UniData

Developing UniData ODBC Applications
Notices

Edition
Publication date: June 2008
Book number: UDT-720-ODBC-1
Product version: UniData 7.2

Copyright

Trademarks
The following trademarks appear in this publication:

<table>
<thead>
<tr>
<th>Trademark</th>
<th>Trademark Owner</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rocket Software™</td>
<td>Rocket Software, Inc.</td>
</tr>
<tr>
<td>Dynamic Connect®</td>
<td>Rocket Software, Inc.</td>
</tr>
<tr>
<td>RedBack®</td>
<td>Rocket Software, Inc.</td>
</tr>
<tr>
<td>SystemBuilder™™</td>
<td>Rocket Software, Inc.</td>
</tr>
<tr>
<td>UniData®</td>
<td>Rocket Software, Inc.</td>
</tr>
<tr>
<td>UniVerse™</td>
<td>Rocket Software, Inc.</td>
</tr>
<tr>
<td>U2™</td>
<td>Rocket Software, Inc.</td>
</tr>
<tr>
<td>U2.NET™</td>
<td>Rocket Software, Inc.</td>
</tr>
<tr>
<td>U2 Web Development Environment™</td>
<td>Rocket Software, Inc.</td>
</tr>
<tr>
<td>wIntegrate®</td>
<td>Rocket Software, Inc.</td>
</tr>
<tr>
<td>Microsoft® .NET</td>
<td>Microsoft Corporation</td>
</tr>
<tr>
<td>Microsoft® Office Excel®, Outlook®, Word</td>
<td>Microsoft Corporation</td>
</tr>
<tr>
<td>Windows®</td>
<td>Microsoft Corporation</td>
</tr>
<tr>
<td>Windows® 7</td>
<td>Microsoft Corporation</td>
</tr>
<tr>
<td>Windows Vista®</td>
<td>Microsoft Corporation</td>
</tr>
<tr>
<td>Java™™ and all Java-based trademarks and logos</td>
<td>Sun Microsystems, Inc.</td>
</tr>
<tr>
<td>UNIX®</td>
<td>X/Open Company Limited</td>
</tr>
</tbody>
</table>
The above trademarks are property of the specified companies in the United States, other countries, or both. All other products or services mentioned in this document may be covered by the trademarks, service marks, or product names as designated by the companies who own or market them.

License agreement
This software and the associated documentation are proprietary and confidential to Rocket Software, Inc., are furnished under license, and may be used and copied only in accordance with the terms of such license and with the inclusion of the copyright notice. This software and any copies thereof may not be provided or otherwise made available to any other person. No title to or ownership of the software and associated documentation is hereby transferred. Any unauthorized use or reproduction of this software or documentation may be subject to civil or criminal liability. The information in the software and documentation is subject to change and should not be construed as a commitment by Rocket Software, Inc.

Restricted rights notice for license to the U.S. Government: Use, reproduction, or disclosure is subject to restrictions as stated in the "Rights in Technical Data-General" clause (alternate III), in FAR section 52.222-14. All title and ownership in this computer software remain with Rocket Software, Inc.

Note
This product may contain encryption technology. Many countries prohibit or restrict the use, import, or export of encryption technologies, and current use, import, and export regulations should be followed when exporting this product.

Please be aware: Any images or indications reflecting ownership or branding of the product(s) documented herein may or may not reflect the current legal ownership of the intellectual property rights associated with such product(s). All right and title to the product(s) documented herein belong solely to Rocket Software, Inc. and its subsidiaries, notwithstanding any notices (including screen captures) or any other indications to the contrary.

Contact information
Rocket Software
275 Grove Street Suite 3-410
Newton, MA 02466-2272
USA
Tel: (617) 614-4321 Fax: (617) 630-7100
Web Site: www.rocketsoftware.com
# Table of Contents

## Chapter 1  Introduction to UniData ODBC
- Accessing Data from the Desktop ........................................ 1-3
- UniData ODBC ..................................................................... 1-3
- UniOLEDB ........................................................................ 1-3
- Visual Schema Generator .................................................. 1-3
- UniData ODBC ..................................................................... 1-4
- Components of ODBC ....................................................... 1-4
- Components of UniData ODBC ........................................... 1-6
- Conformance Levels ........................................................... 1-7
- Definitions Used in this Manual ......................................... 1-9
- Application Program Interface (API) ................................. 1-9
- Client/server ...................................................................... 1-9
- Database ............................................................................ 1-9
- Nested RDBMS ................................................................. 1-9
- Open Database Connectivity (ODBC) .................................. 1-10
- Remote Execution (REX) .................................................. 1-10
- Remote Procedure Call (RPC) ........................................... 1-10
- Schema ............................................................................ 1-10
- Structured Query Language (SQL) ...................................... 1-10

## Chapter 2  Installing and Setting Up UniData ODBC
- Introduction ....................................................................... 2-2
- Before You Begin ............................................................... 2-3
- Requirements ..................................................................... 2-4
- Server Requirements ......................................................... 2-4
- PC Client Requirements .................................................... 2-4
- Setting Up the UCI Configuration File ............................... 2-5
- Verifying That UniRPC Is Running .................................... 2-7
- UCI Connection Timeout Configuration ............................. 2-8
- Running Concurrent UniData Versions ............................... 2-9
Table of Contents

Chapter 3  UniData ODBC Conformance Levels

Making UniData Accounts Accessible  2-10
Tracing Events  2-10
Presenting Data in ODBC–Accessible Format  2-12
Adding a UniData ODBC Data Source  2-13
Modifying a ODBC Data Source  2-16
Deleting a ODBC Data Source  2-18
Viewing ODBC Driver Information  2-19
Tracing Calls to ODBC Drivers  2-20
Viewing ODBC Core Component Information  2-22

