ERROR, Facility: OIOERR

No more database cursors available.

All the available database cursors are in use on this connection and none is available. Close one or more other cursors and retry the operation.

Error ID: 4, Severity: ERROR, Facility: FPSRVERR

No more result sets available.

All the result sets for this request have already been returned. There are no more result sets available.

Error ID: 8, Severity: INFORMATION, Facility: FPSRVERR

No more Windows timers left.

All the available Windows timers are in use on this serial connection. Close other Windows applications or exit and reenter Windows.

Error ID: 113, Severity: SEVERE, Facility: LINKERR

Number too big.

A numeric literal is longer than 254 characters.

Error ID: 28, Severity: ERROR, Facility: FPSRVERR

Numeric conversion or arithmetic expression error fetching column.

A numeric conversion or arithmetic expression error occurred while an application was fetching a value from a column. A null value was returned.

Error ID: 64, Severity: ERROR, Facility: DBCAPERR

Numeric data truncated entering or leaving the database.

Numeric data was truncated while entering or leaving the database.

Error ID: 50, Severity: WARNING, Facility: DBCAPERR

Numeric value out of range.

The supplied buffer is too small to hold the integer part of the requested numeric value. Provide a larger character buffer or a buffer of a numeric type with a larger range.

Error ID: 29, Severity: ERROR, Facility: OIOERR

Object input data format error: versions incompatible.

Your software tried to read object data that was created by a more recent version of the product. The data in question was probably a saved query file.

Error ID: 22, Severity: ERROR, Facility: OIOERR

ODBC-compliance syntax error.

A non-ODBC construct has been encountered at the indicated line and column number of the SQL statement. Verify that the SQL statement follows proper ODBC SQL syntax.

Error ID: 51, Severity: ERROR, Facility: FPSRVERR

ODBC escape clause not properly terminated.

Error ID: 21, Severity: ERROR, Facility: FPSRVERR

Operation canceled.

Error ID: 44, Severity: ERROR, Facility: DBCAPERR

Operation has been canceled.

The request was interrupted and cancelled. Results from the cancelled request may not be reliable.

Error ID: 49, Severity: ERROR, Facility: FPSRVERR

Out of memory.

The UniVerse ODBC driver has run out memory. Reduce the size of the result set you request, close other database connections with large query result sets, or close down other Windows applications that are consuming memory, then retry the operation. If this message persists, you may need to shut down Windows and restart it or reboot your PC. If this message occurs frequently, you need more memory on your PC or more virtual memory on your data source.

Error ID: 1, Severity: FATAL, Facility: MEMERR

Out of result memory in the client.

The result set of a query will not fit in available client memory. Reduce the size of the result set you request, close other database connections with large query result sets, or close down other Windows applications that are consuming memory, then retry the operation. If this message persists, you may need to shut down Windows and restart it or reboot your PC. If this message occurs frequently, you need more memory on your PC.

Error ID: 2, Severity: SEVERE, Facility: LINKERR

Out of sequence request.

The request could not be performed because prerequisite requests have not yet been made.

Error ID: 47, Severity: ERROR, Facility: FPSRVERR

Overflow converting a value to a numeric type.

A text- or number-to-number conversion failed because the number was too large to be represented in the specified format.

Error ID: 24, Severity: ERROR, Facility: DATUMERR

Path truncated.

You typed a value that was longer than a valid path plus filename. The value was truncated to the maximum length defined for the column. This message is informational only, and requires no further action.

Error ID: 30, Severity: WARNING, Facility: DATUMERR

Premature EOF reading file.

End-of-file was reached while an application was prematurely reading data from a file.

Error ID: 13, Severity: ERROR, Facility: OIOERR

Read from invalid file.

An attempt was made to read from an invalid file descriptor.

Error ID: 9, Severity: ERROR, Facility: OIOERR

Read permission denied.

An attempt to read a file failed because you have no access rights to the file.

Error ID: 10, Severity: ERROR, Facility: OIOERR

Received unexpected data: STRING

An event on the data source system caused unexpected data to be sent over the network. Usually this indicates that the server process has terminated, and the unexpected data is a message indicating what happened. Try again.

This error can also happen if the shell configuration scripts on your UNIX host echo data to the terminal. In this case, you must modify the scripts to move or remove the terminal-accessing commands.

If this problem persists, contact IBM Customer Support.

Error ID: 7, Severity: SEVERE, Facility: LINKERR

Receive timeout, closing connection.

The UniVerse ODBC driver timed out after waiting for data from the host server. The host system is either very slow or has failed, or the host server program has failed. Contact your system administrator.

Error ID: 16, Severity: FATAL, Facility: OIOERR

Restricted data type attribute violation.

The data value cannot be converted to the C data type specified by the argument *fCType*.

Error ID: 27, Severity: ERROR, Facility: OIOERR

Serialization failure.

The transaction to which the prepared statement belonged was terminated to prevent deadlock.

Error ID: 39, Severity: ERROR, Facility: DBCAPERR

Server received an increment of input character data longer than the defined chunk length.

The UniVerse ODBC driver received an increment of input character data longer than the defined chunk length. The data was right-truncated.

Error ID: 11, Severity: WARNING, Facility: DBCAPERR

SQL numeric value out of range.

An SQL statement contained a numeric parameter or literal that was out of range for its context.

Error ID: 12, Severity: ERROR, Facility: DBCAPERR

SQL numeric value out of range.

An SQL statement contained a numeric parameter or literal that was out of range for its context.

Error ID: 69, Severity: WARNING, Facility: DBCAPERR

SQL REVOKE failed because of insufficient privilege.

The user had insufficient privilege to revoke a privilege for another user.

Error ID: 5, Severity: ERROR, Facility: DBCAPERR

SQL statement contained ambiguous column reference.

An SQL statement used a column name that had more than one possible referent. The statement probably contained a join in which two tables each had a column with that name.

