

CA-IDMS[®] Server

User Guide
3.0



Computer Associates

This documentation and related computer software program (hereinafter referred to as the "Documentation") is for the end user's informational purposes only and is subject to change or withdrawal by Computer Associates International, Inc. ("CA") at any time.

THIS DOCUMENTATION MAY NOT BE COPIED, TRANSFERRED, REPRODUCED, DISCLOSED, OR DUPLICATED, IN WHOLE OR IN PART, WITHOUT THE PRIOR WRITTEN CONSENT OF CA. THIS DOCUMENTATION IS PROPRIETARY INFORMATION OF CA AND PROTECTED BY THE COPYRIGHT LAWS OF THE UNITED STATES AND INTERNATIONAL TREATIES.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CA PROVIDES THIS DOCUMENTATION "AS IS" WITHOUT WARRANTY OF ANY KIND, INCLUDING WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NONINFRINGEMENT. IN NO EVENT WILL CA BE LIABLE TO THE END USER OR ANY THIRD PARTY FOR ANY LOSS OR DAMAGE, DIRECT OR INDIRECT, FROM THE USE OF THIS DOCUMENTATION, INCLUDING WITHOUT LIMITATION, LOST PROFITS, BUSINESS INTERRUPTION, GOODWILL, OR LOST DATA, EVEN IF CA IS EXPRESSLY ADVISED OF SUCH LOSS OR DAMAGE.

THE USE OF ANY PRODUCT REFERENCED IN THIS DOCUMENTATION AND THIS DOCUMENTATION IS GOVERNED BY THE END USER'S APPLICABLE LICENSE AGREEMENT.

The manufacturer of this documentation is Computer Associates International, Inc.

Provided with "Restricted Rights" as set forth in 48 C.F.R. Section 12.212, 48 C.F.R. Sections 52.227-19(c)(1) and (2) or DFARS Section 252.227.7013(c)(1)(ii) or applicable successor provisions.

First Edition, September 1996

One Computer Associates Plaza, Islandia, NY 11749
All rights reserved.

All trademarks, trade names, service marks, or logos referenced herein belong to their respective companies.

Contents

Chapter 1: Introducing CA-IDMS Server

Overview	1-1
About This Guide	1-3

Chapter 2: Preparing to Install CA-IDMS Server

Overview	2-1
Hardware and Software Prerequisites	2-1
Installing CA-IDMS Server	2-3
Setting Up CA-IDMS Server	2-3

Chapter 3: Setting Up Your Mainframe Environment

Overview	3-1
Installing CA-IDMS Server	3-1
Setting Up CA90s Services	3-1
Defining the CA-IDMS System	3-2
Setting Up for Database Access	3-4

Chapter 4: Installing CA-IDMS Server on the PC

Overview	4-1
Before Installing CA-IDMS Server	4-1
Installing CA-IDMS Server	4-2
Installing in a Networked Windows Environment	4-3

Chapter 5: Managing Data Sources

Overview	5-1
Defining Data Sources	5-2
Setting ODBC Options	5-6
Specifying CA-IDMS Administrator Defaults	5-7
Logging Errors and Trace Information	5-8
Setting Up a Server	5-10
Setting Up CAICCI/PC	5-11
Setting Language Options	5-11

Chapter 6: Connecting to a Data Source

Overview	6-1
Connecting to a Predefined Data Source	6-1
Adding and Connecting to a New Data Source	6-2
Connecting Dynamically to a Data Source Not Previously Defined	6-3

Appendix A: ODBC Programmer Reference

Overview	A-1
Debugging User Sessions	A-1
ODBC Conformance Levels	A-2
Database Type Mapping Between ODBC and CA-IDMS	A-8
SQLDriverConnect Connection String Format	A-10
Driver-Specific Connect Options	A-11
Supported Isolation and Lock Levels	A-12
Bulk Insert Support	A-12

Appendix B: COBOL Programmer Reference

Overview	B-1
Designing and Writing the Application	B-1
Precompiling the Application	B-6
Sample CA-Visual Realia COBOL Program	B-10

Appendix C: PC Client .INI Files

Overview	C-1
The ODBC.INI File	C-1
The CAIDDSI.INI File	C-2

Appendix D : Passing Accounting Information to CA-IDMS

Overview	D-1
Supplying Accounting Information	D-1

Appendix E: Installation Checklists

Overview	E-1
Checklist for Mainframe Software Installed and Configured	E-1
Checklist for PC Software Installed and Configured	E-1

Index

Introducing CA-IDMS Server

Overview

CA-IDMS Server is a connectivity tool consisting of a set of Microsoft Windows and server programs that support access from Windows applications to data stored in CA-IDMS databases. CA-IDMS Server provides support for dynamic SQL using the Microsoft Open Data Base Connectivity (ODBC) application program interface (API), as well as compiled SQL embedded in CA-Visual Realia COBOL applications.

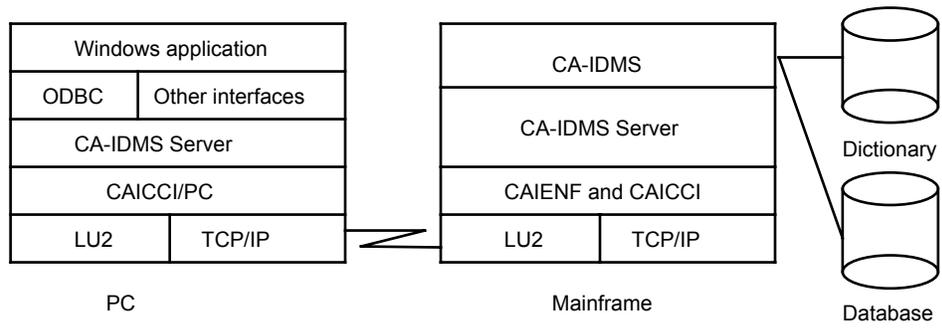
You should use this document if you are a CA-IDMS database administrator, a CA-IDMS system administrator, an ODBC application developer, a COBOL programmer, or an end user of a tool using CA-IDMS Server to access a CA-IDMS database. If you are a database administrator or end user, use this document to define data sources to access CA-IDMS data from a PC application. System administrators should use this document to set up a CA-IDMS system for access by the CA-IDMS Server. If you are an ODBC application developer, use this document to understand how ODBC API requests are implemented in CA-IDMS Server. If you are a COBOL programmer, use this document to understand how to access the CA-IDMS Server from a COBOL application that executes on a PC. This document assumes that you are familiar with the Windows user interface and can perform basic Windows and mouse operations.

Features of CA-IDMS Server

CA-IDMS Server Release 3.0, through its support of ODBC, enables open database access to CA-IDMS databases by CA software and other third-party tools and applications. Open database access allows users to maintain existing corporate databases and make the data available to Windows-based applications. Windows clients can use CA-IDMS Server to issue queries to databases on the server CPU and incorporate the result in the Windows application. Customers may also develop their own Windows applications to access their CA-IDMS databases.

Components of CA-IDMS Server

CA-IDMS Server uses the CA90s Services and CA-IDMS networking software to establish runtime communication between a Windows application on the PC and a CA-IDMS database. The following diagram shows the PC and server software components that are required for CA-IDMS Server to function:



Delivery of CA Components

Delivery of the CA software components utilized by CA-IDMS Server is shown in the chart below.

Software Component	Base Product	Installation Tape
ODBC driver manager and administration utility	CA-IDMS Server	CA-IDMS Server 3.0 diskette
CA-IDMS Server (client interface)	CA-IDMS Server	CA-IDMS Server 3.0 diskette
CAICCI/PC	CA90s Services	CA90s Services – Component must then be downloaded for installation on the PC ¹
CAICCI	CA90s Services	CA90s Services ²
CAIENF	CA90s Services	CA90s Services ²
CA-IDMS	CA-IDMS	CA-IDMS
CA-IDMS Server (server interface)	CA-IDMS Server	CA Host Server or CA-IDMS ³

¹ **For BS2000:** Distributed on diskette.

² **For BS2000:** Located on the CA-IDMS tape.

³ Please consult the product cover letter for the appropriate tape for your environment.

About This Guide

This guide assumes you are a new user of CA-IDMS Server and that you have experience using Microsoft Windows. This guide also assumes you have familiarity with CA-IDMS and database access.

In addition to this chapter, this guide includes:

- Chapter 2, “Preparing to Install CA-IDMS Server,” which provides a summary of the prerequisites and steps for installing all of the product components.
- Chapter 3, “Setting Up Your Mainframe Environment,” which describes information you need to define on the mainframe to establish communications between CA-IDMS and your Windows application using CA-IDMS Server.

- Chapter 4, “Installing CA-IDMS Server on the PC,” which describes how to install CA-IDMS Server.
- Chapter 5, “Managing Data Sources,” which explains how to use the Microsoft ODBC Administrator application to define CA-IDMS data sources.
- Chapter 6, “Connecting to a Data Source,” which explains how to use the CA-IDMS DriverConnect dialog boxes to connect to a data source.
- Appendix A, “ODBC Programmer Reference,” which describes ODBC conformance levels supported by CA-IDMS Server, data type mapping between ODBC and CA-IDMS, and driver-specific option numbers.
- Appendix B, “COBOL Programmer Reference,” which explains how to design, write, precompile, compile, and execute COBOL applications from a PC that accesses data from a CA-IDMS database. This appendix also provides a sample CA-Visual Realia application with embedded SQL statements.
- Appendix C, “PC Client .INI Files,” which provides information about the program files that CA-IDMS Server uses.
- Appendix D, “Passing Accounting Information to CA-IDMS,” which discusses how to pass accounting information from the PC client to the backend CA-IDMS system using CA-IDMS Server.
- Appendix E, “Installation Checklists,” which presents checklists of the server and PC software prerequisites and installation requirements to make CA-IDMS Server fully functional.

For More Information

CA-IDMS Server
Online Help

CA-IDMS Server provides a context-sensitive online reference to dialog boxes in the form of online help. To get online help about a dialog box, click the Help button displayed in the dialog box.

Related
Documentation

You may need to refer to the following documents when setting up CA-IDMS to work with the ODBC server:

- *CA-IDMS Installation*
- *CA-IDMS System Generation*
- *CA-IDMS System Operations*
- *CA-IDMS Database Administration*
- *CA-IDMS SQL Reference*
- *CA-IDMS SQL Programming*
- *CA-IDMS DML Reference – COBOL*
- *CA Host Server Installation*
- CA90s Services documentation, especially the *CA90s Services CAICCI User Guide*

Refer to Microsoft documentation and support for more information about Microsoft ODBC.

Preparing to Install CA-IDMS Server

Overview

For CA-IDMS Server to function, its components must be installed on both the client workstation and the host CPU. Additional prerequisites are a properly configured CA-IDMS environment on the host CPU and properly configured communications interfaces on both the host CPU and the client. Further considerations apply if the client workstation is running Windows from a Local Area Network (LAN) server instead of a local hard disk.

This chapter summarizes the following requirements for running CA-IDMS Server:

- PC hardware prerequisites
- PC software prerequisites
- Mainframe software prerequisites
- CA-IDMS Server installation
- CA-IDMS Server setup

Hardware and Software Prerequisites

CA-IDMS Server requires the mainframe software and the PC hardware and software described in this section. If the PC is connected to a LAN and using the LAN copy of Windows, some of the PC requirements can be fulfilled by the LAN server instead of the client PC.

PC Hardware Prerequisites

To install and use CA-IDMS Server on your PC, you need the following hardware:

- Any system compatible with Windows 3.1 or above
- Communications hardware as described in CA90s Services CAICCI documentation
- Graphics card compatible with Microsoft Windows
- Hard disk with at least 2.2 MB free space on your hard drive for CA-IDMS Server
- At least 4 MB of extended RAM
- Microsoft Windows compatible mouse

In a LAN environment: You will need a computer capable of running as a workstation on a network and supporting CA-IDMS Server memory requirements. CA-IDMS Server supports Novell NetWare, version 2.15 or higher (or network software supporting NETBIOS).

PC Software Prerequisites

CA-IDMS Server requires the following software to be installed either on the PC or on the LAN server to which the PC is attached:

- Microsoft Windows, version 3.1 or above
- A Windows-based 3270 emulator or a TCP/IP package supported by CAICCI
- CAICCI/PC

About CAICCI/PC

CAICCI/PC is a CA90s Services component delivered on the CA90s Services tape for your mainframe operating system. After installing the component on the mainframe, you download it to the PC for installation and configuration with your communications software.

For More Information

Refer to the following documentation:

- For information about installing CA90s Services on the mainframe, downloading and configuring CAICCI/PC, and supported SNA LU2 and TCP/IP products, CA90s Services documentation
- For additional information about configuring CAICCI/PC using the CAICCI/PC Configurator under Windows, *CA90s Services CAICCI User Guide*

Mainframe Software Prerequisites

CA-IDMS Software CA-IDMS Server requires that CA-IDMS Release 12.0 (or higher) be installed on the mainframe. You also need the SQL option.

Refer to the CA-IDMS installation and maintenance guide for your mainframe operating system for information about installing CA-IDMS on the mainframe.

CA90s Services Software CA-IDMS Server requires that the following CA90s Services software be installed on the mainframe:

- CAIENF
- CAICCI

For information about installing CAIENF and CAICCI on the mainframe, refer to CA90s Services documentation.

Installing CA-IDMS Server

To install CA-IDMS Server, you perform an install procedure on:

- The mainframe where CA-IDMS is installed
- Each workstation PC that will access CA-IDMS databases on the host CPU

For Step-By-Step Instructions

Refer to the following documentation:

- To install CA-IDMS Server on the mainframe, *CA-IDMS Installation or CA-Host Server Installation*
- To install CA-IDMS Server on the PC, Chapter 4, "Installing CA-IDMS Server on the PC," in this manual

Setting Up CA-IDMS Server

Mainframe To set up CA-IDMS Server access on the mainframe, each CA-IDMS system to be accessed as a CA-IDMS data source server must be generated with the following definitions:

- A CCI line
- A PTERM/LTERM pair for each PC user that needs concurrent access to the CA-IDMS system
- A CA-IDMS Server task
- The SQL definitions in the catalog area of the dictionary associated with the CA-IDMS system

PC To set up CA-IDMS data source definitions on the PC, you must run the ODBC Data Source Administrator application.

For Step-By-Step Instructions

Refer to the following chapters in this document:

- To set up CA-IDMS Server on the mainframe, Chapter 3, “Setting Up Your Mainframe Environment”
- To set up CA-IDMS Server on the PC, Chapter 5, “Managing Data Sources”

Setting Up Your Mainframe Environment

Overview

This chapter describes what you need to do on the mainframe to establish communications between your Windows application and CA-IDMS using CA-IDMS Server. The discussion describes:

- Installing CA-IDMS Server
- CA90s Services requirements
- CA-IDMS system generation parameters necessary to establish communications
- Things to consider when you define a database to be accessed by CA-IDMS Server

Installing CA-IDMS Server

CA-IDMS Server must be installed on the mainframe. This host component of CA-IDMS Server is delivered on the CA-IDMS installation tape. Refer to *CA-IDMS Installation* or *CA Host Server Installation* for more information about installing CA-IDMS Server on the mainframe.

Setting Up CA90s Services

CA-IDMS Server uses the CAIENF (Event Notification Facility) and the CAICCI (Common Communications Interface) components of CA90s Services. The Server supports the following communication protocols:

- LU2
- LU2 with Structured Fields
- Transmission Control Protocol/Internet Protocol (TCP/IP)

For information about CAIENF and CAICCI, refer to CA90s Services documentation.