API Conformance Levels  3-3
Core API  3-3
Level 1 API  3-3
Level 2 API  3-4
UniData ODBC Conformance Levels  3-4
UniData ODBC-Supported API Functions  3-5
Connecting to a Data Source  3-5
Processing an SQL Statement  3-8
Retrieving Results and Information about Results  3-10
Retrieving Catalog Information  3-11
Retrieving Data Source and Driver Information  3-12
Error Handling  3-12
SQL Conformance Levels  3-14
Minimum SQL Grammar  3-14
Core SQL Grammar  3-14
Extended SQL Grammar  3-14
UniData ODBC SQL Conformance  3-15
UniData ODBC-Supported SQL Capabilities  3-16
Data Types  3-16
Data Definition Language  3-16
Data Manipulation Language  3-17
Scalar Functions  3-18
# Introduction to UniData ODBC

Accessing Data from the Desktop ............................... 1-3
  UniData ODBC ................................................. 1-3
  UniOLEDB ..................................................... 1-3
  Visual Schema Generator ................................. 1-3

UniData ODBC ..................................................... 1-5
  Components of ODBC ........................................ 1-5
  Components of UniData ODBC ............................ 1-7
  Conformance Levels ..................................... 1-8

Definitions Used in this Manual ......................... 1-10
  Application Program Interface (API) .................... 1-10
  Client/server .............................................. 1-10
  Database .................................................... 1-10
  Nested RDBMS ............................................. 1-10
  Open Database Connectivity (ODBC) .................... 1-11
  Remote Execution (REX) ............................... 1-11
  Remote Procedure Call (RPC) .......................... 1-11
  Schema ..................................................... 1-11
  Structured Query Language (SQL) .................... 1-11
This manual describes how to set up UniData ODBC to allow ODBC-compliant applications access to data in UniData from a desktop environment. This manual also describes the ODBC functions UniData ODBC supports.

You can use this manual in conjunction with *Using VSG and the Schema API* to help use or develop client/server applications that access UniData. *Using VSG and the Schema API* describes the Windows-based Graphical User Interface (GUI) tool for making data ODBC/UniData SQL accessible.

This manual is organized in the following way:

- Chapter 1, “Introduction to UniData ODBC,” introduces you to the Visual Schema Generator (VSG), as well as UniData ODBC and key concepts you will see throughout this manual.
- Chapter 2, “Installing and Setting Up UniData ODBC” describes how to install UniData ODBC, the system requirements for using it, and how to configure a UniData ODBC client.
- Chapter 3, “UniData ODBC Conformance Levels,” describes how UniData ODBC conforms to the Microsoft ODBC standard.
Accessing Data from the Desktop

UniData provides several methods of accessing data from the desktop:

UniData ODBC

UniData ODBC allows Open Database Connectivity (ODBC) applications to access the UniData Relational Database Management System (RDBMS) using the ODBC interface and Structured Query Language (SQL) as a standard database access language. UniData ODBC provides an interface for custom applications and third-party tools to connect with UniData. UniData ODBC enables you to:

- Use PC-based applications that conform to the Microsoft ODBC interface, such as Microsoft Query and Microsoft Access, to access UniData.
- Develop your own custom ODBC applications using the ODBC interface.

UniOLEDB

OLE DB is a low-level interface that enables Universal Data Access (UDA), the Microsoft solution to data access challenges. UniOLEDB, the IBM OLE DB provider, harnesses this technology to provide applications in an enterprise network with direct access to UniData and UniVerse data stores.

Visual Schema Generator

Visual Schema Generator (VSG) is a Windows-based Graphical User Interface (GUI) tool for making UniData files ODBC/SQL accessible so you can access them with UniData ODBC or UniOLEDB. VSG provides GUI-based creation, deletion, and modification of UniData SQL subtables and views, as well as a GUI interface to the ANSI privilege (security) settings for the tables, subtables and views.

VSG guides data administrators through the process of defining the 1NF subtables and views that represent the nested relations by visually presenting all available configuration options. VSG performs logical error checking through every step of the schema generation process.

For more information on VSG, see Using VSG and the Schema API.
UniData ODBC

ODBC allows applications to access data in Database Management Systems (DBMS) using SQL as a standard for accessing data. The ODBC interface is a standard, developed by Microsoft, that allows a single application to access data from multiple DBMS’ without the application targeting a specific DBMS. Users of the application can then add a driver that links the application to their selected DBMS.

UniData ODBC is the driver that links desktop applications that support ODBC to UniData.

Components of ODBC

ODBC has components described in the following table.

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td>The application performs processing and calls ODBC functions to submit SQL statements and retrieve results.</td>
</tr>
<tr>
<td>Driver Manager</td>
<td>The Microsoft ODBC Driver Manager loads drivers for the application to use.</td>
</tr>
<tr>
<td>Driver</td>
<td>The driver submits SQL statements to a data source and returns the results. If necessary, the driver changes the syntax of the SQL statement to conform to syntax supported by the associated DBMS.</td>
</tr>
<tr>
<td>Data Source</td>
<td>Consists of the data the user wants to access and its associated DBMS, network (if any), and operating system.</td>
</tr>
</tbody>
</table>
The following diagram illustrates the components of ODBC:
# Components of UniData ODBC

UniData ODBC conforms to the components of ODBC in the ways described in the following table.

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td>An ODBC-compliant application that you can use to access UniData from the desktop. For example:</td>
</tr>
<tr>
<td></td>
<td>* MSQuery</td>
</tr>
<tr>
<td></td>
<td>* MSAccess</td>
</tr>
<tr>
<td></td>
<td>* Business Objects</td>
</tr>
<tr>
<td></td>
<td>Or, an ODBC compliant development tool. For example:</td>
</tr>
<tr>
<td></td>
<td>* Visual Basic</td>
</tr>
<tr>
<td></td>
<td>* Power Builder</td>
</tr>
<tr>
<td></td>
<td>* Delphi</td>
</tr>
<tr>
<td>Driver Manager</td>
<td>The Microsoft ODBC Driver Manager loads drivers for the application to use.</td>
</tr>
<tr>
<td>Driver</td>
<td>The UniData ODBC driver submits SQL statements to the server and returns the results.</td>
</tr>
<tr>
<td>Data Source</td>
<td>The server passes the SQL statements to UniData and returns the results to the UniData ODBC driver.</td>
</tr>
</tbody>
</table>