Error ID: 73, Severity: ERROR, Facility: DBCAPERR

SQL statement referenced a nonexistent column in a table or view.

Error ID: 31, Severity: ERROR, Facility: DBCAPERR

SQL statement referenced a nonexistent index.

Error ID: 29, Severity: ERROR, Facility: DBCAPERR

SQL statement referenced a nonexistent table or view.

Error ID: 27, Severity: ERROR, Facility: DBCAPERR

SQL statement used undefined table name.

An SQL statement used *tablename.columnname* or *tablename.** but did not correctly use *tablename* elsewhere in the statement—for example, in the FROM clause.

Error ID: 74, Severity: ERROR, Facility: DBCAPERR

SQL Syntax error.

The database reported an SQL syntax error.

Error ID: 21, Severity: ERROR, Facility: DBCAPERR

Stack overflow (lexer).

The SQL statement contains too many levels of nested escape sequences.

Error ID: 41, Severity: ERROR, Facility: FPSRVERR

String data/length mismatch.

Less character data was provided than was promised for filling in an input parameter.

Error ID: 16, Severity: ERROR, Facility: DBCAPERR

STRING is illegal in ODBC SELECT...FOR UPDATE statements.

In the ODBC SQL grammar, only a limited form of the SELECT statement can be used with FOR UPDATE.

Error ID: 64, Severity: ERROR, Facility: FPSRVERR

STRING is not a valid UniVerse account or schema.

The UniVerse account is not valid. If an account pathname was specified, verify that the directory exists and is a UniVerse account. If an account name was specified, verify that the UV.ACCOUNT entry is correct. If an SQL schema name was specified, verify that the UV_SCHEMA entry is correct.

Error ID: 76, Severity: ERROR, Facility: DBCAPERR

Syntax error.

A syntax error has been discovered at the indicated line and column number of the SQL statement. Verify that the SQL statement follows proper ODBC SQL syntax.

Error ID: 29, Severity: ERROR, Facility: FPSRVERR

Table name or correlation name conflicts with UniVerse ODBC's correlation name cookie <STRING>, so cannot be used.

UniVerse ODBC sometimes generates correlation names of the form <correlation name cookie><*n*>, where *n* is a nonnegative integer. You cannot use a table or correlation name that also has this form.

Error ID: 60, Severity: ERROR, Facility: FPSRVERR

TCP driver software not loaded.

The TCP/IP driver is not running, or the client TCP/IP driver software has not been loaded. Check to see if the networking software is installed properly.

Error ID: 111, Severity: SEVERE, Facility: LINKERR

Text truncated.

You typed a value that was longer than the space provided by the database for values in this column. The value was truncated to the maximum length defined for the column. This message is informational only, and requires no further action.

Error ID: 10, Severity: WARNING, Facility: DATUMERR

Text truncated to 63,950 characters.

Because the Text class is currently implemented using far pointers (instead of huge pointers), it is not possible to create a Text class containing more than 63,950 characters. If you need to store more than 63,950 characters of textual information, you might want to consider storing the text as binary data (using the Binary class).

Error ID: 44, Severity: WARNING, Facility: DATUMERR

The database reported an error when attempting to access this data item.

Error ID: 54, Severity: ERROR, Facility: DBCAPERR

The minutes and seconds parts of this date/time have been set to 0.

You typed a date and hours value for a column that is defined to include both a date and a complete time. The unspecified time portions have been set to 0. This message is informational only, and requires no further action.

Error ID: 18, Severity: WARNING, Facility: DATUMERR

The name STRING violates ODBC rules for names.

The ODBC 2.0 standard requires names of database objects to begin with a letter, and to contain only letters, digits, and the underscore character.

Error ID: 53, Severity: ERROR, Facility: FPSRVERR

The number did not fit within the specified precision; it was rounded off.

You typed a value that has more significant digits than the precision specified for its column. The number you typed was rounded off to fit within the column's defined precision. This message is informational only, and requires no further action.

Error ID: 27, Severity: WARNING, Facility: DATUMERR

The number is too large; it was rounded to infinity.

You typed a value that was larger than the precision specified for its column. Because the column supports infinite values, the number you typed was converted to infinity. This message is informational only, and requires no further action.

Error ID: 25, Severity: WARNING, Facility: DATUMERR

The operation modified or deleted all rows in the table.

An UPDATE or DELETE statement affected every row in the table, possibly because it had no WHERE clause.

Error ID: 34, Severity: WARNING, Facility: DBCAPERR

The requested operation was inapplicable to the specified cursor.

The requested operation was invalid on the specified cursor. Examples: UPDATE was positioned before FETCH; column attributes were on a non-SELECT statement.

Error ID: 18, Severity: ERROR, Facility: DBCAPERR

The requested value is not available.

Error ID: 46, Severity: ERROR, Facility: FPSRVERR

The Schema (UVSCHEMA) defining the catalog could not be accessed.

Error ID: 57, Severity: ERROR, Facility: DBCAPERR

The Schema defined by the client does not exist or could not be accessed.

Error ID: 55, Severity: ERROR, Facility: DBCAPERR

The search pattern argument STRING contains an escape character which is not itself escaped and which is not followed by %, _, or the escape character.

UniVerse ODBC's interpretation of ODBC is that in search pattern arguments to catalog functions, every escape character that is not itself escaped must be followed by %, _, or the escape character.

Error ID: 52, Severity: ERROR, Facility: FPSRVERR

The seconds part of this date/time has been set to 0.

You typed a date and hours/minutes value for a column that is defined to include both a date and a complete time. The unspecified time portions have been set to 0. This message is informational only, and requires no further action.

Error ID: 19, Severity: WARNING, Facility: DATUMERR

The server does not support the requested function.

Error ID: 25, Severity: ERROR, Facility: DBCAPERR

The Table defined by the client does not exist or could not be accessed.