Defining the CA-IDMS System

The following information describes how to define the CA-IDMS system generation parameters to support communications using CA-IDMS Server. Include these definitions in each CA-IDMS system to be accessed from a CA-IDMS data source.

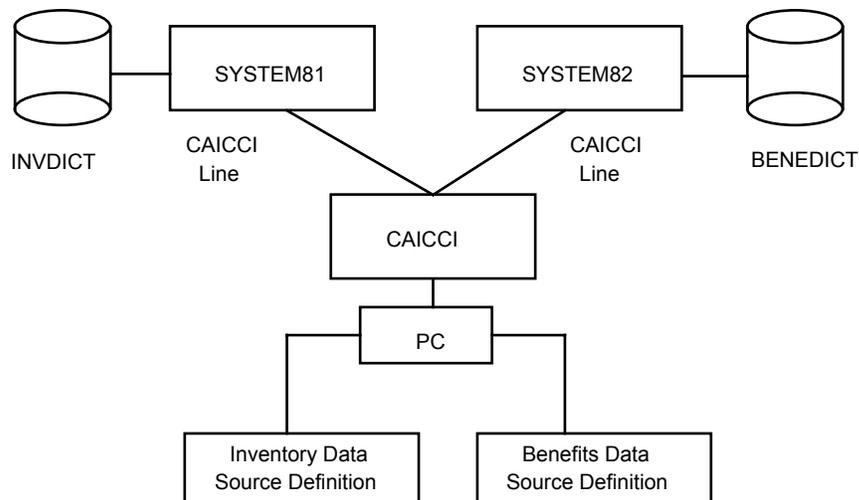
For detailed information about these system generation statements and how to use the system generation compiler, refer to *CA-IDMS System Generation*. Additionally, *CA-IDMS System Operations* provides information about configuring and maintaining CA-IDMS systems.

Examples

The examples provided in this chapter are based on a CA-IDMS system network in which both System 81 and System 82 use CAICCI to communicate with the PC.

Associated with System 81 is INVDICT, an Inventory dictionary. Associated with System 82 is BENEDICT, a Benefits dictionary. INVDICT and BENEDICT contain the definitions of the tables to be accessed by the PC. CA-IDMS Server ODBC Setup on the PC defines the Inventory Data Source and Benefits Data Source definitions that CA-IDMS Server uses to access the INVDICT and BENEDICT dictionaries, respectively, on the mainframe.

The following diagram illustrates the CA-IDMS system network:



Defining a CAICCI Line

CAICCI provides communication between mainframes or between a mainframe and PC that is independent of any particular communications protocol or access method. A CAICCI line connects a CA-IDMS system with the CAICCI network. Define a CAICCI line in each CA-IDMS system that will be accessed as a CA-IDMS data source.

Add one CAICCI line for the CA-IDMS system you are defining plus a PTERM and LTERM pair for each PC user that needs concurrent access to the CA-IDMS system. For example, if five PC users will be logged on to a CA-IDMS system at the same time, define five PTERM and LTERM pairs.

Note: Each Windows application accessing data on a CA-IDMS system uses an LTERM/PTERM pair. Therefore, a single PC user can use more than one LTERM by executing multiple applications simultaneously.

Example

This example defines a CAICCI line for System 81. The LINE, PTERM, and LTERM statements below define a CAICCI line with two physical terminals, which allows two PC users the ability to be logged on to System 81 at the same time:

```
ADD SYSTEM 81
    SYSTEM ID IS SYST0081

ADD LINE CCILINE
    TYPE IS CCI.

ADD PTERM PTECCI01
    TYPE IS BULK

ADD LTERM LTECCI01
    PTERM IS PTECCI01.

ADD PTERM PTECCI02
    TYPE IS BULK

ADD LTERM LTECCI02
    PTERM IS PTECCI02.
```

Creating the CASERVER Task

The TASK statement defines a task and its characteristics, including the code used to invoke the task. The default task code is CASERVER. You can override the default task code if you want to control resources per user or to apply additional security.

The CASERVER task code is similar to the RHDCNP3S task code, which controls the resource limits and time-out values for CA-IDMS external user sessions. Define a CASERVER task code for each CA-IDMS system that will be accessed by CA-IDMS Server. To do this:

1. From the system generation compiler, enter this command:
display task rhdcnp3s as syntax.

The system generation compiler displays the definition of the RHDCNP3S task code.

2. Erase the DISPLAY TASK statement at the top of the screen.
3. Change the name of the task code, RHDCNP3S, to CASERVER (or whatever task code you have chosen) and modify the task definition like this:
 - Add the INTERNAL parameter if it is not already there
 - Make the value for the EXTERNAL WAIT parameter appropriate for your users; for example, 1800 (half an hour)

Note: If you do not define a CASERVER task code, CA-IDMS uses the RHDCNP3S task code to define the characteristics for a CA-IDMS Server session.

Setting Up for Database Access

You must install the CA-IDMS SQL Option and the CA-IDMS Server products on the mainframe if you want to access CA-IDMS databases using the ODBC layer through either an existing application, such as CA-Visual Express, through a user-written application in a language that supports the ODBC function calls, or through a CA-Visual Realia COBOL program with embedded SQL. CA-IDMS Server issues dynamic SQL requests to access CA-IDMS databases for ODBC applications. The database can be a non-SQL-defined database or a database defined using SQL DDL. In either case, you must include the appropriate SQL definitions in the dictionary associated with the CA-IDMS system. The SQL definitions reside in the catalog area of the dictionary.

You must include an SQL schema definition in the appropriate catalog for each database (either SQL-defined or non-SQL-defined) that CA-IDMS Server will access. For more information on working with SQL-defined and non-SQL-defined databases, see the *CA-IDMS SQL Reference*.

You should consider using the CA-IDMS Online Command Facility (OCF) to verify that the following definitions are correctly defined:

- Database definitions for SQL access in the catalog area of the dictionary
- Proper authorizations for accessing database resources

To verify that these definitions are defined correctly, you should test SQL requests that are representative of the types of SQL requests that CA-IDMS Server will issue. For more information on using OCF, see the *CA-IDMS Command Facility*.

Setting Up for SQL Access

Before you attempt to issue SQL requests through CA-IDMS Server, make sure that you have performed the following tasks in the CA-IDMS mainframe environment:

- Define the SQL schemas to be accessed in the SQL catalog. To access non-SQL-defined data, you must associate an SQL schema name with the non-SQL-defined schema.
- Define a DC profile that contains a default schema name for each user. You must do this if the PC client application has no provision for entering a schema name within the PC client application itself.
- If SQL access is to be used with a non-SQL-defined database, then you must verify that the network schema conforms to the rules described later in this chapter.
- To limit the number of tables returned to the PC user in a table list function, create accessible tables views for each individual user or group of users. By creating a view of accessible tables, you generate a more meaningful list of tables for each user and also, improve performance. For more information about accessible tables views, see the ODBC Options dialog box description in Help.
- Define views in the catalog to provide easy access to non-SQL-defined databases or application-specific data. For example, consider using a view when tables are to be joined using the set-name condition. Remember, however, that views created by joining two or more tables cannot be updated.

Utilizing Page Groups

You should consider how data sources utilize CA-IDMS page groups. A page group is a physical database definition attribute set by the DBA during database definition. The catalog and the target database can be in different page groups. However, tables from multiple page groups cannot be accessed with a single request. Additionally, once a table from one page group has been accessed, a table from a different page group cannot be accessed without an intervening COMMIT.

Review these tips to use data sources defining data from multiple page groups:

- If the catalog contains definitions of tables from multiple page groups, define a different data source and a different Accessible Table View for each page group. Each Accessible Table View should include tables from a single page group. Therefore, the end user will not be allowed to accidentally access multiple page groups after a table list function is performed.

- Use the Automatic Commit option when accessing tables in different page groups.

Note: Automatic Commit is enabled by default and can be disabled only by setting an ODBC Connect attribute explicitly.

See *CA-IDMS Database Administration – Volume I* for more information about using page groups.

Setting Up for SQL Access to Non-SQL Databases

This section reviews the transformations used by the SQL engine when it reads definitions of non-SQL records. When you use SQL to access non-SQL records, the entity names you code in the SQL syntax must follow the conventions described below.

Accessing Non-SQL Records Using SQL Statements

To reference an SQL table in SQL statements, you code the table name preceded by a schema name qualifier. For example, in this statement:

```
SELECT * FROM DEMOSCH.SAMPLE
```

SAMPLE is the table name and DEMOSCH is the SQL schema in which it is defined. The combination of schema name and table name allows the SQL compiler to look up the definition of the table in the SQL catalog.

To access a non-SQL record from an SQL statement, you must code the record name the same way. To do this, define an SQL schema that maps to the corresponding non-SQL schema. Then, use the SQL schema name to qualify all subsequent references to non-SQL records in SQL DML statements. For example:

```
CREATE SCHEMA SQLNET  
  FOR NONSQL SCHEMA PRODDICT.CUSTSCHM;  
SELECT * FROM SQLNET."ORDER-REC";
```

For more information about defining SQL schemas, see the *CA-IDMS SQL Reference* for syntax and information about accessing non-SQL databases and *CA-IDMS Database Administration – Volume 1* for process-related information.

Transforming Non-SQL Record and Set Names

Non-SQL record and set names may contain embedded hyphens, which are allowed in the naming conventions for non-SQL schemas, but not in the naming conventions for SQL schemas. To use record and set names with embedded hyphens in an SQL statement, enclose the names in double quotes (for example, "CUST-REC-123").

Transforming Non-SQL Element Names

CA-IDMS automatically transforms embedded hyphens in non-SQL element names to underscores when they are referenced through SQL. Therefore, to access the CUST-NUMBER element in a non-SQL record, you must code CUST_NUMBER in an SQL statement.

Creating SQL Synonyms When a FOR LANGUAGE SQL synonym is defined for a non-SQL record, CA-IDMS uses the element synonyms for all SQL access. SQL synonyms are used **only** for element names.

Defining SQL synonyms for non-SQL records is sometimes the only way to overcome column name limitations within SQL. Some non-SQL element names don't make satisfactory SQL column names, even after the hyphen is changed to an underscore. For example, if a non-SQL element name starts with a numeric character, you must still use double quotes around the element name; for example, to access 123-ORD-NUM, you would code "123_ORD_NUM" in an SQL statement.

Elements That Cannot Be Transformed Group elements, REDEFINES elements, FILLERS, and OCCURS DEPENDING ON elements are simply not available for access by SQL. From the view point of the SQL user, it is as if these elements simply weren't defined in the non-SQL record. However, the subordinate elements of a group definition are available, as are the base elements to which a REDEFINES is directed.

Fixed Occurs Element Definitions Though OCCURS ... DEPENDING ON declarations are not available for SQL access, fixed OCCURS definitions are. To the SQL user, a fixed OCCURS element appears as one column for each occurrence of the element. The column name for each occurrence is the original element name followed by an underscore and an occurrence number. If the element is declared with multiple OCCURS levels, the corresponding column names contain one underscore and one occurrence number for each "dimension" of the OCCURS declaration. For example, the element definition BUD-AMT OCCURS 12 TIMES will generate the following column names:

```
BUD_AMT_01, BUD_AMT_02, BUD_AMT_03, . . . , BUD_AMT_12.
```

Note in the preceding example, that the occurrence number appended to the column name is large enough to hold the largest subscript from the corresponding element definition.

The base element name in combination with the appended occurrence information cannot be greater than 32 characters. If it is, you must define an SQL synonym for the non-SQL record.

Note also that, although the CA-IDMS SQL implementation allows 32-character column names, other SQL implementations restrict column names to 18 characters. In particular, some ODBC client software may require you to use SQL synonyms for non-SQL records to limit the size of the transformed column names to 18 characters.

Tip: Another way to define shorter names is to create a view of the record and specify view column names.

Defining CALC Keys,
System-Owned Index
Set Keys, and
User-Owned Sort Set
Keys

To access a control-key definition (of a CALC, INDEX, or sorted set) using SQL, the control-key definition must not include a FILLER element. If it does, change the non-SQL record definition to assign a name other than FILLER to the element(s) in question.

In addition, the control-key definition cannot incorporate the subordinate elements of a group level REDEFINES when these elements are smaller in size than the base element being redefined, as in this example:

```
02 ELEM1 PIC X(8) .
02 ELEM1REDEF REDEFINES ELEM1.
   03 ELEM1A PIC S9(8) COMP.
   03 ELEM1B PIC S9(8) COMP.
```

An error occurs if ELEM1A and ELEM1B are used in the control key definition because they are smaller than the element that they redefine (even though together they are the same as ELEM1). When this condition occurs, change the redefining group, which contains the smallest subordinate elements, into the base-element definition. Use the base-element definition in the control key specification. For example, ELEM1REDEF should be the base-element definition in the sample above, and ELEM1 should be coded so that it REDEFINES ELEM1REDEF.

For More Information

Refer to the *CA-IDMS SQL Reference* for more information about accessing non-SQL-defined databases using SQL.

Installing CA-IDMS Server on the PC

Overview

This chapter describes what you need to know to install the CA-IDMS Server on your own PC and in a network environment. It also identifies what you need to do before you install the product.

Before Installing CA-IDMS Server

Before you install CA-IDMS Server:

- Install Microsoft Windows 3.1 if it is not already installed.
- Install your Windows-based 3270 emulator or TCP/IP communications software if it is not already installed.
- If necessary, make backup copies of your CA-IDMS Server installation diskette using the DOS DISKCOPY command.

Obtain the following information from your CA-IDMS system administrator:

- The name of the dictionary that contains the definitions of the tables you want to access.
- The node name of the system on which the dictionary resides.
- The task code defined for the CA-IDMS Server. The default task code is CASERVER.

Important! *If you are installing on a workstation that is part of a networked Windows environment, please read [Installing in a Networked Windows Environment](#), later in this chapter, before performing the installation steps described in [Installing CA-IDMS Server](#), below.*

Installing CA-IDMS Server

1. To install CA-IDMS Server, start Microsoft Windows. Choose File Run in either the Program Manager or File Manager and then enter this command in the command line:

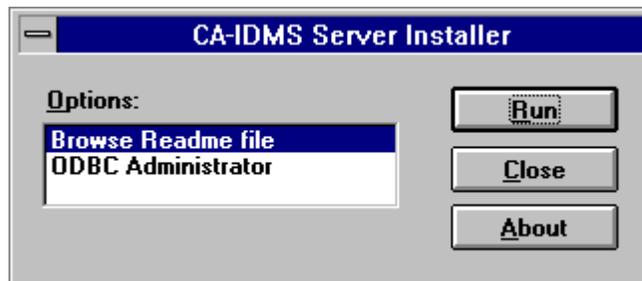
drive-letter:install

For example, to install from drive A, enter **a:install**.

2. Select the files to install and respond to the installation prompts.

Important! Do not select the ODBC Win32s Support component if you are installing under Windows NT or Windows 95.

3. When the installation is complete, the installer displays a **Installation Complete!** message and then this dialog box:



Select one of these options and then Run or choose Close to exit the installer:

- **Browse Readme file** – Displays the IDMS.TXT file in the installation directory. Read this file before using CA-IDMS Server.
- **ODBC Administrator** – Starts the Microsoft ODBC Administrator application. You can also do this by selecting an icon in the Control Panel program group, as described below.