## ODBC Components
The following diagram illustrates the components of UniData ODBC connecting to UniData:

```
Application

ODBC Interface

ODBC Driver Manager

UniData ODBC Driver

UCI Client

Network

UCI Server

UniRPC

UniData
```

**Conformance Levels**

The Microsoft ODBC 2.0 standard lists functionality that a driver can support. These are defined in two areas:

- ODBC API
- ODBC SQL grammar (includes ODBC SQL data types)

This information aids in easily determining if an application can access data from a DBMS by checking the conformance levels of the driver and the application. Developers can use this information when creating an application to see if a driver they want to use supports the functionality they need.
For information on the conformance levels UniData ODBC supports, see Chapter 3, “UniData ODBC Conformance Levels.” For more information on the Microsoft ODBC conformance levels, see Microsoft ODBC 2.0 Programmer’s Reference and SDK Guide.
Definitions Used in this Manual

You will see the following terms and concepts used throughout this manual.

Application Program Interface (API)

An interface that allows a program to interact with functions in another program.

Client/server

Architecture in which the client is the process that requests services, and the server is the process that receives and then processes the request from the client. In the case of UniData, the client is either a PC or UNIX workstation that provides the user interface and performs some or most of the application processing. The server is a UNIX platform running UniData and a server which maintains the database and processes requests from the client to query or update data.

Database

What those from Pick® backgrounds call an “account.” A set of interrelated files that is created and managed by a DBMS.

Nested RDBMS

“Nested” means that the database system uses or allows the use of multivalued and multi-subvalued elements in a single field, and associations consisting of related multivalued and multi-subvalued attributes.

By allowing you to assign multiple values to a single field, the common element can contain more than a single type of data (they can be multivalued), which means that searches can be accomplished in less time and with fewer resources. This nested relational database structure, in combination with the UniData hashing algorithm, vastly reduces the need for disk I/O, compared to more traditional database structures that rely on tables.
Open Database Connectivity (ODBC)

The Microsoft specification that allows applications to access ODBC-compliant database management systems using SQL as the standard for accessing the data.

Remote Execution (REX)

An interface that allows a program to execute a command in a remote location.

Remote Procedure Call (RPC)

An interface that allows a program to call another program in a remote location.

Schema

The definition of an entire database. UniData ODBC and UniOLEDB cannot access UniData files directly and need various levels of translations in order to retrieve data from the UniData database environment. UniData calls this translation the generation of the schema. The schema for a UniData database consists of system tables and internal representations (views) that allow applications to access data in UniData files.

Structured Query Language (SQL)

SQL is a language you use to query and process data in a relational database.
Installing and Setting Up UniData ODBC

Introduction .......................................................... 2-2
Before You Begin ...................................................... 2-3
Requirements .......................................................... 2-4
  Server Requirements ............................................. 2-4
  PC Client Requirements .......................................... 2-4
Setting Up the UCI Configuration File ......................... 2-5
Verifying That UniRPC Is Running ............................... 2-8
  UCI Connection Timeout Configuration ...................... 2-9
Running Concurrent UniData Versions ......................... 2-10
Making UniData Accounts Accessible ......................... 2-11
  Tracing Events .................................................. 2-12
Presenting Data in ODBC–Accessible Format ................. 2-13
  Adding a UniData ODBC Data Source ....................... 2-14
  Modifying a ODBC Data Source ............................. 2-17
  Deleting a ODBC Data Source ............................... 2-19
Viewing ODBC Driver Information ............................ 2-20
  Tracing Calls to ODBC Drivers ............................ 2-22
Viewing ODBC Core Component Information ................ 2-24
Introduction

This chapter describes how to install and set up UniData ODBC 3.0 on your PC:

- For information you will need before you begin using UniData ODBC, see “Before You Begin.”
- For hardware and software requirements for using UniData ODBC, see “Requirements.”
- For information about setting up UniData ODBC, see “Setting Up the UCI Configuration File.”
- The ODBC Administrator manages data sources for ODBC applications. When an ODBC application makes a request to a data source, the ODBC Administrator uses the information you enter about a data source to make the connection. See “Administering ODBC Data Sources in Microsoft ODBC 3.0.”
Before You Begin

Before you can use UniData ODBC to access data in UniData from your PC, you must first:

- Install UniData 7.2 on your server platform. For more information on how to install UniData, see *Installing and Licensing UniData Products*.
- Make sure the files you want to access in UniData are ODBC/SQL accessible. For more information, see *Using VSG and the Schema API*. 
Requirements

The following lists explain the system and software requirements for the UniData ODBC driver.

Server Requirements

- UNIX, Windows NT 4.0 or greater platforms
- TCP/IP
- UniData Release 5.1 or greater

PC Client Requirements

- IBM-compatible personal computer attached to a network
- Microsoft Windows 95 or greater or Windows NT 4.0 or greater
- Approximately 21 megabytes (MB) of free disk space
- A minimum of 8 MB of random access memory
- Appropriate network software

*Install UniData ODBC on the client using the standard Microsoft Windows installation procedure.*
Setting Up the UCI Configuration File

UniData ODBC access UniData data sources through entries in the UCI configuration file (uci.config) on the client machine. This file contains connection parameters needed to route requests to the UCI server. When ODBC attempts to connect to a data source, UniData ODBC reads the UCI configuration file to determine the host system, DBMS type, server name, and other optional information.

**Note:** The UCI configuration file that UniData ODBC uses is specified in the UciCfgFile key in the system registry under \HKEY_LOCAL_MACHINE\SOFTWARE\IBM\UniClient\UCI.

UniData ODBC can access UniData databases residing on various operating systems. Each configuration entry describes the physical attributes of a database in sufficient detail to perform three tasks:

- Establish communications.
- Start a UniData server process.
- Route query and update requests.

In the uci.config file on the client machine, you must define the UCI data sources to which you want UniData ODBC to connect. You can define the UCI data sources in either of the following ways:

- Use any text editor to modify the file.
- Use the UCI Config Editor to manage data sources. For information about defining data sources with the UCI Config Editor, and for information about parameters in the UCI configuration file, see *Administrative Supplement for Common APIs*.