Error ID: 56, Severity: ERROR, Facility: DBCAPERR

The table expansion/explosion was halted at the row limit of NUMBER rows.

Expansion/explosion of the table would generate more than the specified number of rows. The expanded/exploded table was limited to this number of rows and does not represent all the underlying data.

Error ID: 38, Severity: INFORMATION, Facility: DATUMERR

The Table of Columns (UVCOLUMNS) in the Catalog could not be accessed.

Error ID: 59, Severity: ERROR, Facility: DBCAPERR

The Table of Tables (UVTABLES) in the Catalog could not be accessed.

Error ID: 58, Severity: ERROR, Facility: DBCAPERR

The time part of this date/time has been set to midnight.

You typed a date value for a column that is defined to include both a date and a time. The time portion has been set to 0, which represents midnight. This message is informational only, and requires no further action.

Error ID: 17, Severity: WARNING, Facility: DATUMERR

The transaction isolation level cannot be changed during an open transaction.

The transaction isolation level cannot be changed during an open transaction. The current transaction must be committed or rolled back before the transaction isolation level can be changed.

Error ID: 72, Severity: ERROR, Facility: DBCAPERR

The underlying database does not provide support for output parameters.

Error ID: 52, Severity: ERROR, Facility: DBCAPERR

The UniVerse ODBC driver or the underlying database do not provide support for the requested SQL statement.

The UniVerse ODBC driver or the underlying database does not provide support for the requested SQL statement.

Error ID: 49, Severity: ERROR, Facility: DBCAPERR

The UniVerse ODBC driver or the underlying database do not provide the requested feature.

Error ID: 33, Severity: ERROR, Facility: DBCAPERR

The year in this date must be AD.

You typed a date/time value whose year is BC (negative), but the date/time column can support only AD (positive) years.

Error ID: 15, Severity: ERROR, Facility: DATUMERR

This SQL statement was ignored.

This SQL statement is not supported by the underlying database. This SQL statement was not executed by the UniVerse ODBC driver. Success with information is returned instead of an error because some applications abort if an attempted execution of this type of SQL statement returns an error.

Error ID: 63, Severity: WARNING, Facility: DBCAPERR

Too many open sockets.

The connection could not be made because all available TCP/IP sockets are in use. This can happen if many TCP/IP service sessions (telnet, ftp) or many UniVerse ODBC connections are open. Close one or more telnet, ftp, or UniVerse ODBC connections. You may be able to configure your TCP/IP software package to provide more concurrent sockets. See your PC TCP/IP package documentation.

Error ID: 122, Severity: SEVERE, Facility: LINKERR

Too many sockets open.

The local PC limit on the number of concurrently open sockets has been reached. This can occur if there are several concurrent sessions active over the TCP/IP/Ethernet connection, including telnet, ftp, and NFS.

Exit one or more of the concurrent application sessions and try again. If this fails, reboot your PC and try again.

Error ID: 106, Severity: SEVERE, Facility: LINKERR

Transmit timeout, closing connection.

No acknowledgment has been received from the host/server for the last message sent. The connection has been closed. Retry the connection.

Error ID: 120, Severity: SEVERE, Facility: LINKERR

UCI Error. Func: SQL Connect(); State: IM002; UniVerse code: 0; Msg: [IBM][SQL Client]The data source is not in configuration file.

Check the uci.config file to verify that the data source entry exists.

Error ID: 46, Severity: ERROR, Facility: DBCAPERR

UCI Error. Func: STRING; State: STRING; UniVerse code: NUMBER; Msg: STRING

A UCI error was encountered. See *UniVerse Call Interface Guide* to learn more about this error.

Error ID: 46, Severity: ERROR, Facility: DBCAPERR

UCI Warning Func: STRING; State: STRING; UniVerse code: NUMBER; Msg: STRING

A UCI warning was encountered. See *UniVerse Call Interface Guide* to learn more about this warning.

Error ID: 47, Severity: WARNING, Facility: DBCAPERR

Unable to determine the current schema for this connection.

Either no schema has been created for this account or the current schema could not be determined for this account from the account information provided in the Account field in the Database server/Authorization Information section of the configuration file entry.

Error ID: 53, Severity: ERROR, Facility: DBCAPERR

Uni RPC daemon is not running. Startup this daemon from the UniVerse System Administration menu before attempting to use UV/ODBC.

The UniRPC daemon must be running in order for UniVerse ODBC to establish a connection with UniVerse. The UniRPC daemon can be started from the UniVerse System Administration menu in the UV account. The UniRPC service can be started from the Windows NT Control Panel, the UniVerse Control Panel, or from an MS-DOS window.

Error ID: 71, Severity: ERROR, Facility: DBCAPERR

UniVerse SQL does not support UNION with FOR UPDATE.

Error ID: 65, Severity: ERROR, Facility: FPSRVERR

Error ID: 10, Severity: FATAL, Facility: XIOERR

Unknown data type.

An unknown ODBC data type has been passed.

Error ID: 2, Severity: ERROR, Facility: FPSRVERR

Unterminated quoted string.

A quoted string is missing a closing single or double quotation mark.

Error ID: 18, Severity: ERROR, Facility: FPSRVERR

User does not have permission to execute this SQL statement.

Error ID: 23, Severity: ERROR, Facility: DBCAPERR

***UV/ODBC BASIC program STRING failed to execute: [STRING]
STRING***

UniVerse ODBC's attempt to execute a cataloged BASIC helper program failed for the reason mentioned. This could indicate a file access problem or an invalid UniVerse ODBC installation.

Error ID: 55, Severity: ERROR, Facility: FPSRVERR

UV/ODBC BASIC program STRING reported error: STRING

A UniVerse ODBC cataloged BASIC program encountered this error while collecting information about UniVerse tables or files.