After installing CA-IDMS Server, the Control Panel program group contains an icon for the Microsoft ODBC Administrator:



Choose the Microsoft ODBC Administrator to define CA-IDMS data sources to be accessed through ODBC or through SQL embedded in a PC COBOL application.

The installation procedure also creates or modifies the CAIDDSI.INI, ODBC.INI, and ODBCINST.INI files in the Windows directory. It also updates your AUTOEXEC.BAT file to add the installation directory to your path.

Installing in a Networked Windows Environment

To install CA-IDMS Server on a client workstation that is using a LAN server copy of Windows, you must check your environment before installation and you must perform post-installation steps to ensure that your workstation can access required files. You **cannot install** CA-IDMS Server directly into the LAN server Windows directory. You install the product on each client workstation and perform post-installation steps.

Networked Windows Environments

Perform the steps described in this section if your Windows environment conforms to any of these descriptions:

- Your workstation is diskless. Windows is running on a server, and your Windows directory is also located on the server.
- Windows is running on a server, and your home directory is also located on the server.
- Windows is running on a server, and your Windows directory is on your workstation hard disk.

Before You Install

Before you install CA-IDMS Server on a client workstation in a networked Windows environment, verify that you have system administration privilege on the network. This privilege is necessary for only the first installation on a client workstation.

Installing on the First Client Workstation

To install CA-IDMS Server on the first client workstation in a networked Windows environment, perform these steps:

1. Choose the File Run command from the Windows Program Manager; type **a:install** in the Command Line box.
2. Select the files to install and complete the installation.
3. After the components are installed, the two messages listed below may be displayed indicating that two files either can't be found or can't be executed. Choose OK after each message.

Cannot find ODBCINSTL.DLL

Could not execute ODBCADM.EXE

Note: If these messages are not displayed, skip to step 5.

4. Make current the client SYSTEM directory to which CA-IDMS Server installation added files. If the installation process created this directory, it will be in the Windows directory on your local hard disk or in the home directory on your server.

Move the files that CA-IDMS Server installed in the workstation Windows SYSTEM directory to the server Windows directory. For a list of these files, see the IDMS.TXT file.
5. Open the ODBCINST.INI file and modify the Driver= statement in the [CA-IDMS] section to point to the location where you moved the files. Save the file.
6. Open the CAIDDSI.INI file and modify the Driver= statement in the [Custom] section to point to the location where you moved the files. Save the file.
7. At this point, you can access the ODBC Administrator and configure the data sources CA-IDMS Server will access.

Installing on Additional Workstations in the Network

After you install the CA-IDMS Server on one workstation in the network, you need to create the ODBC.INI, ODBCINST.INI, and CAIDDSI.INI files in the Windows directory of the additional client workstations. To do this, select File Run from the Program manager or File manager on the client workstation. Execute this program from the directory when you installed the CA-IDMS Server files:

```
server-directory\caidinst.exe name=CA-IDMS
```

Managing Data Sources

Overview

A data source is a description of a database you want to access from your Windows application. You use the ODBC Administrator tool from the Control Panel to define a connection between the PC and CA-IDMS.

An application running under Windows can access a CA-IDMS database with SQL in either of these ways:

- By issuing CA-IDMS SQL statements
- By issuing ODBC SQL function calls

When you define a data source to be used by an application for either type of SQL access, you must include the identity of the dictionary where the SQL tables are defined and the CA-IDMS system that contains the dictionary.

The definition of a data source also includes settings for options such as access mode that affect the conditions under which ODBC SQL function calls access the database. These are described under Setting ODBC Options.

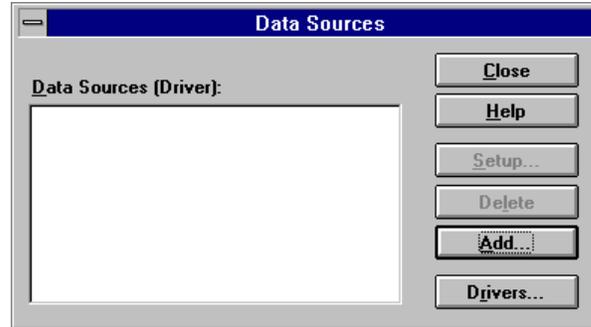
ODBC option settings do not affect CA-IDMS SQL access. Instead, CA-IDMS defaults are in effect unless they are modified by a CA-IDMS SQL statement such as SET TRANSACTION.

This chapter discusses the elements of a data source definition and how to use the ODBC Administrator tool to define and administer a data source definition.

Defining Data Sources

When you double-click the Microsoft ODBC Administrator icon in the Windows Control Panel group, the Data Sources dialog box appears. This box lists the name of each defined data source followed by the database driver in parentheses.

If no data source definition exists, the Data Sources dialog box looks like this:

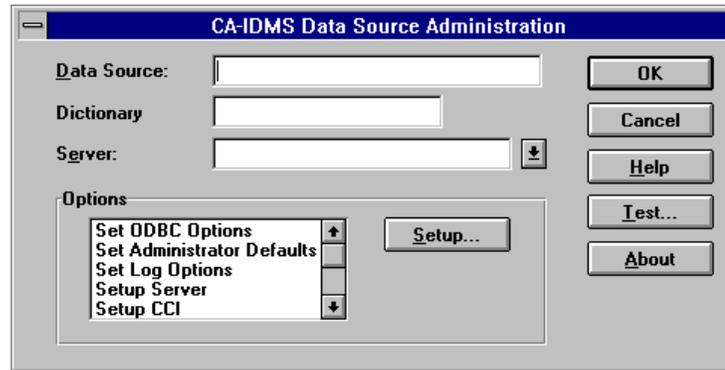


Adding a Data Source

To create a new CA-IDMS data source definition, choose the Add button. The Add Data Source dialog box appears, listing installed drivers. Select CA-IDMS.

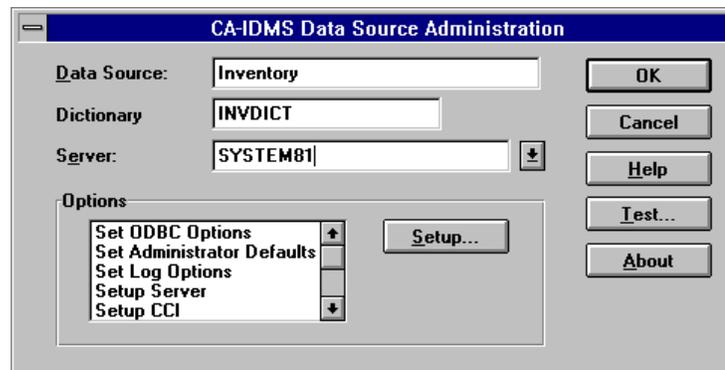


Choose the OK button. The CA-IDMS Data Source Administration dialog box appears.



In the CA-IDMS Data Source Administration dialog box, you supply the essential information for a data source definition – the data source, the name of the dictionary containing definitions of the databases, and the name of the CA-IDMS system. See Help for detailed information about completing this dialog box.

After you complete the dialog box, it looks something like this:



Note: The information displayed in this and other dialog boxes in this chapter is based on the sample CA-IDMS system network illustrated in Chapter 3, “Setting Up Your Mainframe Environment.” The values displayed in the dialog boxes are based on the system generation statements defined for System 81 and System 82.

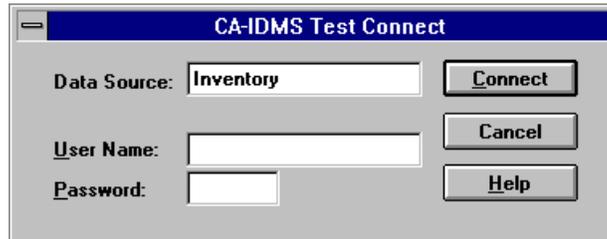
Specifying Options

After you specify the names for the data source, dictionary, and server in the CA-IDMS Data Source Administration dialog box, you can tailor data source settings by selecting an option in the Options box and then Setup. The table below describes each option and identifies the section of this chapter that documents the option in detail.

Option	Select It When	Documented In
Set ODBC Options	You want to change the default runtime attributes for the data source	Setting ODBC Options
Set Administrator Defaults	You are likely to change a particular default ODBC option setting every time you define a CA-IDMS data source	Specifying CA-IDMS Administrator Defaults
Set Log Options	You want to override the default log file specification and options, or to enable/disable tracing	Logging Errors and Trace Information
Setup Server	The name you supply in the Server combo box is not the CA-IDMS system name <i>or</i> You want to add or change a Server definition	Setting Up a Server
Setup CCI	You need to select the protocol to be used for PC-to-host communications	Setting Up CAICCI/PC
Set CECP Language	You want to select the Country Extended Code Page used to translate character data transferred between PC and host	Setting Language Options
Set DBCS Language	You want to enable double-byte character set (DBCS) processing	Setting Language Options

Testing the Data Source Definition

After you define a data source, you should test the connection for the database. To do this, choose Test in the CA-IDMS Data Source Administration dialog box. The CA-IDMS Test Connect dialog box appears.

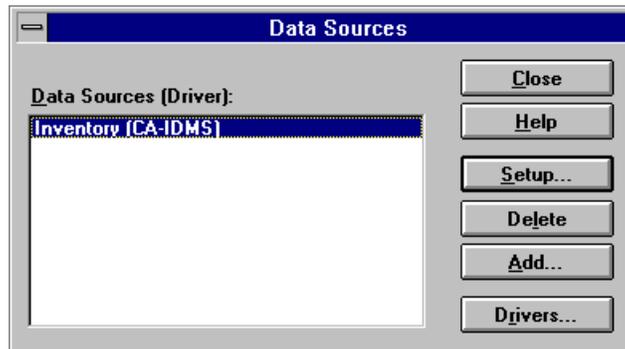


In the **User Name** text box, supply a valid user ID for the CA-IDMS system associated with the data source named in the **Data Source** text box. If security is relevant to the system, supply the password in the **Password** text box. Then choose Connect.

Saving and Maintaining the Data Source Definition

After creating and testing the data source definition in the CA-IDMS Data Source Administration dialog box, choose OK to save the definition to the ODBC.INI file and the CAIDDSI.INI file.

When you choose OK, the Data Sources dialog box appears.



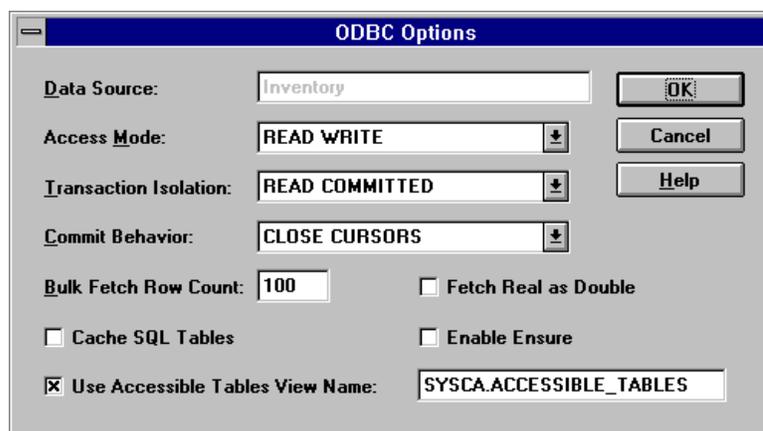
To:

- Add another data source – Choose Add.
- Modify an existing data source – Select the data source from the Data Sources (Driver) box and then choose Setup.
- Delete an existing data source – Select the data source from the Data Sources (Driver) box and then choose Delete.

To exit the Microsoft ODBC Administrator, choose Close.

Setting ODBC Options

To set ODBC options for a specified data source, select Set ODBC Options from the Options list in the CA-IDMS Data Source Administration dialog box. The ODBC Options dialog box appears.



The data source name is protected. To modify the displayed defaults for database access using this data source, choose among the options presented in the dialog box. Each option is described in detail in Help.

Select the OK button to save your changes to the database access defaults in the CAIDDSI.INI file. Select the Cancel button to close the dialog box without saving any new changes.

Performance Considerations for ODBC Options

The options you choose for ODBC can affect the performance of CA-IDMS Server. The list below identifies specific options that can affect performance:

- **Cache SQL Tables** — Use this option to increase performance at the expense of not necessarily having the most current view of existing tables. When you turn this option on, CA-IDMS Server will use the cached result to process repeated SQLTable requests. CA-IDMS Server flushes the cache whenever you turn off this option, change the request parameters, change the name of the Accessible Tables view, or disconnect from a session.
- **Enable Ensure** — The ENSURE parameter of the SQLStatistics functional call in an application normally results in an UPDATE STATISTICS command against the named table. This can be a very lengthy operation that can cause your application to time out and cause contention with other users. To override the SQLStatistics function call in the application and to prevent an application from issuing a command to update table statistics, keep the Enable Ensure option off.
- **Use Accessible Tables View Name** — By default, CA-IDMS Server uses the SYSCA.ACCESSIBLE_TABLES view for the SQLTables function. When a catalog contains a large number of table definitions, you may be better off creating a tailored view of the tables that are of interest to the end user.

Specifying a view name allows the list of tables to be tailored to the user. For example, the SYSCA.ACCESSIBLE_TABLES view returns only those tables to which the user has SELECT authority. If you define your own views, the view definition must include the following columns:

```
SCHEMA (CHAR(18))
TABLE (CHAR(18))
TYPE (CHAR(1))
```

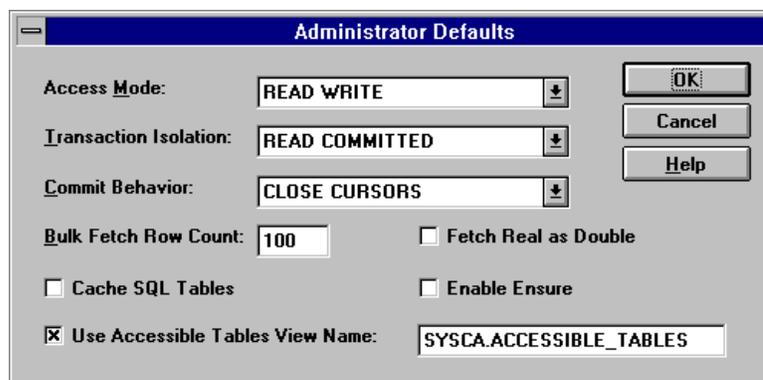
Refer to the *CA-IDMS SQL Reference* for information about the SYSCA.ACCESSIBLE_TABLES view.

Refer to Microsoft documentation about open database connectivity software for more information about the various ODBC functions mentioned in the descriptions above.

Specifying CA-IDMS Administrator Defaults

Administrator defaults establish the default option settings for a new data source. The CA-IDMS Administrator applies the settings each time you define a new data source. Existing data source definitions are not affected.

To change administrator defaults, select Set Administrator Defaults on the Options list in the CA-IDMS Data Source Administration dialog box, and choose Setup. The Administrator Defaults dialog box is displayed.



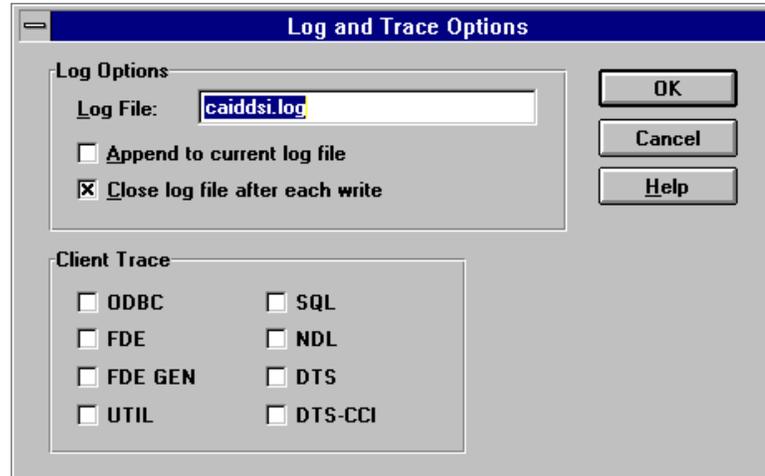
For a description of each option, see Help.