The following example illustrates the default configuration file shipped with UniData:

```
[ODBC DATA SOURCES]
<localuv>
DBMSTYPE = UNIVERSE
NETWORK = TCP/IP
SERVICE = uvserver
HOST = localhost

<localud>
DBMSTYPE = UNIDATA
NETWORK = TCP/IP
SERVICE = udserver
HOST = localhost
```
This default configuration file enables you to access a UniData database on the same hardware platform as the one on which your application is running.

You can add as many of entries in the uciconfig file as you want, each with a different data source name.

To access a remote database on a different platform, you must add an entry to the configuration file. For example, if the remote system you want to access is named hq1 and the account path is /usr/myacct, make up a data source name such as corp1 and add the data source definition to the uci.config file as follows:

```
<corp1>
  DBMSTYPE  = UNIDATA
  NETWORK   = TCP/IP
  SERVICE   = udserver
  HOST      = hq1
  ACCOUNT   = /usr/myacct
  USERNAME  = myloginname
```

The ACCOUNT parameter can be set to any one of the following:

- The full path to a UniData account.
- A valid UniData database name.

A UniData database name is valid if it appears as an entry in the ud_database file. For UNIX systems, this file is located in /usr/ud72/include. For Windows platforms, it is located in \udthome\include. For examples of database entries in the ud_database file, see “Making UniData Accounts Accessible” on page 2-10.

**Note:** If you modify uci.config by using a text editor, make sure you surround the equal signs with spaces.

This chapter describes how to access data in UniData tables and files. You must perform the following tasks:

- Verify that the UniRPC daemon (for UNIX systems) or the UniRPC service (for Windows platforms) is running.
- Make UniData accounts accessible.
- Present the data in UniData in a format that is accessible to UniData ODBC.
Verifying That UniRPC Is Running

When UniData ODBC requests a connection to the UniData UCI server, the UniRPC facility starts the server (udserver) process, which runs on the machine where the database resides. Before you use UniData ODBC, make sure that UniRPC is running.

On UNIX systems, you can verify whether UniRPC is running by typing the following command:

```
ps -ef | grep unirpc
```

On UniData for UNIX, the UniRPC daemon is started when you execute the startud command.

On Windows platforms, you can verify whether UniRPC is running by double-clicking the **Services** icon on the **Control Panel**. The **Services** dialog box appears. If UniRPC is running, it appears in the list of services with the status “Started.” If UniRPC is not running, you can start it from the Services dialog box.

**Note:** For both UNIX and Windows platforms, UniRPC is installed automatically when you install UniData.

On UNIX systems, the UniRPC services file (unirpcservices) contains an entry similar to the following:

```
udserver /usr/ud72/bin/udsrvd * TCP/IP 0 3600
```

The unirpcservices file is located, by default, on the UCI server in the...

On Windows platforms, the unirpcservices file is located on the UCI server in the following default path:

```
x:\IBM\unishared\unirpc
```

where \( x \) is the drive on which the software is installed. It contains an entry that is similar to the following:

```
udserver C:\usr\ud72\bin\udsrvd.exe * TCP/IP 0 3600
```

Your unishared directory might not be located in the default path. To determine its actual location on UNIX systems, enter:

```
cat /.unishared
```
This provides the path for the unishared directory.

For Windows platforms, you can find the path for unishared by looking in the system registry under `\HKEY_LOCAL_MACHINE\SOFTWARE\IBM\unishared`.

When the client system requests a connection to a UCI server, the local UniRPC daemon or service uses the unirpcservices file to verify that the client can start the requested server (in this case, udserver).

**Note:** With the Server edition of UniData, each UniData ODBC connection to a UniData UCI server consumes one UniData license. With the Workstation, Workgroup, or Enterprise edition of UniData, device licensing enables each client system to establish up to ten connections to the UCI server from one device while consuming only one database license — regardless of whether more than one user login is used to make the connections. For more information about device licensing, see Installing and Licensing UniData Products.

### UCI Connection Timeout Configuration

The six-minute default inactivity timeout value (3600) for UCI connections can be too short. If users leave UniData ODBC connections open, but the connections remain inactive for longer than six minutes, they could receive UniRPC error code 81015. To increase this timeout value, use any text editor to modify the unirpcservices file (for the path to this file, see “Verifying That UniRPC Is Running” on page 2-7).

**Note:** To edit the unirpcservices file, you must have root privileges on UNIX or Administrator privileges on Windows platforms.

The following example illustrates how to set the connection timeout delay to 24 hours. On the line starting with udserver, change the right-most number to 864000 (the number of “tenths of a second” in 24 hours). The line could appear as follows:

```
udserver /usr/ud72/bin/udsvrd * TCP/IP 0 864000
```
Running Concurrent UniData Versions

When you install UniData 7.2 on a machine where UniData 6.1 was previously installed, the unirpcservices file is overwritten with UniData 7.2 information. If you want to run UniData 6.1 concurrently with UniData 7.2, you must edit the unirpcservices file to enter the UniData 6.1 definitions. See *Administrative Supplement for Common APIs* for the format of the unirpcservices file.

The following example illustrates unirpcservices file entries when running UniData 6.1 concurrently with UniData 7.2:

```
udserver_61 c:\IBM\ud61\bin\udsrvd.exe * TCP/IP 0 3600
udserver_72 c:\IBM\ud72\bin\udsrvd.exe * TCP/IP 0 3600
```

Make sure the uci.config file contains an entry for the server for each version of UniData, as shown in the following example:

```
<unidata61>
  DBMSTYPE = UNIDATA
  network = TCP/IP
  service = udserver_61
  host = server1
<unidata72>
  DBMSTYPE = UNIDATA
  network = TCP/IP
  service = udserver_72
  host = server1
```
Making UniData Accounts Accessible

UniData databases are organized into accounts. UniData ODBC connects to a UniData account and can access the files there. You optionally can define the account as a database in the ud_database file on the server machine. You also can include the account path or database name in the UCI data source definition in the UCI configuration file (uci.config). For information about setting up the UCI configuration file, see “Setting Up the UCI Configuration File” on page 2-5.