Error ID: 56, Severity: WARNING, Facility: FPSRVERR

Note: This error message actually describes a warning from a BASIC program, not an error.



UV/ODBC BASIC program STRING reported error: STRING

A cataloged BASIC program encountered this error while collecting information about UniVerse tables or files.

Error ID: 59, Severity: ERROR, Facility: FPSRVERR

UV/ODBC cannot describe the result set of a UniVerse procedure which has not been executed.

You requested information about the result set of a UniVerse procedure which has been prepared, but not yet executed. This information cannot be determined until you actually execute the procedure.

Error ID: 63, Severity: ERROR, Facility: FPSRVERR

UV/ODBC dictionary cannot prepare SQL statement [STRING].

UniVerse ODBC extracts information about columns of database tables from this database by preparing an SQL statement for execution. The database rejected the SQL statement, probably because access privileges were inadequate.

Error ID: 58, Severity: WARNING, Facility: FPSRVERR

UV/ODBC's file access control prevents writing to this file.

The user attempted to write to a file whose accessibility was specified in the HS_FILE_ACCESS file as READ or NONE.

Error ID: 54, Severity: ERROR, Facility: FPSRVERR

Write to invalid file.

An attempt was made to write to an invalid file descriptor.

Error ID: 17, Severity: ERROR, Facility: OIOERR

Write to invalid pipe.

An attempt was made to write to a pipe not open for writing.

Error ID: 18, Severity: ERROR, Facility: OIOERR

Wrong number of input parameters to SQL statement.

The client application provided more, or fewer, parameter values than there were place holders in the SQL statement.

Error ID: 41, Severity: ERROR, Facility: DBCAPERR

ODBC Environment Variables

UniVerse ODBC provides features you can configure through environment variables.

On Windows platforms, environment variables are set in the Windows Registry under the following key:

HKEY_LOCAL_MACHINE\IBM\UniVerse ODBC Driver

To view or change them, use a registry editor such as Microsoft Registry Editor (*regedit.exe*).

The following table lists the UniVerse ODBC environment variables on Windows platforms.

Environment Variable	Default	Controls...
HS_NAME_MAPPING	YES	Whether UniVerse ODBC automatically maps UniVerse schema, table, view, file, column, and field names to conform with ODBC naming restrictions.
HS_RETRY_AS_UV_SQL	NO	Whether UniVerse ODBC passes through statements it cannot interpret as ODBC SQL. For more information, see SQL-Related Connection Options .
HS_USE_FILEINFO	YES	Whether UniVerse ODBC uses its file information cache to construct the list of accessible ODBC tables. This variable is used only with 3.0.x versions of the UniVerse ODBC driver; later versions of the driver ignore it and use the client connection option FastConnect/ Refresh OTL on Connect instead.
UVFPDELAY	0	A delay in the UniVerse ODBC connection sequence, measured in seconds.

UniVerse ODBC Environment Variables (Windows Platforms)

Driver Process Log Files

D

The UniVerse ODBC driver maintains a log file called uvodbc.log to monitor the progress of UniVerse ODBC and provide administrative feedback on the use of UniVerse ODBC. These log files are also useful for diagnosing problems.

For each connection, these log files contain information about startup and connecting, followed by information about each client request paired with the UniVerse database server's response, in sequence. The log can also include messages generated by certain UniVerse database server errors.

Driver Process Log Filenames

Log filenames have the following format:

uvodbc.log

Saving and Examining the Log File

Usually logs are temporary and are automatically deleted following a client disconnect request and normal database server exit. However, if an error occurs, you may want to retain the log file to track down problems. To do so, complete the following steps:

1. Access the Microsoft ODBC Data Source Administrator tool by choosing **Start -> Programs -> IBM U2 -> UniVerse ODBC Driver -> ODBC Admins**.

The **ODBC Data Source Administrator** dialog box appears.

2. Select the **User DSN** or **System DSN** tab, and then select the appropriate data source name (DSN).
3. Click the **Configure** button.

The **Universe ODBC Data Source Setup** dialog box appears.

4. Select the **Create Debug Log** check box.
5. Click **OK**.

Locating the UniVerse ODBC Driver Log File

The uvodbc.log file is created in the current working directory.

Sample Accounts

E

This appendix provides details about the sample UniVerse accounts that the UniVerse ODBC installation script creates. It describes where to find them and the information they contain.

Locating the Sample UniVerse Accounts

Each UniVerse database server has two sample accounts: HS.SALES and HS.SERVICE. When you install UniVerse, these sample accounts are created in the UV account directory, so you can use them to learn how to use UniVerse ODBC before working with your own data. The files for these accounts are located in two directories called HS.SALES and HS.SERVICE.

If you defined HS.SALES and HS.SERVICE as ODBC data sources using UCI Config Editor, they should be accessible to client products and sample applications.

Database Definitions

The HS.SALES and HS.SERVICE accounts each contain several files. The following sections list the files contained in each account, the columns found in each file, and the relationships among the files.

HS.SALES Account

The HS.SALES account contains information about copiers and fax machines sold by a fictitious manufacturer. It contains information about sales prospects, customers who bought machines, details of their purchases, and the products purchased. This information is typically used by the Sales department.

HS.SALES comprises three files: CUSTOMER, PRODUCTS, and STATES.

CUSTOMER	PRODUCTS	STATES
CUSTID	PRODID	CODE
SAL	LIST	NAME
FNAME	DESCRIPTION	
LNAME		
COMPANY		
ADDR1		
ADDR2		
CITY		
STATE		
ZIP		
PHONE		
PRODID		
SER_NUM		
PRICE		

HS.SALES Files

CUSTOMER	PRODUCTS	STATES
BUY_DATE		
PAID_DATE		
SVC_PRICE		
SVC_START		
SVC_END		
SVC_PAID_DATE		

HS.SALES Files (Continued)

HS.SERVICE Account

The HS.SERVICE account contains information about customer calls, commonly reported problems, and products under maintenance. This information is typically used by the Service department.