Logging Errors and Trace Information

You can write error messages and tracing information generated during a database session to a log file. The error messages indicate the status of the database connection. The trace information traces each action performed during the ODBC session. You can turn on ODBC and CA-IDMS tracing using the method described below or using ODBC commands. See Appendix A, “ODBC Programmer Reference,” for instructions on enabling tracing through an ODBC request.

Normally, you would enable tracing only to research a problem in conjunction with technical support. Log options affect all data sources. For example, if you specify a log file name, all trace entries are written to the specified file. You cannot specify different log options for different data sources.

To initiate logging, select Set Log Options from the Options list in the CA-IDMS Data Source Administration dialog box. The Log and Trace Options dialog box appears.



The default log file name is CAIDDSI.LOG. CA-IDMS Server writes messages to the log file about the status of the database connection. Client trace options control tracing on the PC. Server trace options control tracing on the server (the backend); currently they have no affect on CA-IDMS mainframe servers, but are intended for future use. For detailed information about each dialog option, see Help.

Other Log/Tracing Options

You can also use application-specific tracing and CCI tracing to help you solve problems you encounter when using CA-IDMS Server.

Application-Specific Tracing

Each application may offer tracing as an optional feature. The tracing may be the application's own internal tracing or it may be ODBC tracing, which is turned on by an ODBC function call.

CCI Tracing

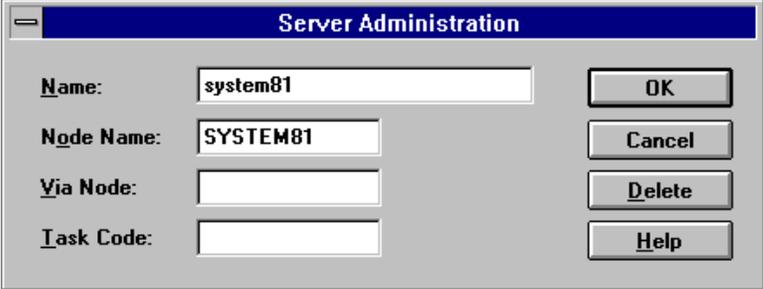
You can turn on tracing for CCI by checking the Enable Trace box in the CCI Configurator. For LU2 users, CCI tracing creates a file, CCIDUMPA.TRC, in the current directory. To make this file readable, you must execute CCITRACE.EXE against this file by entering:

CCITRACE CCIDUMPA

The CCITRACE program looks for a .TRC file and creates a CCIDUMPA.TXT file. For TCP/IP users, the default trace file created is C:\CCITCP.DAT. This file is a readable ASCII file.

Setting Up a Server

To define a Server, select Setup Server from the Options list in the CA-IDMS Data Source Administration dialog box. The Server Administration dialog box appears.



The screenshot shows a dialog box titled "Server Administration". It contains four text input fields and four buttons. The "Name" field contains "system81", the "Node Name" field contains "SYSTEM81", the "Via Node" field is empty, and the "Task Code" field is empty. The buttons are "OK", "Cancel", "Delete", and "Help".

To set up a Server:

1. Enter a server name in the **Name** text box, or if one is already displayed, you can overwrite it to associate the Server attributes with the new name. The server name must be one of the following names:
 - A descriptive name up to 32 characters containing any combination of letters, numbers, spaces, or special characters.
 - The name of the CA-IDMS system having access to the dictionary that contains the definitions of the tables to be accessed. This name must match the value of the SYSTEM ID specified in the SYSTEM statement of CA-IDMS system generation.
2. If the entry in the Name text box is not the system ID, enter the system ID in the **Node Name** text box.

If you do not specify a node name value, then CA-IDMS Server uses the first eight characters of the Name value as the system ID of the system to be accessed.

Note: The **Via Node** attribute is not used in this release.

3. Specify a task code in the **Task Code** text box. The task code identifies the task that will be initiated for the Server on the CA-IDMS target system. The task code is used to determine the resource limits and timeout values to be used when this Server is accessed.

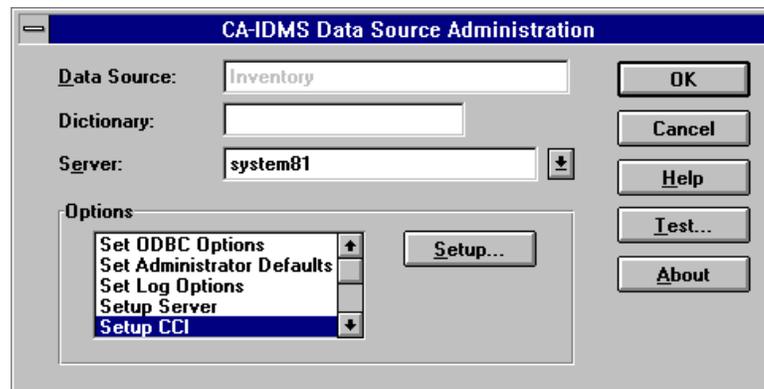
The default task code is CASERVER. You may want to create your own task code to tailor the timeout and resource limits for this Server. If you do, define the task to the CA-IDMS system with the TASK system generation statement; otherwise, CA-IDMS uses, by default, the RHDCNP3S task to determine the timeout and resource limits. For more information about the TASK statement, refer to *CA-IDMS System Generation*.

5. Select OK or Cancel to accept or cancel your Server name attributes; select Delete to remove the Server name from the CAIDDSI.INI file.

Setting Up CAICCI/PC

If you do not configure CAICCI/PC before executing your CA-IDMS Server application, your application will not be able to establish communication with a host. Select the communication protocol and set the necessary parameters before running any CA-IDMS Server application. Be aware that all CA-IDMS Server data sources will use the communications protocol and parameters most recently set.

You can access the CAICCI Configuration dialog box from the CA-IDMS Data Source Administration dialog box. Select Setup CCI from the Options list and choose Setup.



For detailed information on how to configure CAICCI/PC, see the *CA90s Services CAICCI User Guide*.

Setting Language Options

CA-IDMS Server translates character data transferred between the PC and the host system. By default, it translates EBCDIC character data on the host to ASCII on the PC using U.S. English code pages. You can override the default by creating a customized translation table that uses the Country Extended Code Pages (CECP) for different languages. You can also enable Double Byte Character Set (DBCS) processing, which serves character-based languages such as Kanji.

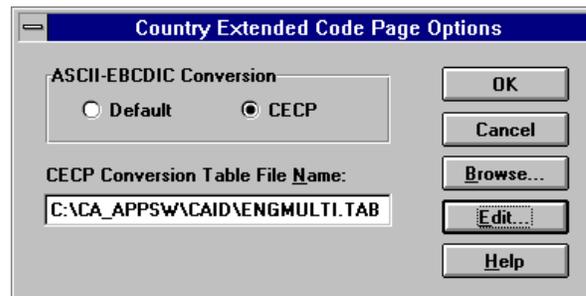
Note: The language setting is a global option which affects all data sources. You cannot establish different language options for different data sources.

Selecting, Creating, and Editing CECP Translation Tables

When CA-IDMS transfers character data between the host system and a PC, it uses a translation table based on the Country Extended Code Pages used on the host machine and PC. The default translation table uses code pages based on English as spoken in the United States (U.S. English). You may need to override the default and create a customized translation table if your host system or PC uses code pages based on another language, such as German.

To select, create, or edit a CECP translation table, select Set CECP Language from the Options list in the CA-IDMS Data Source Administration dialog box and then select Setup.

In the Country Extended Code Page Options dialog box, select the Default button to use the default translation table or select the CECP button to specify, create, or edit a customized translation table.

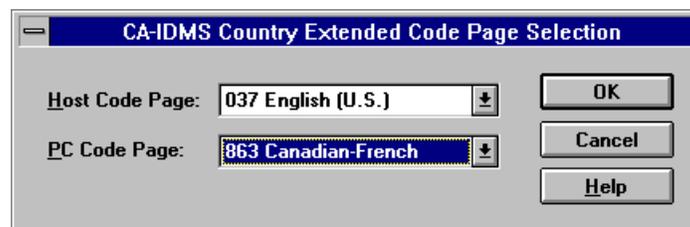


Selecting a Translation Table

To select a translation table, enter the name of the file in the **CECP Conversion Table File Name** text box or choose the Browse button to select from a list of .TAB files, if any have been created in the directory specified during CA-IDMS Server installation.

Creating or Editing a Translation Table

To create or edit a translation table, choose the Edit button. When you select the Edit button, CA-IDMS Server activates the Translation Editor. To open an existing translation table, choose Open from the File menu. For a new or existing translation table, choose Code Pages from the Edit menu to access the CA-IDMS Country Extended Code Page Selection dialog box. This dialog box, shown below, allows you to select the code pages to use for your translation table.



The **Host Code Page** list box represents these Country Extended Code Pages for the EBCDIC character set on the mainframe:

Code Page	Representative Language
037 English (U.S.)	English and most other European languages
273 German, Austrian	German and Austrian German
277 Norwegian	Norwegian
278 Finnish, Swedish	Finnish and Swedish
280 Italian	Italian
284 Spanish	Spanish
285 English (U.K.)	English and most European languages
297 French (AZERTY)	French, using the AZERTY keyboard
500 Belgian, Swiss	Belgian, Swiss French, and Swiss German

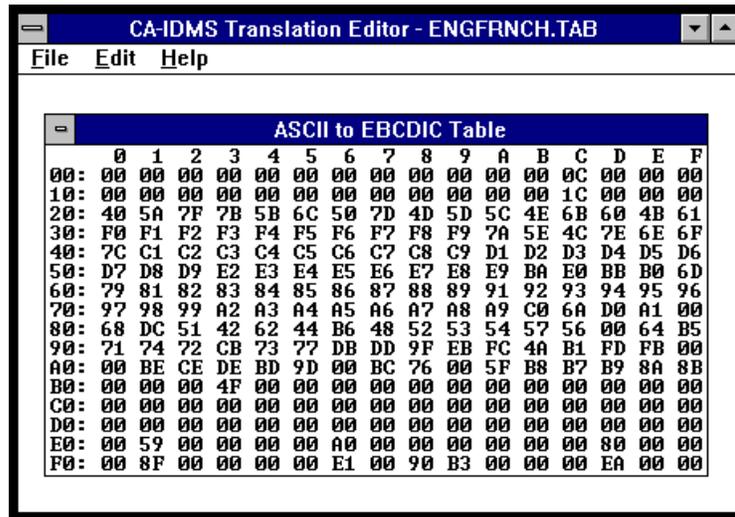
The **PC Code Page** list box represents these Country Extended Code Pages for the ASCII character set:

Code Page	Representative Language
437 English (U.S.)	English and most other European languages
850 Multilingual (Latin I)	Most languages using the Latin alphabet
852 Slavic (Latin II)	Slavic languages using the Latin alphabet
860 Portuguese	English and Portuguese
863 Canadian-French	English and French Canadian
865 Nordic	Scandinavian languages (that is, Swedish, Norwegian)

Customizing a Translation Table

After you create a translation table, you may need to add EBCDIC/ASCII conversions that are not supported in the standard code pages. The Translation Table Editor provides two edit windows – one for the ASCII to EBCDIC translation table and the other for the EBCDIC to ASCII translation table. To activate each window, select the appropriate option from the Edit menu.

Each window displays a table of 256 hexadecimal values. Each entry in the table represents the output character set code value indexed by the input character set code value. For example, this window represents the ASCII to EBCDIC translation table for Canadian French on the PC and U.S. English on the host machine. The ASCII value for a space (' ') in the Canadian French code page is 20 (in hexadecimal). The corresponding EBCDIC value for a space in the U.S. English code page is 40.



To customize the table, use the mouse or keyboard to select a hexadecimal value and replace it with another. The editor ignores all characters except the numbers 0 through 9 and letters A through F (including lowercase). You can use the mouse to move the cursor or these keystrokes:

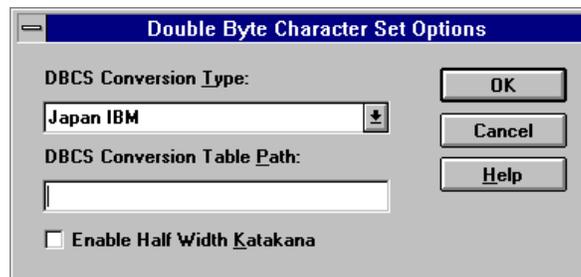
Key	Moves Cursor
Arrow keys	One digit in the direction of the arrow
Home	To beginning of row
End	To end of row
PageUp	To top row
PageDown	To bottom row
Enter	Beginning of next row
Ctrl+Left Arrow, Right Arrow	Left or right one entry
Ctrl+Home, End	Beginning or end of table

Saving a Translation Table To save a new translation table under a new name, choose Save As from the File menu. To save an existing translation table, choose Save from the File menu. The default file extension for translation table files is .TAB.

Enabling DBCS Processing

To enable DBCS processing, select Set DBCS Language from the Options list in the CA-IDMS Data Source Administration dialog box and then select Setup.

In the Double Byte Character Set Options dialog, select one of the conversion types installed on your system in the **DBCS Conversion Type** list box. In the **DBCS Conversion Table Path**, enter the path of the subdirectory that contains the DBCS conversion tables; the default path is the directory specified during CA-IDMS Server installation. When you select OK, CA-IDMS Server enables DBCS processing for all host connections. See Help for detailed information about completing this dialog box.



Connecting to a Data Source

Overview

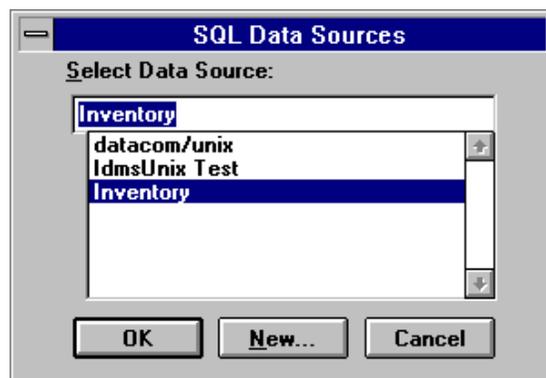
Many ODBC applications can use the CA-IDMS DriverConnect dialog boxes to connect to a data source. If your application uses them, the CA-IDMS DriverConnect dialog boxes allow you to:

- Connect to an existing data source
- Add and connect to a new data source
- Connect dynamically to a data source that has not been defined previously

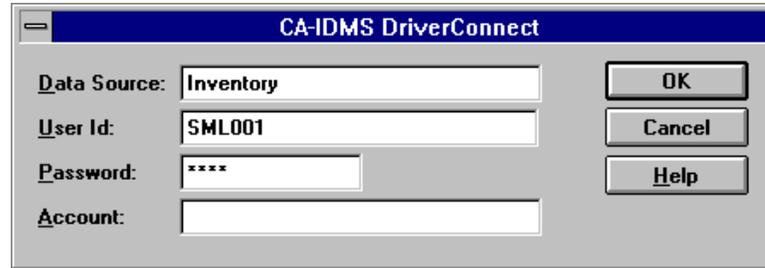
This chapter discusses the elements of data source connection and how to use the CA-IDMS DriverConnect dialog boxes to connect to a data source.