Note: You also can specify the account path or database name each time you attempt to connect to the account. In this case, you would not need to include the account path or database name in the UCI configuration file. When you attempt to connect, you are prompted to specify the full path to the account or the database name.

If you want to access an account that has a UDTHOME directory different than the default UDTHOME directory, you must include a definition for that account in the ud_database file on the server machine. For UNIX systems, this file is located in /usr/ud60/include. For Windows platforms, it is located in \udthome\include. You can find the path for udthome by looking in the system registry under \HKEY_LOCAL_MACHINE\SOFTWARE\IBM\UniData\7.2. Use any text editor to modify the ud_database file.

Note: To determine your default UniData home directory, use the UNIX env command.

The following Windows NT example shows an entry in the ud_database file for a database named db2:

```
DATABASE=db2
UDTHOME=d:\disk2\test72
UDTACCT=d:\disk2\test72\testacct.
```

In the ud_database file entry, the UDTHOME parameter is optional. You should include it only when the UDTHOME directory is different than the default UDTHOME directory.

Tracing Events

By using the tracing feature, you can create logs of events between clients and the database through the server. Logs enable IBM support personnel to help troubleshoot problems. You can define trace levels for database entries in the ud_database file.
The following table describes the valid trace levels and the associated information that is written to the trace log.

<table>
<thead>
<tr>
<th>Trace Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Includes all fatal error information.</td>
</tr>
<tr>
<td>1</td>
<td>Includes all UCI commands in addition to the information provided by trace level 0.</td>
</tr>
<tr>
<td>2</td>
<td>Includes parameter information and column descriptions in addition to the information provided by trace levels 0 and 1.</td>
</tr>
<tr>
<td>3</td>
<td>Includes data values in addition to the information provided by trace levels 0, 1, and 2.</td>
</tr>
</tbody>
</table>

**Trace Levels**

The trace log is named `udsrv_database.processID` where `database` is the database name (for example, `db2`) and `processID` is the process number ID. By default, it is located in your temporary directory (typically, `/tmp` for UNIX; `x:\temp` for Windows platforms). For UNIX, you can find the location of the trace log file in the `TMP` parameter in the `udtconfig` file under `/usr/ud72/include`.

The following UNIX example shows a tracing level setting for a database named `dbase3`:

```
DATABASE=dbase3
UDTHOME=/disk1/ud72
UDTACCT=/home/test/newtest
TRACE_LEVEL=3
```
Presenting Data in ODBC–Accessible Format

Data in UniData is organized differently from the way UniData ODBC expects it to be organized. Two areas in which the data differs from what UniData ODBC expects are:

- Data types
- Multivalued and multi-subvalued data

**Data Types**

UniData does not define data types for data contained in its files. On the other hand, UniData ODBC expects data types for all data. In addition, data in UniData can be of variable length, but UniData ODBC expects data to have either a fixed or a maximum length. To make the data look more like what UniData ODBC expects, you must use the Visual Schema Generator (VSG) or the Schema API.

**Multivalued and Multi-Subvalued Data**

UniData ODBC expects data to be organized in first normal form (1NF) format. Although some files could be in 1NF format, which means that only one value is stored in each column of each row, many UniData files have columns that store multiple values in the columns of a row (NF² format). To instruct UniData SQL to present data in 1NF format, you must use VSG or the Schema API. For more information, see Using VSG and the Schema API.
Administering ODBC Data Sources in Microsoft ODBC 3.0

The ODBC Administrator is a Microsoft Windows Control Panel device that manages data sources for ODBC applications. When an ODBC application makes a request to a data source, the ODBC Administrator uses the information you enter about a data source to make the connection.

The ODBC Administrator enables you to:

- Add a new data source.
- Modify the configuration of an existing data source.
- Remove an existing data source.
- View ODBC driver information.
- Trace calls to ODBC drivers.
- View information about the ODBC core components.

For more information about the ODBC Administrator, see your Microsoft Windows documentation.

Adding a UniData ODBC Data Source

To add a data source, complete the following steps:
1. Access the ODBC Data Source Administrator

From the Start menu, click Control Panel, then double-click Administrative Tools, and then double-click Data Sources (ODBC). The ODBC Data Source Administrator dialog box appears, as shown in the following example:

![ODBC Data Source Administrator](image)

The first tab, User DSN, displays user data sources. An ODBC user data source stores information about how to connect you to the data provider you select.

*Note:* You can select either User DSN or System DSN.
2. Add the Data Source

To add a new data source, click Add. The Create New Data Source dialog box appears, as shown in the following example:

Select IBM UniData ODBC Driver, and then click Finish.

Note: You can run the UniData ODBC Driver simultaneously with the UniDesktop/ODBC driver. Run the UniDesktop/ODBC driver to access versions of UniData prior to Release 5.1.

The UniData ODBC Driver Setup dialog box appears, as shown in the following example:
3. Enter Data Source

Type the **Data Source Name**, **Description**, **Server**, **Database**, and **User** information. The following table describes each entry you must make.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source Name</td>
<td>A logical name that identifies the data source. It can be any name that does not include any of the following characters: &lt;br&gt; ? = equal sign &lt;br&gt; ? ; semicolon &lt;br&gt; ? [ ] square brackets &lt;br&gt; ? ( ) parentheses</td>
</tr>
<tr>
<td>Description</td>
<td>A description of the data source.</td>
</tr>
<tr>
<td>Server</td>
<td>The name of your server as defined in the uci.config file. The UniData ODBC driver server name must exist in this file as the UCI Data Source name.</td>
</tr>
<tr>
<td>Database</td>
<td>The name of the database on the server. You can enter either the full path of the database, or the name of the database as defined in the ud_database file on the server.</td>
</tr>
<tr>
<td>User</td>
<td>The name you use to log on to the server platform.</td>
</tr>
</tbody>
</table>

**Adding Data Source Information**

When you complete the **UniData ODBC Driver Setup** dialog box, click **OK**.