HS.SERVICE comprises three files: CALLS, PROBLEMS, and PRODS.

CALLS	PROBLEMS	PRODS
CALL_ID	PROB_NUM	PRODUCT
CUST_NUM	PRODUCT	VERSION
F_NAME	VERSION	DESCRIPTION
L_NAME	PROB_DESCRIPTION	FIRST_SHIP
PHONE	STATUS	
PROB_NUM	FIX	
PRODUCT	NOTES	
VERSION	OCCURRENCES	
SERIAL		
CALLED_ON		

HS.SERVICE Files

CALLS	PROBLEMS	PRODS
CALL_DATE		
CALL_TIME		
CALL_MINUTES		
PROBLEM		
RESOLUTION		
HS.SERVICE Files (Continued)		

Sample Data

The following files show the sample data contained in the HS.SALES and HS.SERVICE accounts.



***Note:** If you update any of these files, the actual data you see can differ from these samples.*

CUSTOMER File – HS.SALES Account

CUSTID	SAL	FNAME	LNAME	COMPANY	ADDR1	ADDR2	CITY	STATE	ZIP	PHONE
1	Mr.	Samuel	Smith	Better Beer, Inc.	10 Commercial St		Concord	NH	02131	(603) 555-3212
2	Ms.	Diana	Morris	Fast Copy Center	431 Third Ave.		Waltham	MA	01133	(617) 555-9823
3	Mr.	David	Argonne	Fast Copy Center	75 Great Road		Bedford	MA	01182	(617) 555-3468
4	Ms.	Jill	Kahn	Fast Copy Center	12 School St		Boston	MA	01103	(617) 555-7396
5	Mr.	Kenneth	Williams	Ocean State Fish Company	837 Ocean Ave		Providence	RI	03171	(401) 555-6512
6	Ms.	Betty	Burke	Lightning Computer Corp.	400 Technology Path	MS 10-27	White River	VT	01644	(802) 555-9854
7	Dr.	Martha	Gill	Central Hospital	555 Main Street		Derry	NH	04429	(603) 555-5437
8	Mr.	Steven	Holland	Copies, Inc.	4325 Hill Road		Lowell	MA	01386	(508) 555-2365

CUSTOMER File – HS.SALES Account

CUSTID	SAL	FNAME	LNAME	COMPANY	ADDR1	ADDR2	CITY	STATE	ZIP	PHONE
9	Ms.	Nicole	Orlando	A1 Used Auto	820 Middlesex Turnpike		Burlington	MA	01173	
10	Dr.	Andrew	McCaig	HGT Dental Center	999 Hill Road		Brattle-boro	VT	03356	(802) 555-6534
11	Mr.	Skip	Lewis	Skip's Whale Watch	10 Dock Street		Plymouth	MA	01382	(508) 555-2368
12	Mrs.	Laurie	Patry	Rustic Printers	10 Rustic Trail		Littleton	MA	01142	(508) 555-9426

CUSTOMER File – HS.SALES Account (Continued)

CUST_ID	PRODID	SER_NUM	PRICE	BUY_DATE	PAID_DATE
1	M2000	501278	\$4200	1991-01-07	1991-01-28
2	C2000	600782	\$6600	1991-01-08	1991-02-05
2	M3000	700422	\$12000	1991-01-08	1991-02-05
2	S3000	101456	\$900	1991-01-22	1991-02-12
3	M2000	501310	\$4250	1991-01-08	1991-01-10
4	C3000	800311	\$16500	1991-01-09	1991-02-07
5	M1000	403485	\$1900	1991-01-14	1991-02-14
5	M1000	403723	\$1900	1991-02-15	1991-02-28
7	M2000	501233	\$4490	1991-01-20	
7	S2000	101212	\$990	1991-01-20	
8	M3000	700514	\$1200	1991-01-21	1991-02-21
8	S3000	201399	\$900	1991-01-21	1991-02-21

CUSTOMER File – HS.SALES Account (Continued)

CUST_ID	PRODID	SER_NUM	PRICE	BUY_DATE	PAID_DATE
10	M1000	203510	\$1990	1991-01-28	1991-02-14
10	M1000	203600	\$1900	1991-01-29	1991-02-28
10	C2000	600791	\$6500	1991-01-30	

CUSTOMER File – HS.SALES Account (Continued)

CUSTID	PRODID	SER_NUM	SVC_PRICE	SVC_START	SVC_END	SVC_PAID_DATE
1	M2000	501278	\$600	1991-01-13	1992-01-15	1991-01-28
2	C2000	600782	\$900	1991-01-13	1992-01-15	1991-02-05
3	M3000	700422	\$500	1991-01-13	1992-06-12	1991-02-05
4	S3000	101456	\$150	1991-01-13	1991-01-15	1991-02-12
5	M1000	403485	\$600	1991-01-13	1991-06-12	1991-02-14
6	C3000	800311	\$600	1991-01-13	1991-06-12	1992-02-07
9	M3000	700514	\$1000	1991-02-03	1992-02-05	1991-02-21
10	S3000	201399	\$150	1991-02-03	1992-02-05	1991-02-21
11	M1000	203600	\$400	1991-02-03	1992-02-05	1991-02-28
12	C2000	600791	\$950	1991-02-15	1992-02-19	
13	M1000	403723	\$600	1991-03-02	1991-08-07	1991-02-28

PRODUCTS File – HS.SALES Account

PRODID	LIST	DESCRIPTION
M1000	\$1,990	Low cost, entry level, light duty, monoc

PRODUCTS File – HS.SALES Account (Continued)