Connecting to a Predefined Data Source

Many applications use the SQL Data Sources dialog box, shown below, to connect to a data source that has been defined with the ODBC Administrator tool.



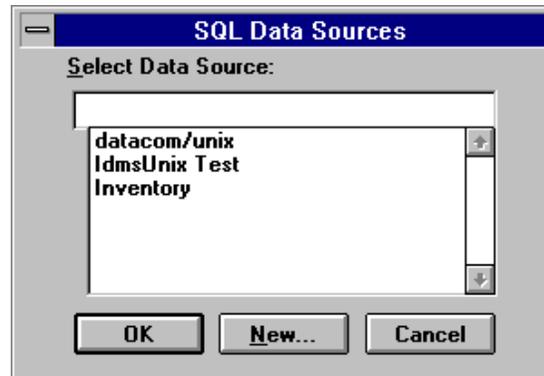
The SQL Data Sources dialog box allows you to select the desired data source from a list of defined sources. Select the data source and then OK. The CA-IDMS DriverConnect dialog box appears.



In the CA-IDMS DriverConnect dialog box, specify the user ID, password, and account, if your application uses it. See Help for detailed information. Click OK to connect to the selected data source.

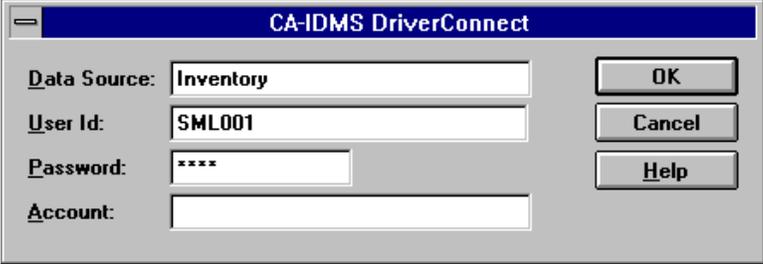
Adding and Connecting to a New Data Source

To connect to a data source that has not been previously defined with the ODBC Administrator tool, select New on the SQL Data Sources dialog box.



The SQL Data Sources dialog box runs the ODBC Administrator tool and displays the Add Data Source dialog box. Define the data source, as discussed in Chapter 5, "Managing Data Sources."

After you add the data source, the SQL Data Sources dialog box appears again with the new data source selected. Click OK. The CA-IDMS DriverConnect dialog box appears.

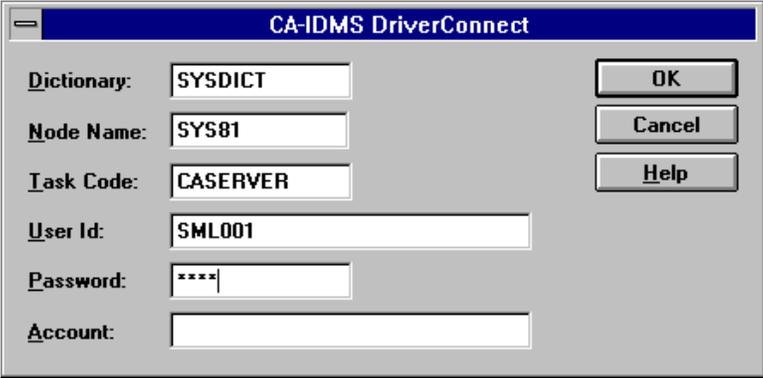


The image shows a dialog box titled "CA-IDMS DriverConnect". It contains four text input fields: "Data Source" with the value "Inventory", "User Id" with the value "SML001", "Password" with the value "****", and "Account" which is empty. To the right of the fields are three buttons: "OK", "Cancel", and "Help".

Specify the user ID, password, and account information to connect to the new data source. See Help for more detailed information about this dialog box.

Connecting Dynamically to a Data Source Not Previously Defined

Some applications allow you to connect to a data source dynamically without defining or adding the data source first. The CA-IDMS DriverConnect dialog box appears if you can connect dynamically.



The image shows a dialog box titled "CA-IDMS DriverConnect". It contains six text input fields: "Dictionary" with the value "SYSDICT", "Node Name" with the value "SYS81", "Task Code" with the value "CASERVER", "User Id" with the value "SML001", "Password" with the value "****", and "Account" which is empty. To the right of the fields are three buttons: "OK", "Cancel", and "Help".

Supply the data source connection information that will be in effect for the duration of the session. This information is a subset of the data source definition information specified with the ODBC Administrator tool. See Help for detailed information about completing this dialog box.

Note: Unless you define the specified data source with the ODBC Administrator tool, the data source no longer exists when you end the session.

ODBC Programmer Reference

Overview

The Open Data Base Connectivity (ODBC) interface allows a Windows application to access different databases using SQL without specifically targeting any particular database. A module called an ODBC driver is used to link an application to a specific database.

The ODBC interface was developed by Microsoft and is aligned closely with the international-standard ISO/IEC 9075-3:1995 Call-Level Interface. This will be adopted shortly as an ANSI standard.

Debugging User Sessions

CA-IDMS Server writes messages to the default PC log file, CAIDDSI.LOG, specified in the Log and Trace Options dialog box described in Logging Errors and Trace Information of Chapter 5, "Managing Data Sources." These messages relay the status of the PC-to-mainframe database connection. Common messages relate to a user's authorization to sign on to the database, CCI timeouts, and unsuccessful connections because the CA-IDMS system is down.

Error Messages

Error messages returned by CA-IDMS Server have one of the following formats, depending on the component in which the error is detected:

```
[CA][CAIDODBC DLL]Message text...  
[CA][CAIDODBC DLL][CA-IDMS]Message text...
```

The first type of message is generated by the CA-IDMS ODBC driver as a result of an error condition it detected. The second type of message is generated as a result of an error detected within the ODBC data source, which includes CAICCI, CA-IDMS, and the network components.

Pseudoconversational Processing

When accessing the mainframe through the ODBC interface, CA-IDMS Server will automatically cause the backend task to pseudoconverse. This occurs whenever the ODBC driver would normally issue an autocommit (even if the application had turned autocommit processing off). This means that the backend task will pseudoconverse:

- When a cursor is closed, provided no other cursors are open
- After execution of any command that updates the database, including INSERT, UPDATE, DELETE, and DDL

An ODBC application can cause the backend task to pseudoconverse at other times by executing a SUSPEND SESSION request. The session will be resumed automatically when the next ODBC request requiring mainframe access is issued.

Note: SUSPEND SESSION is not defined in the ODBC SQL grammar. Use of this command may restrict the portability of the application.

ODBC Conformance Levels

Microsoft ODBC documentation specifies ODBC conformance in two areas: ODBC API conformance and ODBC SQL conformance. A driver must support all functionality in a conformance level in order to claim conformance to that level, but is not restricted from supporting a subset of the functionality of higher levels. ODBC defines functions that allow an application to determine the functionality supported by a driver in detail, including the API and SQL conformance levels, as well as specific API function, data type, and scalar function support.

API Conformance Levels

The ODBC API includes three conformance levels:

- Core API supports a set of core functions in the X/Open and SAG CLI specification.
- Level 1 supports core functionality plus an extended set of functions.
- Level 2 supports core and Level 1 functionality plus an extended set of functions.

The three conformance levels are described below.

Core API

The Core API corresponds to the functions in the X/Open and SQL Access Group (SAG) CLI specification, with a few minor differences. The Core API provides the minimum services to support dynamic SQL, including connection establishment and termination, SQL statement execution, retrieval of results, and transaction control.

The CA-IDMS ODBC driver supports all Core API functions. The Core API functions are as follows:

Core Function	Comments
ConfigDSN	For use by configuration programs.
SQLAllocConnect	
SQLAllocEnv	
SQLAllocStmt	
SQLBindCol	
SQLCancel	
SQLColAttributes	
SQLConnect	
SQLDescribeCol	
SQLDisconnect	
SQLError	
SQLExecute	
SQLExecDirect	
SQLFetch	
SQLFreeConnect	
SQLFreeEnv	
SQLFreeStmt	
SQLGetCursorName	Supported but not needed because CA-IDMS Release 12.0 does not support WHERE CURRENT OF <i>cursor</i> for dynamic SQL (see below).
SQLNumResultCols	
SQLPrepare	
SQLRowCount	

Continued

Continued

Core Function	Comments
SQLSetCursorName	
SQLSetParam	Supported by ODBC Driver Manager for ODBC 1.0 applications only. Replaced by SQLBindParameter (see below).
SQLTransact	

Level 1 API

The Level 1 extension to the API supports the Core functionality and also allows the application to set connection and statement options, retrieve information about driver and data source capabilities, and retrieve catalog information.

The CA-IDMS ODBC driver supports all Level 1 API functions. The Level 1 API functions are as follows:

Level 1 Function	Comments
SQLBindParameter	Replaces SQLSetParam
SQLColumns	
SQLDriverConnect	
SQLGetConnectOption	
SQLGetFunctions	Implemented by ODBC driver manager
SQLGetInfo	
SQLGetStmtOption	
SQLGetTypeInfo	
SQLParamData	
SQLPutData	
SQLSetConnectOption	
SQLSetStmtOption	No asynchronous operation; can set timeout, but it is ignored
SQLSpecialColumns	CA-IDMS does not support auto update columns
SQLStatistics	
SQLGetData	
SQLTables	

Level 2 API

The Level 2 extension to the API supports the Core and Level 1 functionality and also allows scrollable bulk cursors, retrieval of input parameter descriptions, and retrieval of additional catalog information.

The CA-IDMS ODBC driver supports the following Level 2 API functions:

Level 2 Function	Comments
SQLParamOptions	
SQLMoreResults	

SQL Conformance Levels

The ODBC defines three conformance levels for SQL grammar:

- Minimum grammar supports minimal DDL, simple SELECT, INSERT, searched UPDATE and DELETE, simple expressions, and the CHAR data type.
- Core grammar supports the SQL statements and functionality proposed by the X/Open and SQL Access Group CAE specification (1992).
- Extended grammar supports common SQL extension supported by different database management systems.

The three conformance levels are described next.

Minimum SQL Grammar

The Minimum SQL Grammar is defined by ODBC to provide a basic level of conformance. It supports minimal DDL, simple SELECT, INSERT, searched UPDATE and DELETE, simple expressions, and the CHAR and VARCHAR data types.

CA-IDMS supports the Minimum SQL Grammar.

Integrity Enhancement Facility (IEF)

CA-IDMS Server supports referential integrity. Although CA-IDMS Server supports Integrity Enhancement Facility (IEF) functionality, it does not support the syntax defined in the ODBC specification, as noted in the following table. Implementation of IEF is optional for the ODBC driver and does not affect the SQL Conformance Level supported by the driver.

SQL Statement	Comments
CREATE TABLE clauses:	
DEFAULT	CA-IDMS supports WITH DEFAULT and allows default values of NULL, 0, or blank.
UNIQUE	CA-IDMS does not support specification of uniqueness constraints at either the column or table level. A unique index can be defined to provide the same effect.
PRIMARY KEY	CA-IDMS does not support specification of a primary key at either the column or table level. A unique index can be defined to provide the same effect.
REFERENCES	CA-IDMS does not support specification of referential constraints on the CREATE TABLE statement, at either the column or table level. The CREATE CONSTRAINT statement can be used to define referential constraints.
CHECK	CA-IDMS does not support specification of CHECK constraints at the column level. CHECK constraints can be specified at the table level.
DROP TABLE RESTRICT	CA-IDMS does not support the RESTRICT keyword, but does support CASCADE. The absence of CASCADE implies RESTRICT.
REVOKE CASCADE/RESTRICT	CA-IDMS does not support the CASCADE and RESTRICT options on REVOKE.

Core SQL Grammar

The Core SQL Grammar corresponds to the X/Open and SQL Access Group CAE draft specification (1991). Although similar to the ANSI SQL2 standard, it is not exactly the same. The Core SQL Grammar supports the Minimum SQL Grammar and adds more DDL, full SELECT, positioned UPDATE and DELETE, subquery, set expressions, and additional data types.

CA-IDMS supports the complete Core SQL Grammar.

The following ODBC SQL statements are converted by the CA-IDMS driver to CA-IDMS SQL:

SQL Statement	Comments
CREATE INDEX <i>user.index</i> DROP INDEX <i>user.index</i>	The CA-IDMS ODBC driver removes the qualifier from the index name if present.
GRANT UPDATE (<i>column-list</i>) REFERENCES (<i>column-list</i>)	CA-IDMS does not support column level security. The CA-IDMS driver removes the column list and grants UPDATE to all columns of the table.

Extended SQL Grammar

The Extended SQL Grammar supports the Minimum and Core SQL Grammar and adds outer joins, scalar functions, date/time literals, batch SQL statements, procedure calls, and additional data types.

CA-IDMS does not support the Extended SQL Grammar. Although CA-IDMS does support outer joins, it does not support the outer join syntax. CA-IDMS does not support procedure calls.

The ODBC driver supplied with CA-IDMS Server does not support outer joins and procedure calls. Scalar functions in escape sequences are passed to the data source unchanged except that the escape sequence itself is converted to spaces. SQL statements submitted in batch jobs are also not supported.

CA-IDMS supports data types that map to all ODBC data types.

The following statements specified in the ODBC Extended SQL syntax are not supported by the ODBC driver supplied with CA-IDMS Server Release 3.0:

Statement	Comments
ODBC procedure extension	CA-IDMS does not support procedures.
<i>outer join</i> LEFT OUTER JOIN	CA-IDMS supports outer join using the PRESERVE keyword, but the semantics are not identical.
<i>statement-list</i>	CA-IDMS and the CA-IDMS ODBC driver do not support this statement.
ODBC <i>scalar function extension</i>	CA-IDMS supports scalar functions that map to a subset of the ODBC scalar functions, but does not support the ODBC escape sequences for translation of scalar functions.

Continued

Continued

Statement	Comments
DELETE WHERE CURRENT OF <i>cursor</i>	CA-IDMS supports only in embedded SQL.
SELECT FOR UPDATE	CA-IDMS supports in DECLARE CURSOR, in embedded SQL.
UPDATE WHERE CURRENT OF <i>cursor</i>	CA-IDMS supports only in embedded SQL.

Database Type Mapping Between ODBC and CA-IDMS

The tables below describe how ODBC data types map to CA-IDMS database data types. The tables organize the data types by SQL conformance level. You can also use the SQLGetTypeInfo ODBC function to return detailed information about the mapping of ODBC and CA-IDMS data types.

Data Types

CA-IDMS to ODBC
Data Type Mapping

The following chart shows how CA-IDMS data types map to ODBC data types.

CA-IDMS Data Type	ODBC Data Type	Comments
BINARY	SQL_BINARY	
CHAR	SQL_CHAR	
CHARACTER VARYING	SQL_VARCHAR	Synonym for VARCHAR
DATE	SQL_DATE	
DECIMAL	SQL_DECIMAL	
DOUBLE PRECISION ¹	SQL_DOUBLE	
FLOAT ¹	SQL_FLOAT	
GRAPHIC ²	SQL_BINARY	If DBCS not enabled
INTEGER	SQL_INTEGER	
LONGINT	SQL_BIGINT	
NUMERIC	SQL_NUMERIC	

Continued

Continued

CA-IDMS Data Type	ODBC Data Type	Comments
REAL ¹	SQL_REAL	
SMALLINT	SQL_SMALLINT	
TIME	SQL_TIME	
TIMESTAMP	SQL_TIMESTAMP	
UNSIGNED DECIMAL	SQL_DECIMAL	
UNSIGNED NUMERIC	SQL_NUMERIC	
VARCHAR	SQL_VARCHAR	
VARGRAPHIC ²	SQL_VARBINARY	If DBCS not enabled Y

¹ Floating point conversion subject to rounding errors due to format differences.