**Modifying a ODBC Data Source**

To modify a data source, complete the following steps:
1. Access the ODBC Data Source Administrator

From the Start menu, point to Settings, click Control Panel, click Administrative Tools, and then double-click Data Sources (ODBC). The ODBC Data Source Administrator dialog box appears, as shown in the following example:

![ODBC Data Source Administrator](image)

2. Select the Data Source

To modify a data source, double-click the data source you want to modify, and then click Configure. A dialog box similar to the following appears:

![UniData ODBC Driver Setup](image)
3. Modify the Data Source

Modify the Data Source Name, Description, Server, Database, and/or User information.

Click OK to save your changes, or click Cancel to exit the program without saving changes.

The ODBC Data Source Administrator appears. You can modify more data sources or click OK to exit.

Deleting a ODBC Data Source

To delete a data source, complete the following steps:

1. Access the ODBC Data Source Administrator

From the Start menu, click Control Panel, click Administrative Tools, and then double-click Data Sources (ODBC). The ODBC Data Source Administrator dialog box appears, as shown in the following example:
2. Select Data Source

Select the data source you want to delete from the User Data Sources list, and click Remove.

Viewing ODBC Driver Information

To view information about the ODBC drivers installed on your computer, complete the following steps:

1. Access the ODBC Data Source Administrator

From the Start menu, click Control Panel, click Administrative Tools, and then double-click Data Sources (ODBC). The ODBC Data Source Administrator dialog box appears, as shown in the following example:
2. View Drivers

Click the Drivers tab. The ODBC Data Source Administrator displays information about the drivers installed on your system, as shown in the following example:

![ODBC Data Source Administrator](image)

### Tracing Calls to ODBC Drivers

Using ODBC tracing, you can create logs of the calls to your ODBC drivers. This feature enables support personnel to help you debug your applications.

To trace calls to ODBC drivers, complete the following steps:
1. Access the ODBC Data Source Administrator

From the Start menu, click Control Panel, click Administrative Tools, and then double-click Data Sources (ODBC). The ODBC Data Source Administrator dialog box appears, as shown in the following example:
2. **Start Tracing**

Click the **Tracing** tab. A dialog box similar to the following appears:

![ODBC Data Source Administrator](image)

See your Microsoft Windows documentation for information about tracing to complete this dialog box.

**Viewing ODBC Core Component Information**

To view information about the ODBC core components, complete the following steps:
1. Access the ODBC Data Source Administrator

From the **Start** menu, click **Control Panel**, click **Administrative Tools**, and then double-click **Data Sources (ODBC)**. The **ODBC Data Source Administrator** dialog box appears, as shown in the following example:
2. View Component Information

Click the About tab. The **ODBC Data Source Administrator** displays information about ODBC’s core components, as shown in the following example:
Chapter 3

UniData ODBC
Conformance Levels

API Conformance Levels ............................................ 3-3
  Core API ............................................................. 3-3
  Level 1 API .......................................................... 3-3
  Level 2 API .......................................................... 3-4
UniData ODBC Conformance Levels .................................. 3-4
UniData ODBC-Supported API Functions ........................... 3-5
  Connecting to a Data Source ................................... 3-5
  Processing an SQL Statement ................................. 3-8
  Retrieving Results and Information about Results ....... 3-10
  Retrieving Catalog Information ............................ 3-11
  Retrieving Data Source and Driver Information ......... 3-12
  Error Handling ..................................................... 3-12
SQL Conformance Levels ............................................ 3-14
  Minimum SQL Grammar ........................................... 3-14
  Core SQL Grammar ................................................. 3-14
  Extended SQL Grammar ........................................... 3-14
  UniData ODBC SQL Conformance ............................ 3-15
UniData ODBC-Supported SQL Capabilities ....................... 3-16
  Data Types .......................................................... 3-16
  Data Definition Language ....................................... 3-16
  Data Manipulation Language ................................... 3-17
  Scalar Functions .................................................. 3-18
This chapter introduces you to UniData ODBC and explains how it conforms to version 2.0 of Microsoft’s ODBC specification. For more information on ODBC conformance levels, instructions on developing applications with ODBC, and ODBC API function syntax, see the Microsoft ODBC 2.0 Programmer’s Reference and SDK Guide.

This chapter describes:

- Definitions of the ODBC API Conformance Levels. For more information, see “API Conformance Levels” on page 3.
- UniData ODBC supported API functions. For more information, see “UniData ODBC-Supported API Functions” on page 5.
- Definitions of the ODBC SQL Conformance Levels. For more information, “SQL Conformance Levels” on page 14.
- UniData ODBC supported SQL capabilities. For more information, see “UniData ODBC-Supported SQL Capabilities” on page 16.
API Conformance Levels

ODBC specifies three levels of API conformance:

- Core API
- Level 1 API
- Level 2 API

The following are the definitions of the ODBC API Conformance Levels as specified in the Microsoft ODBC 2.0 Programmer’s Reference and SDK Guide.

Core API

- Allocate and free environment, connection, and statement handles.
- Connect to data sources. Use multiple statements on a connection.
- Prepare and execute SQL statements. Execute SQL statements immediately.
- Assign storage for parameters in an SQL statement and result columns.
- Retrieve data from a result set. Retrieve information about a result set.
- Commit or roll back transactions.
- Retrieve error information.

Level 1 API

- Core API functionality.
- Connect to data sources with driver-specific dialog boxes.
- Set and inquire values of statement and connection options.
- Send part or all of a parameter value.
- Retrieve part or all of a result column value.
- Retrieve catalog information (tables, columns, primary keys, foreign keys, special columns, statistics, procedures, and procedure columns).
- Retrieve information about driver and data source capabilities (API and SQL levels, data types, scalar functions).
Level 2 API

- Core and Level 1 functionality.
- Browse available connections and list available data sources.
- Send arrays of parameter values. Retrieve arrays of result column values.
- Retrieve the number of parameters and describe individual parameters.
- Use a scrollable cursor.
- Retrieve the native form of an SQL statement.
- Retrieve enhanced catalog information (privileges, keys, procedures).
- Call a translation DLL.