PRODID	LIST	DESCRIPTION
M2000	\$4,490	Moderate duty monochrome copier
C2000	\$6,890	Moderate duty, entry level, color copier
M3000	\$12,990	Heavy duty monochrome copier
C3000	\$17,990	Heavy duty color copier
S2000	\$990	Sorting attachment for M2000/C2000
S3000	\$1,990	Sorting attachment for M3000/C3000

STATES File – HS.SALES Account

CODE	NAME		CODE	NAME
AK	Alaska		MT	Montana
AL	Alabama		NC	North Carolina
AR	Arkansas		ND	North Dakota
AZ	Arizona		NE	Nebraska
CA	California		NH	New Hampshire
CO	Colorado		NJ	New Jersey
CT	Connecticut		NM	New Mexico
DE	Delaware		NV	Nevada
FL	Florida		NY	New York
GA	Georgia		OH	Ohio
HI	Hawaii		OK	Oklahoma
IA	Iowa		OR	Oregon
ID	Idaho		PA	Pennsylvania
IL	Illinois		RI	Rhode Island
IN	Indiana		SC	South Carolina
KS	Kansas		SD	South Dakota
KY	Kentucky		TN	Tennessee
LA	Louisiana		TX	Texas
MA	Massachusetts		UT	Utah
MD	Maryland		VA	Virginia
ME	Maine		VT	Vermont

STATES File – HS.SALES Account

CODE	NAME		CODE	NAME
MI	Michigan		WA	Washington
MN	Minnesota		WI	Wisconsin
MO	Missouri		WV	West Virginia
MS	Mississippi		WY	Wyoming

CALLS File – HS.SERVICE Account

CUST_NUM	F_NAME	L_NAME	PHONE	PROB_NUM	PRODUCT	VERSION	SERIAL	CALLED_ON	CALL_MINUTES	PROBLEM	RESOLUTION
2	Diana	Morris	(617) 555-9823	4	C2000	1D	600782	1991-02-14 13:45:01	2		Didn't read the manual
10	Andy	McCaig	(802) 555-6534		M1000	2	203600	1991-02-05 9:05:04	4	Copier doesn't work in the morning	Copier in 10 minute warm-up cycle; didn't read manual and understand the flashing 'wait' light
10	Andy	McCaig	(802) 555-6534		M1000	2	203600	1991-02-05 12:56:37	4	Lines down the middle of the page	Drum needs cleaning; service technician dispatched.
1	Sam	Smith	(603) 555-3212	5	M2000	1	501278	1991-03-02 9:32:45	3		

CALLS File – HS.SERVICE Account (Continued)

2	Diana	Morris	(617) 555- 9823	7	M3000	2	700422	1991- 02-18 14:58:10	5		
5	Ken	Williams	(401) 555- 6512	3	M1000	1	403485	1991- 02-20 15:30:02	2		
5	Ken	Williams	(401) 555- 6512	8	M1000	1A	403723	1991- 02-24 11:12:52			
8	Steve	Holland	(508) 555- 2365	7	C3000	2	700514	1991- 02-24 13:12:33		Stapler jams	Wasn't using proper kind of staples.

PROBLEMS File – HS.SERVICE Account

PROB_NUM	PRODUCT	VERSION	PROB_DESCRIPTION	STATUS	FIX	NOTES	OCCURRENCES
1	S200	1	Sorter jams when collating stapled copies	fixed	Cycle sorter power before every collated, stapled print job	Fixed in rev 1A	15
2	M1000	1	Toner light won't go out, even after adding more toner	fixed	Cycle power after adding toner	Fixed in rev 2	4
3	M1000	1	Frequent paper jams	fixed	Fan paper thor- oughly, don't use moist paper	Rev 2 paper tray is much better	27

PROBLEMS File – HS.SERVICE Account (Continued)

PROB_NUM	PRODUCT	VERSION	PROB_DESCRIPTION	STATUS	FIX	NOTES	OCCURRENCES
4	C2000	1	Smeared colors after changing toner cartridge	doc	Make 20 copies of color test sheet after changing toner	The manual now recommends this as standard procedure	25
5	M2000	1	Paper jam light indicates location 10, no paper jammed there	fixed	Paper is jammed between locations 9 and 10. Turn roller 9 several times to right and remove from location 10	Rev 1A shows both locations 9 and 10 for this kind of jam	3
6	C3000	1	Smeared colors after changing toner cartridge	doc	Make 20 copies of color test sheet after changing toner	The manual now recommends this as standard procedure	17
7	M3000	2	Frequent original jams when 2-sided copying large originals	pending	Feeder needs service. Should install ver. 2A feeder long-life upgrade.		0
8	M1000	2	Wrong paper size displayed when using 8.5x14 paper tray	doc	Use only rev2 paper trays	Explained in rev 3 manual, page 10.	4

PRODS File – HS.SERVICE Account

Product	Version	Description	First_Ship
M1000	2	Low cost, entry level, light duty, monochrome copier. Rated 1000 pages/month. Annual service recommended.	1991-01-31
M2000	1A	Moderate duty monochrome copier. Rated 10,000 pages/month. Bimonthly service recommended.	1991-02-14
C2000	1D	Moderate duty, entry level, color copier. Rated 10,000 pages/month. Bimonthly service recommended.	1991-02-14
M3000	2	Heavy duty monochrome copier. Rated 100,000 pages/month. Monthly service recommended.	1990-04-05
C3000	3A	Heavy duty color copier. Rated 100,000 pages/month. Monthly service recommended.	1990-05-10
S2000	2	Sorting attachment for M2000/C2000	1991-01-30
S3000	2	Sorting attachment for M3000/C3000	1991-01-30

Index

Symbols

&SELECTLISTS& file 4-14
 @ phrase 4-16, 4-27
 @ASSOC_KEY.mvname record 4-27
 @ASSOC_ROW column 4-27
 @EMPTY.NULL record 4-25
 @SELECT phrase 4-16, 4-27