² If DBCS processing is enabled, GRAPHIC is mapped to CAID_GRAPHIC; VARGRAPHIC is mapped to CAID_VARGRAPHIC. These are CA-IDMS driver-specific data types.

ODBC to CA-IDMS Data Type Mapping

The following chart shows how ODBC data types map to CA-IDMS data types.

ODBC Data Type	CA-IDMS Data Type
CAID_GRAPHIC ¹	GRAPHIC
CAID_VARGRAPHIC ¹	VARGRAPHIC
SQL_BINARY	BINARY
SQL_LONGVARBINARY	BINARY
SQL_CHAR	CHAR
SQL_DATE	DATE
SQL_DECIMAL	DECIMAL
SQL_DOUBLE	DOUBLE PRECISION
SQL_FLOAT ²	DOUBLE PRECISION
SQL_REAL ²	REAL
SQL_INTEGER	INTEGER

SQL_BIGINT

LONGINT

Continued

Continued

ODBC Data Type	CA-IDMS Data Type
SQL_NUMERIC	NUMERIC
SQL_BIT	SMALLINT
SQL_SMALLINT	SMALLINT
SQL_TINYINT	SMALLINT
SQL_TIME	TIME
SQL_TIMESTAMP	TIMESTAMP
SQL_LONGVARCHAR	VARCHAR
SQL_VARCHAR	VARCHAR

¹ Mapped when DBCS processing is enabled

² Floating point conversion subject to rounding errors due to format differences

Driver-Specific Data Types

When DBCS processing is enabled, the CA-IDMS GRAPHIC and VARGRAPHIC data types are mapped to driver-specific ODBC SQL data types, as allowed by the ODBC 2.10 specification. These types are defined as CAID_GRAPHIC and CAID_VARGRAPHIC in the CAIDOPT.H header file which is installed in the \CA_APPSW\CAID directory. These types will be returned by SQLColumns, SQLDescribeCol, and SQLColAttributes, and they should be used with SQLBindParameter to define input parameters for GRAPHIC and VARGRAPHIC columns.

Since most applications are not specifically designed to handle DBCS data as defined by CA-IDMS, these types are treated in the same manner as SQL_CHAR and SQL_VARCHAR. The default C type for both is SQL_C_CHAR, and the precision is specified in bytes. Note that on CA-IDMS the length is specified in DBCS characters, which is half the precision specified using the CA-IDMS ODBC driver.

When DBCS is not enabled, GRAPHIC and VARGRAPHIC are mapped to SQL_BINARY and SQL_VARBINARY, with a default C type of SQL_C_BINARY and precision equal to the length in bytes.

SQLDriverConnect Connection String Format

CA-IDMS supports additional keywords for the SQLDriverConnect connection string.

The connection string has the form:

DSN=data_source_name

[;attribute=value[;attribute=value]...]

Supported Attribute Keywords and Attribute Values

The table below provides a summary of the connection string attribute keywords and attribute values supported on the SQLDriverConnect function. This table includes both the keywords defined as part of the Microsoft ODBC specification and those defined as extensions for CA-IDMS Server. These keywords correspond to the fields in the DriverConnect dialog boxes as well as to the information used to define data sources and servers in the ODBC Administrator tool.

Keyword	Defined By	Attribute Value
DSN	Microsoft	Data source name
DRIVER	Microsoft	Driver name (cannot use with DSN)
DICT	CA	Dictionary name (use with DRIVER only)
NODE	CA	CA-IDMS System ID (use with DRIVER only)
TASK	CA	Alternate task code (use with DRIVER only)
UID	Microsoft	User ID
PWD	Microsoft	Password
ACCT	CA	Account information (if used)

Example

An example of a connection string for CA-IDMS Server is shown below.

DSN=CA-IDMS database; UID=JELKA01; PWD=XYZZY; ACCT=R45-87

For More Information

Refer to the Microsoft *ODBC Programmer's Reference* for more information about calling the SQLDriverConnect function. See Chapter 6, "Connecting to a Data Source," earlier in this manual, and Help for information about the DriverConnect dialog boxes. See Chapter 5, "Managing Data Sources," for more information about attribute values.

Driver-Specific Connect Options

The ODBC options that can be specified for a data source using the ODBC Configurator can also be specified during program execution using SQLSetConnectOption and SQLSetStmtOption. These options and their values are defined in CAIDOPT.H, installed in the \CA_APPSW\CAID directory.

Supported Isolation and Lock Levels

Transaction isolation is set with the SQLSetConnectOption ODBC API function. The default transaction isolation can be set using the ODBC Administrator (a CA-IDMS Server ODBC driver extension). The table below lists the transaction isolation levels that the CA-IDMS Server ODBC Driver supports and the corresponding CA-IDMS SET TRANSACTION options. The default is READ COMMITTED.

ODBC SQLConnect Option	CA-IDMS SET TRANSACTION Option
SQL_READ_UNCOMMITTED	TRANSIENT READ
SQL_READ_COMMITTED	CURSOR STABILITY

Bulk Insert Support

CA-IDMS Server supports the Core and Level 1 API functions listed in API Conformance Levels, earlier in this appendix. To facilitate Bulk Inserts, the CA-IDMS Server also supports these Level 2 functions:

- SQLParamOptions
- SQLMoreResults

To ensure that the ODBC driver takes advantage of the CA-IDMS INSERT...BULK feature, use parameter markers (?) in the Values clause of the Insert Into Tables statement; do not use a combination of parameter marks and constant values.

Retrieving Network Set Information

You can use the SQLExecuteDirect function with the following syntax to return information about network sets used to join network records accessed as SQL tables.

```
$SETS <owner> <table> <table>
```

where:

- *<owner>* is the name of the SQL schema that contains the names of the dictionary and network schema where the records are defined. This value applies to all tables and appears to the ODBC application as the TABLE_OWNER returned by SQLTables.
- *<table>* is the name of a record in the network schema. You can enter from zero to two *<table>* arguments. Each *<table>* argument must be unique and must be defined in the same network schema. This value appears to the ODBC application as the TABLE_NAME returned by SQLTables.

The *<owner>* and *<table>* name arguments are case-sensitive.

The following table identifies what the result set contains depending on what you specify for the *<table>* arguments:

If You Specify	The Result Set Contains
No <i><table></i> argument	A list of all sets in the network schema referenced by <i><owner></i>
One <i><table></i> argument	A list of all sets in the network schema referenced by <i><owner></i> in which <i><table></i> is either the owner or a member
Two <i><table></i> arguments	A list of all sets in the network schema referenced by <i><owner></i> between the two <i><table></i> s, where either is the owner or member

The result columns are:

Column Name	Data Type	Description
SET_NAME	VARCHAR(18)	Network set name
SCHEMA_NAME	VARCHAR(18)	SQL schema name (ODBC owner)
OWNER_NAME	VARCHAR(18)	Network owner record name (ODBC table)
MEMBER_NAME	VARCHAR(18)	Network owner record name (ODBC table)

COBOL Programmer Reference

Overview

Through the services of CA-IDMS Server, you can code and execute CA-Visual Realia COBOL applications on your PC that access data stored on a CA-IDMS database on the mainframe. This appendix describes:

- Designing and writing the application
- Precompiling the application
- Compiling and linking the application
- Executing the application

Designing and Writing the Application

This section describes what's involved with designing and writing a COBOL application that transmits data back and forth between your PC and a CA-IDMS system. It describes what types of requests your application can issue to CA-IDMS, how to start and end sessions on the mainframe, and how to convert transmitted data to a format understood by the receiving platform.

For more information about application design and other application components, see the documentation provided with CA-Visual Realia.

Issuing CA-IDMS Requests

A PC COBOL application can issue the following types of CA-IDMS requests:

- SQL
- Task management (COMMIT/FINISH/ROLLBACK TASK)

These requests have the same semantics as they do when embedded within a program executing on the mainframe, although there are some differences associated with determining which backend database or server to access. These differences are explained in *Initiating and Ending User and Service Sessions*, below. Refer to *CA-IDMS SQL Programming* and the *CA-IDMS DML Reference – COBOL* for more information about embedding CA-IDMS requests in COBOL programs.

The precompiler is used to transform the CA-IDMS requests into standard COBOL source code for subsequent compilation by a COBOL compiler. The COBOL code generated in place of a CA-IDMS request is the same as that in the mainframe environment except as noted below.

SQL Requests

The protocols used to issue SQL requests in a PC COBOL application differ from their mainframe counterparts in these ways:

- The called program name is P_IDMSSQL.
- An RPB program block is not passed on a request.
- The Server passes format descriptors as the last parameter on runtime calls. These are generated as data constants in the application program.
- The Protocol mode specified in the IDMS-CONTROL section has no influence over how the CA-IDMS request is expanded.

See Sample CA-Visual Realia COBOL Program, later in this appendix, for a sample CA-Visual Realia COBOL program that issues SQL requests against data in a CA-IDMS database.

Initiating and Ending User and Service Sessions

A *user session* is an association between a Windows application and CA-IDMS. A user session begins when the application issues its first CA-IDMS request and terminates when the application issues a signoff. There can only be one user session active at any one time within a single Windows application.

A *service session* is an association between a Windows application and a specific CA-IDMS service such as SQL or DC. Any number of service sessions can be active within a single user session and therefore, within a Windows application. Different service sessions may access different backend nodes within a CA-IDMS network.

Initiating a User Session

The first request that the application issues within a user session must be a call to IDMSUSER to identify the user to CA-IDMS. No service session can be initiated until the application issues an IDMSUSER call.

The IDMSUSER call does not initiate communication with a backend CA-IDMS Server. Instead, CA-IDMS Server caches the information passed on the IDMSUSER call within the PC so that it is available when a service session is initiated. If the application issues more than one IDMSUSER call, the information in the cache reflects the information passed on the most recent call.

The IDMSUSER call syntax is:

```
CALL 'P_IDMSUSER' USING  
      userid, password, account, zonedesign
```

where:

- *userid* is an 18-character variable containing the user ID on the CA-IDMS system.
- *password* is an eight-character variable containing the user's password.
- *account* is a 32-character variable containing accounting information for chargeback purposes. If it is not used, it should be filled with spaces.
- *zonedesign* is a one-byte field indicating how signed zoned decimals (S9(n) DISPLAY) should be represented. Enter the value 'R', which in CA-Visual Realia interprets x'3n' for positive and x'2n' for negative.

Terminating a User Session

An application terminates a user session by issuing a call to IDMSOFF. This is similar to issuing a signoff request within a CA-IDMS/DC environment.

IDMSOFF:

- Terminates all service sessions associated with the user session, if any are still active
- Rolls back all uncommitted database changes
- Frees all resources held for the user session

The syntax for the IDMSOFF call is:

```
CALL 'P_IDMSOFF'
```

Establishing a Service Session

An application initiates an SQL service session within the PC just as from within the mainframe. An SQL service session starts explicitly by issuing a CONNECT command or implicitly by issuing some other SQL command.

The first service session initiated on each backend node validates the user and password within that node. The user must have signon authority to the CA-IDMS system in order for the service session to be successfully initiated. Subsequent service sessions associated with the same backend node do not incur the overhead of validating the user ID.

When you initiate a service session from the PC, the target node or database is determined differently than when the service session is initiated on the mainframe. In the PC environment, CA-IDMS Server uses the data source definitions defined through the ODBC Administrator tool to determine the target resource.

Initiating an SQL Session

When an SQL session is started, CA-IDMS Server determines the target node and dictionary as follows:

1. If the application issues a `CONNECT` command, the dictionary name passed on the `CONNECT` is used as the name of a data source. Otherwise, the data source **IDMS DEFAULT** is used.
2. Once CA-IDMS Server determines the data source, it searches the `CAIDDSI.INI` file to locate a matching data source definition. If it does not find a match, the SQL request fails.
3. If CA-IDMS Server finds a matching data source definition, it determines the target node from the Server associated with the data source. It determines the target dictionary from either the dictionary associated with the data source, if one was specified, or the name of the data source.

Terminating a Service Session

CA-IDMS Server terminates a database service session by using the appropriate SQL command: `COMMIT [RELEASE]`, `ROLLBACK [RELEASE]`, and `RELEASE`.

Just as on the mainframe, `FINISH` and `ROLLBACK TASK` commands also terminate all database service sessions, except for suspended SQL sessions. If there are multiple service sessions active and you want to commit all database changes as a unit, issue either a `COMMIT` or `FINISH TASK` command. Issuing one of these commands guarantees that all service sessions associated with the same backend node are committed as a unit.

For more information about these commands, see the *CA-IDMS SQL Reference*.

Performance Considerations

The most important thing that affects how quickly CA-IDMS Server transmits data between the mainframe and the PC application is the number of communications between the PC and the mainframe. The time required to send and receive data far outweighs the time it takes to process the data on the PC. Consequently, you should try and minimize the number of service requests issued.

Minimizing Server Requests

This list itemizes some techniques you can use to reduce the number of communications with the Server:

- Save information from a previously retrieved table rather than retrieving the table each time the program needs it
- Use bulk access when the program needs to fetch or insert multiple rows
- Use table procedures to process complex structures such as bill of materials
- Use SQL capabilities such as table joins and Where clause selection to reduce the number of mainframe requests

Writing Pseudoconversational Applications

Another important performance consideration is designing your application so that it executes pseudoconversationally. Just as when writing an online application for DC or CICS, a program should hold as few resources as possible while waiting for the user to respond. One way to accomplish this from the PC is to suspend your SQL session at strategic points.

The following sequence of requests would be typical of a well-behaved application. Each time the application suspends an SQL session, it causes the backend task to pseudoconverse.

```
CONNECT ...      Validates the user and starts the SQL session

SQL DML request(s)  Retrieves and/or updates mainframe data

COMMIT           Terminates the database transaction, commits
                 updates and frees locks

SUSPEND SESSION Terminates the backend task
...
Wait for the user to respond.
...
SQL DML request(s) Starts a new backend task, starts a new
                 transaction and manipulates mainframe data

COMMIT           Terminates the transaction

SUSPEND SESSION Terminates the back-end task
...
Repeat as often as necessary
...
RELEASE         Terminates the SQL session
```

The application program keeps the SQL session open as long as it anticipates that additional mainframe access will be needed. This eliminates the overhead of establishing a new communications session and validating the user on each pseudoconverse. However, it terminates the database transaction each time it pseudoconverses so that as few resources (especially database locks) are held during the time the application is waiting for the user to respond. The SQL session is resumed automatically on the first SQL request following the SUSPEND.

Note: You may optionally code a RESUME SESSION to resume the session explicitly if desired.