UniData ODBC Conformance Levels

The UniData ODBC driver is a read/write release supporting the Level 2 API conformance level.
UniData ODBC-Supported API Functions

UniData ODBC supports the following API functions which are separated into groups according to the tasks they perform. For more information on the functions in this section, see Microsoft ODBC 2.0 Programmer’s Reference and SDK Guide.

Connecting to a Data Source

The connection information for each data source is stored in the registry. When an application calls SQLConnect, the ODBC Driver Manager uses the information from the registry to load the appropriate driver and passes the SQLConnect arguments to it. The following figure shows the proper sequence of function calls for connecting to a data source and then disconnecting from the data source.
UniData ODBC supports the following functions in this category:

- SQLAllocEnv

UniData ODBC-Supported API Functions 3-6
- SQLAllocConnect
- SQLBrowseConnect
- SQLConnect
- SQLDriverConnect
- SQLGetConnectionOption
- SQLDisconnect
- SQLFreeConnect
- SQLFreeEnv
- SQLSetConnectionOption

**SQLConnect vs. SQLDriverConnect**

In order to make a connection to a UniData data source, you must have the following information:

- Server name
- Database name
- User ID
- Password

Both SQLConnect and SQLDriverConnect functions load a driver and establish a connection to a data source. The SQLConnect function uses the information entered using ODBC Administration to make the connection to a data source. For information about ODBC Administration, see Chapter 2, “Administering ODBC Data Sources in Microsoft ODBC 3.0”. With the SQLDriverConnect function, you can either establish a connection to a data source using the information through ODBC Administration, or allow the driver to prompt for a valid server name and database name to establish a connection.

For more information on SQLConnect and SQLDriverConnect, see the *Microsoft ODBC 2.0 Programmer’s reference and SDK Guide*. 
Processing an SQL Statement

An ODBC application can execute any SQL statement supported by a DBMS. ODBC defines a standard syntax for SQL statements but does not require you to use this syntax. UniData SQL-specific statements can be sent through the same interface, although an ODBC application using UniData SQL’s proprietary syntax extensions will not be able to access any other data source.

There are two ways to execute a statement:

- Prepared Execution
- Direct Execution

Prepared Execution

An application uses prepared execution if it needs to execute the same SQL statement more than once, or if it needs information about the result set prior to execution.

Direct Execution

An application uses direct execution if it needs to execute an SQL statement once and does not need information about the result set prior to execution.

Supporting Parameters

An SQL statement can contain parameter markers to indicate values that are supplied at execution time. The following three examples show how an application might use parameter markers in three types of statements: SELECT, non-SELECT, and procedure calls.

Example 1 — SELECT statements with a parameter marker:

```
SELECT CUST, TAPES-RENTED FROM CUSTOMER_TAPES_MV_SUB1 WHERE CUST = ?
```

Example 2 — INSERT statement:

```
INSERT INTO CUSTOMER_TAPES_MV_SUB1 VALUES (?,?)
```

Example 3 — call a procedure in place of an SQL statement:

```
{CALL MYPROC(?,?,?)
```
Supporting Transactions

Any application can specify auto-commit or manual-commit with SQLSetConnectOption. The default is auto-commit. In manual-commit mode, an application must call SQLTransact with COMMIT or ROLLBACK in order to terminate a transaction.

Note: When a transaction is committed or rolled back with SQLTransact, the cursor is closed and the result set is freed.

Diagrams of Processing SQL Statements

The following figure shows a simple sequence of ODBC function calls to execute SQL statements. UniData ODBC supports the following functions in this category:

- SQLAllocStmt
- SQLSetStmtOption
- SQLGetStmtOption
- SQLPrepare
- SQLBindParameter
- SQLEnumParams
- SQLParamData
- SQLParamOptions
- SQLPutData
- SQLExecute
- SQLExecDirect
- SQLTransact
- SQLCancel
- SQLRowCount
- SQLEnterCriticalSection
- SQLGetCursorName
- SQLSetCursorName

Note: The capability of SQLParamOptions to specify multiple values for parameters is not yet supported. SQLOpenParam is not supported.

You can process multiple statements simultaneously for a given connection handle.
## Retrieving Results and Information about Results

The data you retrieve as a result of executing an SQL SELECT statement is called a result set. It can contain zero or more rows. Your application can assign storage for results before or after it executes an SQL statement. It then calls SQLFetch to move to the next row of the result set and retrieve the data. The following figure shows a typical data retrieval operation.
SQL UPDATE, INSERT, and DELETE statements do not return result sets. Instead, your application can query the number of rows affected by an INSERT, DELETE, or UPDATE statement. The following figure shows a typical INSERT, DELETE, or UPDATE operation:

UniData ODBC supports the following functions in this category:

- SQLNumResultCols
- SQLDescribeCol
- SQLColAttributes
- SQLBindCol
- SQLGetData
- SQLFetch
- SQLRowCount

**Note:** UniData ODBC supports only forward-scrolling capabilities.

UniData ODBC does not support rowset cursors and positioning within a rowset.

**Retrieving Catalog Information**

UniData ODBC supports the following functions in this category:
UniData ODBC-Supported API Functions 3-12

■ SQLTables
■ SQLColumns
■ SQLPrimaryKeys
■ SQLForeignKeys
■ SQLSpecialColumns
■ SQLStatistics
■ SQLProcedures
■ SQLProcedureColumns
■ SQLTablePrivileges
■ SQL ColumnPrivileges

Retrieving Data Source and Driver Information

UniData ODBC supports the following functions in this category:

■ SQLDataSources
■ SQLDrivers
■ SQLGetInfo
■ SQLGetTypeInfo
■ SQLGetFunctions

Error Handling

UniData ODBC supports the following error handling protocol requirements of ODBC:

■ Use of the error text to identify the source of an error.
■ Rules to ensure consistent and useful error information.
■ Responsibility for setting the ODBC SQLSTATE based on the native error.
ODBC defines a set of ODBC SQLSTATEs for each ODBC API function. UniData ODBC maps each of the native errors or states into ODBC SQLSTATEs and assigns appropriate SQLSTATEs at the environment, connection, or statement level. The error text reported back by the ODBC driver uses the following syntax for errors that occur in the UniData data source:

"[IBM U2] [SQL Client] [UNIDATA] data-source-supplied-text"

and for errors generated outside of the Unidata data source:

"[IBM U2] [UniData ODBC Driver] component-supplied-text"

UniData ODBC supports the SQLError function in this category.
SQL Conformance Levels

ODBC specifies three levels of SQL conformance:

- Minimum SQL Grammar
- Core SQL Grammar
- Extended SQL Grammar

The following is the definition of the ODBC conformance levels as specified in the Microsoft ODBC 2.0 Programmer’s Reference and SDK Guide.