A

A- and S-descriptors 4-17
 accessing
 accounts 4-3, 4-6–4-8
 columns 4-4, 4-16
 fields 4-4, 4-16
 files 4-3, 4-9–4-15
 schemas 4-3, 4-6–4-8
 System Administration menu 2-5
 tables 4-3, 4-9–4-15
 views 4-3, 4-9–4-15
 accounts
 activating UniVerse ODBC file
 access 2-3
 making accessible 4-3, 4-6–4-8
 sample
 definitions E-3
 HS.SALES E-3
 HS.SERVICE E-4
 ALTER TABLE statement A-4
 API ODBC conformance 1-6
 associating select lists 4-14
 association keys 4-27–4-28
 associations 4-27
 autocommit mode A-20

C

cache, file information 4-11
 CASCADE qualifier A-3
 character columns 4-23
 clients
 troubleshooting checklist 5-15
 column aliases A-12
 columns
 @ASSOC_ROW 4-27
 character 4-23
 making accessible 4-4, 4-16
 multivalued 4-5, 4-16
 commands
 HS.SCRUB 4-22
 communications
 isolating problems 5-4
 manual verification 5-10
 configuration files
 UCI 3-17
 configuration manager 1-4
 configuration parameters
 MAXFETCHBUFF 3-18, 4-24
 MAXFETCHCOLS 3-17, 4-24
 configurations
 defining 4-6
 configuring UCI for UniVerse
 ODBC 3-17
 conformance
 API 1-6
 ODBC 1-6
 connection problems 5-11
 conversion codes 4-26
 conversion errors 4-25
 core grammar
 CASCADE qualifier A-3
 LIKE USER keyword A-3

ODBC SQL A-3
 qualified index names A-3
 RESTRICT qualifier A-3
 table name qualifiers in DDL A-3
 CREATE INDEX statement A-3
 CREATE TABLE statement A-4

D

data
 conversion errors 4-25
 dirty 4-25
 length 4-4
 multivalued 4-5
 data source 1-5
 data types 4-4, 4-22–4-23
 specifying A-3
 SQL A-2
 supported A-2
 database transactions, UniVerse ODBC
 support of A-20
 databases
 sample E-2
 DATATYPE field 4-22, 4-24
 DATE
 in ALTER TABLE statement A-4
 in CREATE TABLE statement A-4
 defining configurations 4-6
 diagnosing connection problems 5-11
 diagnostic tool, ODBC 3-20
 dirty data 4-25
 Dr. DeeBee Spy 3-20, 5-11
 log file 3-21, 3-22
 monitoring configurations 3-20
 DRDEEBEE.LOG file 3-21, 3-22
 DROP INDEX statement A-3

E

empty strings 4-25
 empty-null mapping 4-25
 environment files 5-8
 environment variables
 HS_NAME_MAPPING C-2
 HS_RETRY_AS_UV_SQL A-18, C-2
 HS_USE_FILEINFO C-2
 PATH 5-8
 UVFPDELAY C-2

Windows NT Registry C-1
 error messages B-1
 errors, data conversion 4-25
 ESCAPE clause A-5
 executing UniVerse SQL A-17
 extended grammar
 date types A-4
 ESCAPE clause in LIKE
 predicate A-5
 ODBC SQL A-4
 outer joins A-4
 scalar functions A-6
 time types A-4
 UNION operator A-4
 extensions, SQL A-18

F

fields
 making accessible 4-4, 4-16
 multivalued 4-5, 4-16
 unassociated multivalued 4-27
 file access utility 4-9, 4-11–4-12
 file information cache 2-6, 4-11
 files
 &SELECTLISTS& 4-14
 DRDEEBEE.LOG 3-21, 3-22
 environment 5-8
 hosts 5-11
 HS.INSTALL.LOG 5-14
 HS_FILE_ACCESS 2-6, 4-3, 4-9, 4-10, 4-11, 5-14
 making accessible 4-3, 4-9–4-15
 permissions and privileges 4-10
 remote 4-6
 UCI configuration 3-17
 unirpcservices 3-19
 UniVerse, *see* UniVerse files
 UV.ACCOUNT 2-6
 first normal form 4-5, 4-17
 FORMAT field 4-24
 functions, scalar A-6

G

generic ODBC applications,
 interoperability with UniVerse
 procedures A-16
 grammar

core ODBC SQL A-3
 extended ODBC SQL A-4

H

hosts file 5-11
 HS.INSTALL.LOG file 5-14
 HS.ODBC.LISTS record 4-14
 HS.SALES account E-3
 CUSTOMER table E-6
 PRODUCTS table E-9
 STATES table E-9
 HS.SCRUB command 4-20, 4-22
 HS.SCRUB program 2-6
 HS.SERVICE account E-4
 CALLS table E-10
 PROBLEMS table E-11
 PRODS table E-13
 HS.SHOW.CONFIG program 2-5
 HS.UPDATE.FILEINFO
 command 4-3, 4-10
 HS.UPDATE.FILEINFO program 2-6
 HS_DEFAULT record 4-10
 HS_FILE_ACCESS file 2-6, 4-3, 4-9, 4-10, 4-11, 5-14
 HS_NAME_MAPPING environment
 variable C-2
 HS_RETRY_AS_UV_SQL
 environment variable A-18, C-2
 HS_USE_FILEINFO environment
 variable C-2

I

INSERT statements 4-28
 integrity modes A-20
 isolating problems 5-4

K

keys
 association 4-27–4-28
 primary 4-27
 stable 4-27
 unstable 4-27

L

left outer joins A-4

length
 of character columns 4-23
 of data 4-4
 LIKE keyword A-5
 LIKE USER keyword A-3
 limitations
 catalog information A-16
 default values with procedure
 parameters A-16
 functional A-11
 qualifiers A-15
 return values A-15
 of UniVerse procedures A-15
 log files
 description D-1
 examining D-3