IDMSIN01 Support

IDMSIN01 is a mainframe subroutine provided for the use of application programs. The following IDMSIN01 functions are applicable to CA-IDMS Server:

- Extracting error messages from the SQLCA
- Returning the current date and time in SQL TIMESTAMP format

The calling conventions for the PC version of IDMSIN01 are identical to those on the mainframe except that:

- The program name must be specified as **C_IDMSIN01**
- An RPB is not required and should not be passed as the first parameter

The following is an example of a call to IDMSIN01 to extract error messages from the SQLCA into a user-defined message area:

```
CALL 'C_IDMSIN01' USING REQ-WK, SQLCA, SQLMSG5.
```

Refer to *CA-IDMS SQL Programming* for a complete description of the IDMSIN01 calling arguments.

Precompiling the Application

After you design and write your program, you need to **precompile** it using a precompiler. The precompiler:

- Checks the syntax of the SQL statements in your application
- Replaces the SQL statements in your application with program language calls to CA-IDMS
- Executes precompiler directives

- Stores the precompiled code as a relational command module (RCM) if no errors occur

Note: The CA-IDMS precompiler on the PC ignores the protocol (for example, BATCH, DC, CICS), that you specify with the MODE parameter in the PROTOCOL portion of the IDMS-CONTROL SECTION compiler directive statement. On the mainframe, the precompiler uses the protocol mode to generate call statements for the program's PROCEDURE DIVISION DML statements; on the PC, this parameter does not affect how calls are expanded. However, the protocol mode does influence what version of a record or module to use. Refer to *CA-IDMS DML Reference – COBOL* and *CA-IDMS Navigational DML Programming* for more information about protocol modes.

To precompile your application, use CA-Visual Realia. The CA-Visual Realia documentation set provides detailed information about precompiling, compiling, linking, and executing applications using CA-Visual Realia. This section describes how to select the precompiler for CA-IDMS.

Defining the CA-IDMS Precompiler

To define the CA-IDMS precompiler to CA-Visual Realia, the system administrator should edit the precompiler definition file, PRECOMP.INI, the default, in the \CAVR directory. The CA-IDMS= line in the [Precompiler Groups] section should read:

```
CA-IDMS=Installed
```

Below is a portion of the PRECOMP.INI file:

```
[Precompiler Groups]
DL with SQL=Installed
Datacom/PC=Installed
Datacom/PC SQL=Installed
Auto Format=Installed
Testing=Installed
CA-IDMS=Installed

[Group CA-IDMS]
Step01=CA-IDMS

[Step CA-IDMS]
Environment=Windows
Process DLL=CAIDDMLC.DLL
Options DLL=CAIDDMLC.DLL
Help=CAIDDMLC.HLP
LinkLibraries=CAIDQCLI.LIB+CAIDDCLI.LIB
```

Selecting the CA-IDMS Precompiler

To select the CA-IDMS precompiler within CA-Visual Realia, select the Options Compile Defaults menu command. In the Category list box, select Preprocessors and then choose CA-IDMS as the program type.

Note: In the Program Options category, do *not* select the Perform SQL Access checkbox for the SQL Access option. When you select this checkbox, CA-Visual Realia uses a precompiler for ODBC.

Setting Precompiler Options

When you select the CA-IDMS program type in the Option Compile Defaults menu dialog, and then click the Change button, the CA-IDMS precompiler displays the CA-IDMS Precompiler Options dialog box that you can use to set precompiler options.

The dialog box prompts you to specify the name of the data source to connect to, your host user name and password, the names of the resource command module (RCM) and access module (AM) to create, what standard to use when the precompiler checks the syntax of your application's SQL statements, and date and time formats to use. For detailed information about the CA-IDMS Precompiler Options dialog box, see Help.

Codes Precompiler Error

If the precompiler detects an error during processing, it will display error information. Errors may be detected during execution of the SQL requests or the navigational DML requests (which the precompiler uses when processing such statements as COPY IDMS).

If an error occurs during execution of a navigational DML request, it will normally be represented as a standard error status as documented in the *CA-IDMS DML Reference – COBOL*. However, certain errors are unique to the PC environment and are represented by the following minor codes:

- 34 – Format descriptors could not be found. This is typically the result of an installation error. Format descriptors are installed into the directory specified during the installation process. The Path parameter in the Custom section of the CAIDDSI.INI file points to the format descriptors.
- 35 – An error occurred during the execution of a DTS/DNS request. Refer to the log file for more information.
- 99 – An error occurred during the execution of a Windows function request. Refer to the log file for more information.

Generally, whenever an error occurs, you should look in the log file for additional information to help resolve the problem.

Precompiler Input and Output Files

Input Files

The CA-IDMS Precompiler expects, for input, the source code for a COBOL application. The application can include embedded SQL statements.

Note: Your COBOL source statements can begin in column 1 or column 7, depending on the extension of the file name containing the source; an extension of **.CBL** indicates that source statements begin in column 1; an extension of **.COB** indicates that they begin in column 7.

Output Files

After the precompiler executes, it produces a file of the translated source code, named *program-name.001*. If errors exist, it also generates an error message listing, named *program-name.err*. If no errors occur during the compilation, CA-IDMS Server stores a relational command module (RCM) in the SQL dictionary of the data source. If you want to save time and resources, save the translated source code so that you do not have to precompile the program repeatedly; precompiling programs can be time-consuming.

Note: The first time you precompile an application, you must manually create the access module for the RCM in the SQL dictionary associated with the data source on the mainframe. Once you define the access module, CA-IDMS automatically rebuilds it each time you recompile the application. You can use the Online Command Facility (OCF) to create the access module. For information about access module syntax, see the *CA-IDMS SQL Reference*. For information about OCF, see the *CA-IDMS Command Facility*.

Linking the Application

When you develop an application, the CA-Visual Realia linker resolves external symbol references by searching import libraries, designated by file extension .LIB. The import libraries for CA-IDMS are CAIDQCLI.LIB, CAIDDCLI.LIB, and CAIDNCLI.LIB. These libraries resolve symbols such as IDMSSQL, IDMSUSER, and so on. By default, CA-IDMS Server stores these files in the C:\CA_APPSW\CAID directory during installation. To make these libraries accessible to CA-Visual Realia, you can either copy them into the \CAVR\LIB directory or specify the default directory in the Import Library (.LIB) Path entry in the CA-Visual Realia OPTIONS-PATH dialog. If you copy the libraries, be sure to recopy them whenever you install future releases of CA-IDMS Server.

Sample CA-Visual Realia COBOL Program

The sample CA-Visual Realia COBOL program below issues SQL requests against data in a CA-IDMS database. This program contains embedded SQL statements that fetch data from a CA-IDMS table definition. It also:

- Initiates a user session using a P_IDMSUSER call
- Terminates a user session using a P_IDMSOFF call

```
IDENTIFICATION DIVISION.
PROGRAM-ID.          FETSAMP1.
AUTHOR.             USER.
INSTALLATION.       COMPUTER ASSOCIATES INTERNATIONAL.
DATE-WRITTEN.       1/25/95.
DATE-COMPILED.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER.   IBM-PC-COMP.
OBJECT-COMPUTER.   IBM-PC-COMP.

DATA DIVISION.

WORKING-STORAGE SECTION.
01 first-time      pic 9 value 1.
01 user-name       pic x(18) value 1.
01 passwd          pic x(8) value spaces.
01 acct            pic x(32) value spaces.
01 zone            pic x value 'R'.

EXEC SQL BEGIN DECLARE SECTION END-EXEC .
exec sql include table demoempl.division
      as divrec end-exec.
EXEC SQL END DECLARE SECTION END-EXEC.

LINKAGE SECTION.
01 LINK-DATA.
   05 link-headid   pic 9(4) binary.
   05 link-name     pic x(40).
01 rec-count.      pic 9(4) binary.
01 linkcode        pic -----9.
01 linkcerc        pic -----9.
01 expect          pic x(20).
```

```

01 RESULT                                PIC S9(4) BINARY.

PROCEDURE DIVISION USING BY REFERENCE
LINK-DATA, rec-count, linkcode, linkcerc, expect,
GIVING RESULT.

MAIN-LOGIC.
IF FIRST-TIME = 1 THEN

    CALL 'P_IDMSUSER' USING
        user-name
        passwd
        acct
        zone

    EXEC SQL
        WHENEVER NOT FOUND
            goto CLOSE-CONNECTION
    END-EXEC
    EXEC SQL
        WHENEVER sqlerror
            goto sqlerr
    END-EXEC

    EXEC SQL
        WHENEVER sqlwarning continue
    END-EXEC
    EXEC SQL
        CONNECT TO SYSQA03
    END-EXEC
    MOVE 0 TO rec-count
    EXEC SQL
        DECLARE d1 CURSOR FOR
            SElect div-head-id, div_name
                from demoempl.division where div_code = 'D09'
    END-EXEC
    EXEC SQL OPEN d1 END-EXEC
    MOVE 0 TO FIRST-TIME
    move 1 to result
    move 'gets corporate' to expect
END-IF.
PERFORM READ-CRS-1.

GOBACK.

READ-CRS-1.
MOVE 0 TO FIRST-TIME.
EXEC SQL
    FETCH d1 INTO
        :div-head-id :div-head-id-i, :div-name
END-EXEC.
PERFORM MOVEDATA thru movdata-exit.

MOVEDATA.
    add 1 to rec-count.
    move div-head-id to link-headid.
    MOVE div-name to link-name.
    move sqlcode to linkcode.
    move sqlcerc to linkcerc.
MOVEDATA-EXIT.

CLOSE-CONNECTION.
MOVE 0 TO RESULT.
move sqlcode to linkcode.
EXEC SQL CLOSE D1 END-EXEC.

```

```
EXE SQL COMMIT RELEASE  END-EXEC.  
CALL 'P_IDMSOFF'.  
GOBACK.  
  
SQLERR.  
  MOVE 0 TO RESULT.  
  move sqlcode to linkcode.  
  move sqlcerc to linkcerc.  
EXEC SQL  
  WHENEVER sqlerror continue  
END-EXEC.  
EXEC SQL CLOSE d1      END-EXEC.  
EXEC SQL commit release  END-EXEC.  
call 'P_IDMSOFF'.  
GOBACK.
```

PC Client .INI Files

Overview

This appendix describes the ODBC.INI and CAIDDSI.INI files that are used by CA-IDMS Server and created by CA-IDMS Server ODBC Setup in the Microsoft ODBC Administrator application.

This information is provided to help you identify any problem that may arise. If you want to reconfigure the ODBC.INI or CAIDDSI.INI files, use CA-IDMS Server ODBC Setup. Making changes directly to these files may prevent your application from accessing CA-IDMS.

The ODBC.INI File

The Microsoft ODBC Administrator application creates or modifies the ODBC.INI file in the Windows directory when you add, modify, or remove data sources from CA-IDMS Server.

The ODBC.INI file lists the data sources in the [ODBC Data Sources] section and associates each data source with a driver.

Example

The ODBC.INI file created for the sample application used in this document appears below:

```
[ODBC Data Sources]
inventory data source=IDMS
benefits data source=IDMS

[inventory data source]
Driver=C:\WINDOWS\SYSTEM\caidodbc.dll
Description=Inventory database on system 81

[benefits data source]
Driver=C:\WINDOWS\SYSTEM\caidodbc.dll
Description=human resources db on system 82
```

Note: **Description is optional with ODBC 2.0.**

The CAIDDSI.INI File

The CAIDDSI.INI file provides information that CA-IDMS uses to establish a connection to a CA-IDMS database anywhere in a network of CA-IDMS systems. The ODBC Administrator application calls CA-IDMS Server ODBC Setup, which creates the CAIDDSI.INI file in the Windows directory. The CAIDDSI.INI file contains parameters for each data source you define as described in Chapter 5, “Managing Data Sources.”

The CAIDDSI.INI file can contain these sections:

- **Data Sources Section** to associate a data source name with a database driver
- **Data Source Name Section** for each defined data source
- **Servers Section** to associate a server name with a CA database server product name
- **Server Name Section** for each server to define its database access path information
- **Options Section** for the global overrides for default database settings, log settings, and language settings
- **Custom Section** describing information required by CA-ODBC setup
- **DBCS Types Section** describing the supported double-byte character sets

The Data Sources, Data Source Name, Servers, Server Name, Options, Custom, and DBCS Types sections are described next.

Data Sources Section

The Data Sources Section lists all data sources defined using the CA-IDMS Data Source Administration dialog box under the Microsoft ODBC Administrator application.

Syntax

```
[Data Sources]  
data_source_name=driver_name
```

Parameters

Data_source_name specifies the user-defined data source name.

Driver_name specifies the name of the CA-IDMS Server client interface driver.

Data Source Name Section

The CAIDDSI.INI file contains an entry for each data source you define in using the ODBC Administrator application and any database settings you defined for that data source. For information about the database settings parameters, see the Options Section later in this appendix.

Syntax

```
[data_source_name]  
Dictionary=dictionary_name  
Server=server_name
```

Parameters

Data_source_name identifies a data source. This value comes from the Data Source text box in the CA-IDMS Data Source Administration dialog box.

Dictionary_name identifies the DBNAME or segment name of the CA-IDMS dictionary as defined in the CA-IDMS DBNAME table on the target CA-IDMS system. This value comes from the Dictionary text box in the CA-IDMS Data Source Administration dialog box. If no dictionary name is specified, the default is the first eight characters of the data source name.

Server_name specifies the user-defined server name.

Servers Section

The Servers Section lists all servers defined using the CA-IDMS Data Source Administration dialog box under the Microsoft ODBC Administrator application.

Syntax

```
[Servers]  
server_name=driver_name
```

Parameters

Server_name specifies the user-defined server name.

Driver_name specifies the name of the CA-IDMS Server client interface driver.

Server Name Section

The CAIDDSI.INI file contains a Server Name Section for each server identified in a Data Source Name.

Syntax `[Server=server_name]
Resource=node_name
AlternateTask=task_code`

Parameters *Server_name* is a server name listed in the Servers section.

Node_name optionally identifies the value of SYSTEMID as specified in the SYSTEM statement of the system generation of the target system. If a node name is not specified, CA-IDMS Server uses the first eight characters of the *server_name* to identify the target system.

Task_code optionally identifies an alternate task that defines the resource limits and timeout values for a session. The default is CASERVER. If you accept the default or define another task, be sure it is defined as a task on the CA-IDMS System Generation TASK Statement. This value comes from the Task Code text box in the CA-IDMS Server Administration dialog box.

For more information on resource limits for external user sessions, see *CA-IDMS System Generation* and *CA-IDMS System Operations*.

Options Section

The Options Section contains parameters you selected as Administrator default overrides in the ODBC Administrator application. The CAIDDSI.INI file is updated with the values from the options you selected in the Set Administrator Defaults, Set Log Options, Set CECP Language, and Set DBCS Language dialog boxes accessed from the Options list in the CA-IDMS Data Source Administration dialog box.

Syntax

```

[Options]
; The following parameters
; are for global options
LogFile=log_file_name
LogOptions=log_option_values
DbcsType=dbcs_character_set
DbcsPath=dbcs_path
AsciiEdcDicTables=translation_table_name
OdbcTrace=integer_value
SQLTrace=integer_value
NdlTrace=integer_value
DtsTrace=integer_value
DnsTrace=integer_value
CmTrace=integer_value
FdeTrace=integer_value
UtilTrace=integer_value
; The following parameters
; are for default options
; for new data sources
FetchRows=number_of_rows
FetchDouble=0/1
ReadOnly=0/1
TxnIsolation=0/1
CommitBehavior=0/1/2
AccessibleTables=0/1
CatalogTable=view_name
EnableEnsure=0/1
CacheSQLTables=0/1

```

Parameters

LogFile=log_file_name specifies the name of the log file, if you entered something other than the default log name. This value comes from the Log File Name text box in the Log and Trace Options dialog box accessed from the Options list in the CA-IDMS Data Source Administration dialog box.