Minimum SQL Grammar

- Data Definition Language: CREATE TABLE and DROP TABLE.
- Data Manipulation Language: simple SELECT, INSERT, searched UPDATE and DELETE.
- Expressions: simple.
- Data Types: VARCHAR, CHAR, LONG VARCHAR

Core SQL Grammar

- Minimum SQL Grammar.
- Data Definition Language: ALTER TABLE, CREATE INDEX, DROP INDEX, CREATE VIEW, DROP VIEW, GRANT and REVOKE.
- Data Manipulation Language: full SELECT.
- Expression: subquery, set functions (like SUM, MIN etc).
- Data Type: DECIMAL, NUMERIC, SMALLINT, REAL, INTEGER, FLOAT, DOUBLE PRECISION.

Extended SQL Grammar

- Minimum and Core SQL Grammar.
- Data Modification Language: outer joins, positioned UPDATE, positioned DELETE, SELECT for UPDATE, unions.
- Expressions: scalar functions, date, time, timestamp literals.
- Data Types: BIT, TINYINT, BIGINT, BINARY, VARBINARY, LONGVARBINARY, TIMESTAMP, DATE, TIME.
- Batch SQL statements.
- Procedure calls.

**UniData ODBC SQL Conformance**

UniData ODBC is a read/write release supporting the minimal SQL grammar conformance level. Additionally, the UniData ODBC supports selected Core and Extended SQL capabilities.

*Note:* CREATE and DROP statements are not supported in this release of UniData ODBC.

**Returning Data**

The UniData ODBC driver returns all data residing in a character-type attribute (VARCHAR type column), rather than truncating it to the length specified in the format field of the corresponding attribute definition in the UniData dictionary. For example, if the format specification of a dictionary attribute is 10L, prior to UniData 6.0, UniData ODBC truncated the data to 10 characters before returning it to the application. Beginning at this release, data is no longer truncated.
UniData ODBC-Supported SQL Capabilities

UniData ODBC supports the following SQL capabilities.

Data Types

UniData ODBC supports the following capabilities in this category:

Minimum SQL support:
- VARCHAR

Core SQL support:
- INTEGER
- FLOAT

Extended SQL support:
- DATE — yyyy-mm-dd format
- TIME

Data Definition Language

UniData ODBC supports the following capabilities in this category:

Core SQL support:

```
GRANT {ALL | grant-privilege [, grant-privilege]...}
ON table-name
TO {PUBLIC | user-name [, user-name]... }

REVOKE {ALL | revoke-privilege [, revoke-privilege]... }
ON table-name
FROM {PUBLIC | user-name [, user-name]...}
```

Exceptions/Limitations:
- The privilege of REFERENCES is not supported.
- The restrict option of REVOKE is not supported.
- The CASCADE option of REVOKE is supported implicitly (it is the default value but you cannot specify it in you SQL statement).
Column privileges are not supported.

Data Manipulation Language

UniData ODBC supports the following capabilities in this category:

Minimum SQL support:

```
DELETE FROM table-name [WHERE search-condition]

INSERT INTO table-name [( column-identifier [, column-identifier]...)]
VALUES (insert-value[, insert-value]...)

SELECT [ALL | DISTINCT ] select_list
FROM table_reference_list
[WHERE search_condition]
[order_by_clause]

UPDATE table-name
SET column-identifier = {expression | NULL}
 [, column-identifier = {expression | NULL}]...
[WHERE search-condition]
```

Exceptions/Limitations to minimum SQL support:

- The USER keyword in `insert-value` is not supported in the INSERT statement.

Core SQL support:

```
INSERT INTO table-name [( column-identifier [, column-identifier]...)]
  {query-specification | VALUES (insert-value[, insert-value]...)}

SELECT [ALL | DISTINCT ] select_list
FROM table_reference_list
[WHERE search_condition]
[GROUP BY column_name [, column_name]... ]
[HAVING search_condition ]
[order_by_clause]
```

Exceptions/limitations to core SQL support:

- The optional AS keyword in `select-list` is not supported
- The USER keyword in `insert-value`, `select-list`, LIKE and IN predicates is not supported
Extended SQL support:

```sql
ODBC-std-esc-initiator CALL {procedure_name | procedure_name
(procedure_parameter [, procedure_parameter]... ) ODBC-std-esc-
terminator

SELECT [ALL | DISTINCT ] select_list
FROM table_reference_list
[WHERE search_condition]
[GROUP BY column_name [, column_name].. ]
[HAVING search_condition ]
[UNION select_statement]... [order_by_clause]
```

Exceptions/limitations to extended SQL support:

- The optional function specification (=) is not supported in the procedure call statement
- The USER keyword in `insert-value`, `select-list`, LIKE and IN predicates is not supported
- The optional AS keyword in `select-list` is not supported
- The ALL keyword in a UNION expression is not supported.

**Scalar Functions**

ODBC specifies two types of scalar functions which form a part of the extended SQL grammar support. The scalar functions are classified into the following sections:

- String functions
- Numeric functions

*Note: These functions are supported only when the argument you specify is a column name.*

UniData ODBC supports the following string manipulation functions:

- CHAR
- CONCAT
- INSERT
- LCASE
- LEFT
- LENGTH
UniData ODBC supports the following numeric functions:

- ABS
- ACOS
- ASIN
- ATAN
- COS
- EXP
- LOG
- MOD
- SIGN
- SIN
- SQRT
- TAN