M

manual mode A-20
 mapping
 empty-null 4-25
 name 4-29–4-30
 MAXFETCHBUFF parameter 3-18,
 4-24
 MAXFETCHCOLS parameter 3-17,
 4-24
 monitoring
 configurations
 Dr. DeeBee Spy 3-20
 server processes D-1
 multivalued
 columns 4-16
 data 4-5
 fields 4-16
 fields, unassociated 4-27

N

name mapping 4-29–4-30
 names
 mapping 4-29–4-30
 unqualified table 4-7
 native SQL syntax 4-11, 4-30, A-17
 null value 4-25

O

ODBC

conformance 1-6
 API 1-6
 diagnostic tool 3-20
 table types A-22
 ODBC applications, generic A-16
 ODBC clients
 left outer joins A-4
 limitations A-11
 ODBC file access utility 4-9, 4-11–
 4-12
 ODBC name mapping 4-29–4-30
 ODBC qualifiers 4-6–4-8
 outer joins A-4
 overview
 of UCI Config Editor tool 1-4
 of UniVerse ODBC client 1-4
 of UniVerse ODBC drivers 1-4
 of UniVerse ODBC server 1-3

P

parameters, procedures A-14
 PATH environment variable 5-8
 permissions 4-10
 Ping utility 5-10
 precision 4-23
 primary keys 4-27
 privileges 4-10
 problems
 isolating 5-4
 reporting 5-5
 procedures
 behavior of SQLRowCount
 function A-15
 interoperability with generic ODBC
 applications A-16
 limitations A-15
 catalog information A-16
 default values with procedure
 parameters A-16
 qualifiers A-15
 return values A-15
 SQLProcedureColumns
 function A-16
 SQLProcedures function A-16
 parameters A-14
 result sets A-14
 syntax A-13
 UniVerse A-13

Q

Q-pointers 4-6
 qualified index names in CREATE
 INDEX statement A-3
 qualifiers 4-6–4-8
 table name A-3

R

record IDs 4-27, 4-28
 registry, *see* Windows NT: Registry
 remote files 4-6
 RESTRICT qualifier A-3
 result sets A-14
 SQLColAttributes function A-14
 SQLDescribeCol function A-14
 SQLNumResultCols function A-14
 SQLPrepare function A-14
 retrying as UniVerse SQL A-18

S

sample accounts E-1
 scalar functions A-6
 schemas, making accessible 4-3, 4-6–
 4-8
 select lists, associating 4-14
 server
 integrity modes A-20
 monitoring D-1
 software
 checking version number 5-6
 transaction support A-20
 SICA 4-16, 4-23
 SM field 4-17
 SQL
 core ODBC grammar A-3
 extended ODBC grammar A-4
 native syntax 4-11, 4-30
 native syntax extension A-17
 processor A-17
 syntax A-17
 extensions A-18
 native A-17
 retrying as UniVerse SQL A-18
 UniVerse SQL A-17
 SQL data types, supported A-2

SQL_CURRENT_QUALIFIER
 option 4-7
 SQLColAttributes function A-14,
 A-16
 SQLDescribeCol function A-14, A-16
 SQLError function 3-20
 SQLExecute function 3-20, A-16
 SQLGetInfo function 3-22
 SQLNumResultCols function A-14,
 A-16
 SQLPrepare function 3-20, A-14,
 A-16
 SQLProcedureColumns function A-16
 SQLProcedures function A-16
 SQLRowCount function A-15
 SQLSetConnectOption function 4-7
 SQLTables function 4-17
 SQLTrace function 3-22
 SQLTYPE field 4-22, 4-24
 stable association keys 4-27
 syntax
 native SQL 4-11
 procedure calls A-13
 system
 obtaining information 5-6, 5-7
 System Administration menu
 accessing 2-5
 options 2-5

T

table names, unqualified 4-7
 table types A-22
 tables
 making accessible 4-3, 4-9–4-15
 unqualified names 4-7
 TCP/IP, verifying a connection 5-10
 telnet 5-11
 TIME
 in ALTER TABLE statement A-4
 in CREATE TABLE statement A-4
 transaction support A-20
 transactions
 DBMS A-20
 server support A-20
 UniVerse ODBC driver A-21
 troubleshooting
 client checklist 5-15
 client information 5-6

isolating problems 5-4
 server information 5-6
 UniVerse ODBC server
 checklist 5-14
 UNIX server checklist 5-13
 Windows NT server checklist 5-12

U

UCI
 configuration file 3-17
 configuring for UniVerse
 ODBC 3-17
 UCI Config Editor tool
 overview 1-4
 unassociated multivalued fields 4-27
 UNION operator A-4
 UniRPC daemon 5-13
 UniRPC service 5-12
unirpcservices file 3-19
 UniVerse 2-5
 checking version 5-6
 procedures A-13
 troubleshooting checklist 5-14
 UniRPC daemon 5-13
 UniRPC service 5-12
 UniVerse ODBC
 configuring UCI for 3-17
 UniVerse ODBC client, overview 1-4
 UniVerse ODBC Config 4-6
 UniVerse ODBC drivers 1-3
 overview 1-4
 transactions A-21
 UniVerse ODBC server
 overview 1-3
 upgrading 4-12
 UniVerse SQL
 executing A-17
 native syntax extension A-17
 UNIX
 troubleshooting checklist 5-13
 unqualified names in DROP INDEX
 statement A-3
 unqualified table names 4-7
 unstable association keys 4-27
 UPDATE statements 4-28
 updates, transaction support A-20
 upgrading UniVerse ODBC server
 software 4-12

UV.ACCOUNT file 2-6
 UVFPDELAY environment
 variable C-2

V

version numbers, checking 5-6
 views, making accessible 4-3, 4-9–
 4-15
 virtual fields 5-14

W

Windows NT
 checking version 5-6
 Registry C-1
 troubleshooting checklist 5-12