LogOptions=log_option_values specifies log options with the following meanings:

Value	Description
1	Appends logging information to the log file each time an ODBC session is started.
2	Closes the log file each time a write occurs to the log file. This is the default.
4	Keeps the log file open.
8	Creates a separate log for each task.

Options 1 and 2 come respectively from the Append and Close check boxes in the Log and Trace Options dialog box accessed from the Options list in the CA-IDMS Data Source Administration dialog box.

You can also specify a log option value that is the sum of individual log options. For example, value 3 indicates that both options 1 and 2 are enabled. Option 6, however, which is the sum of options 2 and 4, is not valid because option 4 overrides 2.

DbscType=*dbcs_character_set* specifies the numeric value that corresponds to the double-byte character set you selected to enable DBCS processing you selected in the Double Byte Character Set Options dialog box accessed from the Options list in the CA-IDMS Data Source Administration dialog box.

DbscPath=*dbcs_path* specifies the path that contains the double-byte character set conversion tables. This value comes from the Double Byte Character Set Options dialog box accessed from the Options list in the CA-IDMS Data Source Administration dialog box.

AsciiEbcDicTables=*translation_table_name* specifies the name of the translation table you selected to convert EBCDIC data on the server to ASCII data on the PC and vice versa. The value comes from the Country Extended Code Page Options dialog box accessed from the Options list in the CA-IDMS Data Source Administration dialog box.

xxxxTrace=*integer_value* specifies the flag bits used to control tracing. The Options section can also contain trace flags that Computer Associates Technical Support uses to resolve CA-IDMS Server problems. The *integer_value* must be in the range of 0, which signifies all options off, to 65535, which signifies all options on. This value can be specified as a decimal or hexadecimal integer. Descriptions of the individual bit flags are defined in the IDMS.TXT file, normally installed in the \CA_APPSW\CAID directory.

The ODBC configurator is normally used to enable and disable tracing. Since tracing can add overhead and affect performance, it should be disabled under normal circumstances.

FetchRows=*number_of_rows* specifies the number of database rows CA-IDMS Server should fetch at a time. The default is 100. This value comes from the Bulk Fetch Row Count text box in the Set Administrator Defaults dialog box accessed from the Options list in the CA-IDMS Data Source Administration dialog box.

FetchDouble specifies whether four-byte floating point numbers should be converted to eight-byte floating point before CA-IDMS Server fetches the data. The value 1 indicates that the option is on; the value 0 indicates that the option is off. This value comes from the Fetch Real as Double check box in the Set Administrator Defaults dialog box accessed from the Options list in the CA-IDMS Data Source Administration dialog box.

ReadOnly specifies whether the Windows application can both read and update the CA-IDMS database or only read it. The value 0 indicates that it can read and update (READ WRITE); the value 1 indicates that it can only read it (READ ONLY). This value comes from the Access Mode list box in the Set Administrator Defaults dialog box accessed from the Options list in the CA-IDMS Data Source Administration dialog box.

TxnIsolation specifies the degree to which your transactions are impacted by (and impact) other users accessing the same data. The value 1 indicates READ UNCOMMITTED; the value 2 indicates READ COMMITTED. The default is READ COMMITTED. This value comes from the Transaction Isolation list box in the Set Administrator Defaults dialog box accessed from the Options list in the CA-IDMS Data Source Administration dialog box.

CommitBehavior determines how a COMMIT operation affects cursors in CA-IDMS. The value 0 indicates Close and Delete Cursors, which closes and deletes cursors. The application must prepare and execute the next statement. The value 1 signifies Close Cursors, which closes cursors. Applications can execute a statement without preparing the statement again. The value 2 signifies Preserve Cursors, which preserves cursors in the same position as before the COMMIT operation. The application can execute or fetch without preparing the statement again. This value comes from the Commit Behavior list box in the Set Administrator Defaults dialog box accessed from the Options list in the CA-IDMS Data Source Administration dialog box.

AccessibleTables tells the ODBC driver whether to use the SYSCA.ACCESSIBLE_TABLES view (or another view defined by you) for the SQLTables function. The value 0 indicates do not use the view; the value 1 indicates use the view. This value comes from the Use Accessible Tables View Name check box in the Set Administrator Defaults dialog box accessed from the Options list in the CA-IDMS Data Source Administration dialog box.

CatalogTable=*view_name* specifies the name of the view to use for the SQLTables function if you specified something other than the default view name. This value comes from the Accessible Tables View Name text box in the Set Administrator Defaults dialog box accessed from the Options list in the CA-IDMS Data Source Administration dialog box.

EnableEnsure instructs the CA-IDMS Server to honor the ENSURE parameter of the SQLStatistics function call. This value comes from the Enable Ensure check box in the Set Administrator Defaults dialog box accessed from the Options list in the CA-IDMS Data Source Administration dialog box. The value 1 indicates that the option is on; the value 0 indicates that the option is off.

CacheSQLTables instructs CA-IDMS Server to cache the table list returned from an SQLTables call. This value comes from the Cache SQL Tables check box in the Set Administrator Defaults dialog box accessed from the Options list in the CA-IDMS Data Source Administration dialog box. The value 1 indicates that the option is on; the value 0 indicates that the option is off.

Custom Section

The Custom section contains information about the installation of CA-IDMS Server.

Syntax

```
[Custom]
Path=path_name
```

Parameters

Path=path_name specifies the directory where files used by CA-IDMS Server are installed. The default is *x*: \CA_APPSW\CAID, where *x*: identifies the disk drive. Note that the ODBC and CA-IDMS Server DLLs are installed in the Windows system directory.

DBCS Types Section

The DBCS Types Section contains a list of the double-byte character set options that CA-IDMS Server supports. If you enable DBCS processing, CA-IDMS Server searches the Options section for the value that corresponds to the type of processing you selected in the Double Byte Character Set Options dialog box accessed from the Options list in the CA-IDMS Data Source Administration dialog box.

Syntax

```
[DBCS Types]
dbcs_character_set=character_set_value
4261795
```

Parameter

Dbsc_character_set=character_set_value specifies the numeric value that corresponds to the double-byte character set you selected to enable DBCS processing you selected in the Double Byte Character Set Options dialog box accessed from the Options list in the CA-IDMS Data Source Administration dialog box.

Passing Accounting Information to CA-IDMS

Overview

This appendix describes how to use CA-IDMS Server to pass accounting information from the PC client to the backend CA-IDMS system.

Supplying Accounting Information

You can supply as many as 32 bytes of accounting information using one of the following methods:

- The ODBC DriverConnect dialog box
- A parameter on the call to IDMSUSER from a CA-Visual Realia application program

The first character of accounting information must be a space; otherwise, it will be ignored.

How Accounting Information Is Used

The accounting information is passed to the backend and is stored in an area accessible from the PTE for use by accounting and security exits. For example, the information could be used by either exit 4 or 5 to change header information in the statistics block described by #STRDS.

The following table describes how the accounting data can be found.

Field Name	Control Block	Comments
TCELTEA	TCE	Points to the LTE. Determine whether it is a CCI-related LTE (type is LTEBULK).
LTEPTEA	LTE	Points to the PTE.
PTELACCT	PTE	Points to a 32-byte accounting information area. This area will be binary zeros if no accounting information was passed from the PC or if the frontend is not a PC.

Example

Sample code for locating accounting information is shown below.

```

USING TCE,R9
  L   R5,TCELTEA           Get the LTE
USING LTE,R5
  CLI LTETYP,LTEBULK      Is this a BULK type
  BNE RETURN              No...Return
  L   R6,LTEPTEA          Get the PTE address
USING PTE,R6
  L   R7,PTELACCT         R7 points to 32-byte area
    
```

Installation Checklists

Overview

This appendix presents checklists of the software components that must be installed on the mainframe and the PC to make CA-IDMS Server fully functional.

Checklist for Mainframe Software Installed and Configured

The following software must be installed and configured on the mainframe:

- CA-IDMS Release 12.0 or higher, including the SQL option
- CA-IDMS Server
- CA90s Services components CAIENF and CAICCI
- CA-IDMS system generated with definitions for CCI line, LTERM/PTERM pairs, and CA-IDMS Server task
- CA-IDMS database and dictionary definitions

Checklist for PC Software Installed and Configured

Stand-Alone PC or
LAN Server

The following software must be installed and configured on a stand-alone PC or LAN server:

- Windows 3.1 or higher
- Emulator software or TCP/IP communications software
- CAICCI/PC component files downloaded from the mainframe to the Windows SYSTEM directory and configured

Stand-Alone PCs

The following software must be installed and configured on stand-alone PCs:

- CA-IDMS Server

Networked PCs
Running LAN Server
Windows

- CA-IDMS data source(s) added and configured

The following software must be installed and configured on networked PCs running LAN server Windows:

- CA-IDMS Server installed on each client workstation
- DLL and .HLP files present in LAN server Windows SYSTEM directory (copied from installation on a PC client)
- ODBC.INI, OBDCINST.INI, and CAIDDSI.INI files present in each client home directory

Index

A

Accessible tables view, 3-5, C-7

Accounting information
 passing to CA-IDMS, D-1
 sample code for locating, D-2

Adding data sources, 5-2

Alternate task code, defining, 3-3

API conformance levels, A-2
 Core API, A-3
 Level 1, A-4
 Level 2, A-5

Application
 designing, B-1 to B-6
 performance considerations, B-4
 trace options, 5-9
 writing pseudoconversational, B-5 to B-6

ASCII data
 code pages for, 5-13
 translating, 5-11 to 5-15

B

Bulk Fetch Row Count database option
 and performance, B-5
 in CAIDDSI.INI file, C-6

C

CA90s Services
 setting up, 3-1
 software requirements, 2-3

CacheSQLTables parameter, in CAIDDSI.INI file, C-7

CAICCI/PC
 defining to a CA-IDMS system, 3-2
 documentation, 1-5
 installing, 2-2
 interface, 3-1

mainframe requirements, 2-3
setup for, 5-11
trace options, 5-9
use by CA-IDMS Server, 3-1

CAIDDSI.INI file, 4-2
 Custom Section, C-8
 Data Source Name Section, C-3
 Data Sources Section, C-2
 DBCS Types Section, C-8
 Options Section, C-4
 section contents, C-2
 Server Name Section, C-4
 Servers Section, C-2, C-3

CAIDINST.EXE file, 4-4

CA-IDMS
 data type mapping and ODBC, A-8
 defining, 3-2
 DriverConnect dialog box, 6-1, 6-2, 6-3
 isolation and lock levels, A-12
 service session, ending, B-4
 software requirements, 2-3, E-1

CA-IDMS Data Source Administrator tool data
source, defaults, 5-7

CA-IDMS request
 calling protocols, B-2
 used in application, B-1

CA-IDMS Server
 Administration dialog box, 5-10
 components of, 1-2
 example network, 3-2
 features of, 1-1
 hardware and software requirements, 2-1
 installation prerequisites, 4-1
 installing, 3-1, 4-2, E-1
 language options, 5-11 to 5-15
 overview of, 1-1

CAIENF facility, 2-3, 3-1

CALC key and SQL access, 3-8

CASERVER task code, 3-3, 4-1, 5-10

CatalogTable parameter, in CAIDDSI.INI file, C-7

CCI line, defining, 3-2

CCIDUMPA.TRC trace file, 5-9

CCILINE in system generation, 3-2

CCITRACE.EXE file, 5-9
CECP translation table, 5-13
Character data, translating, 5-11 to 5-15
Checklists, for installation, E-1
COBOL application. *See* Application
Code page
 for host, 5-13
 for PC, 5-13
CommitBehavior database option, C-7
Communication protocols, supported, 3-1
Conformance levels, A-2
 API, A-2
Control panel icons, 4-2
Core API conformance level, A-3
 supported functions, A-3
Core SQL Grammar conformance level, A-6
Country Extended Code Page
 for host languages, 5-13
 for PC languages, 5-13
 table in CAIDDSI.INI file, C-6
Creating translation tables, 5-12
Custom Section, in CAIDDSI.INI file, C-8

D

Data source
 adding, 5-2
 connecting to, 6-1 to 6-3
 defaults, 5-7
 maintaining, 5-5
 options, 5-4, 5-7 to 5-8
 testing connection for, 5-5
Data Source Administration dialog box, 5-8
Data Source Name Section, in CAIDDSI.INI file, C-3
Data Sources Section, C-2
Data type mapping, A-8 to A-10, A-8
Data types, CA-IDMS, A-8
Data, converting
 floating point numbers, C-6
 language options, 5-11 to 5-15, C-6
 performance considerations, B-4

Database access
 and page groups, 3-5
 setting up for, 3-4
DBCS
 parameters in CAIDDSI.INI file, C-6, C-8
 processing, enabling, 5-14 to 5-15
 Types Section in CAIDDSI.INI file, C-8
Debugging user sessions, A-1
 error messages, A-1
Default
 data source options, 5-7, C-4
 resource name, 5-10
 task code, 5-10
Defining
 CA-IDMS host systems, 3-2
 CASERVER task code, 3-3
 CCI line, 3-2
 data sources, using ODBC, 5-2
Designing applications, B-1 to B-6
Dictionary name, in CAIDDSI.INI file, C-3
DML request. *See* Navigational DML database access
Double-byte character set processing. *See* DBCS

E

EBCDIC data
 code pages for, 5-13
 translating, 5-11 to 5-15
Editing translation tables, 5-14
Enable Ensure database option, C-7
Ending
 service sessions, B-4
 user sessions, B-3
Error messages, logging, 5-9
Event Notification Facility, 3-1
Example system network, 3-2
Extended SQL Grammar conformance level, A-7
EXTERNAL WAIT parameter, on Task statement, 3-4

F

Fetch Real as Double database option, C-6
FetchRows parameter, in CAIDDSI.INI file, C-6
FILLER record elements, 3-7
FIXED OCCURS element definitions, 3-7
Floating point number, conversion, C-6
Format descriptors, B-8
Functions
 Core API, A-3
 Level 1 API, A-4
 Level 2 API, A-5

G

Group record elements, 3-7

H

Hardware requirements, personal computer, 2-2
Host code page, 5-13
Hyphens
 in record and set names, 3-6
 in record element names, 3-7

I

Icons, in Control Panel, 4-2
IDMSIN01 support, B-6
IDMSOFF user session call, B-3
IDMSUSER user session call, B-2
INACTIVE INTERVAL parameter, on Task statement, 3-4
Index key and SQL access, 3-8
INI files, 4-2
 CAIDDSI.INI file, C-2
 ODBC.INI file, C-1
Installation checklists, E-1

Installing CA-IDMS Server
 CAICCI/PC, 2-2
 hardware and software requirements, 2-1
 in a networked Windows environment, 2-2, 4-3
 on the mainframe, 3-1
 on the PC, 4-2
 prerequisites, 2-3, 4-1
Integrity Enhancement Facility (IEF), A-5
INTERNAL parameter, on Task statement, 3-4
Isolation level, CA-IDMS, A-12

L

LAN environment, installing CA-IDMS Server in, 2-2, 4-3, E-1
Level 1 API conformance level, supported functions, A-4
Level 2 API conformance level, supported functions, A-5
Line, CCI, 3-2
Linking applications, B-9
Lock level, CA-IDMS, A-12
Log file, C-5
 for application and CCI, 5-9
 to debug user sessions, A-1
 writing trace information to, 5-8
LogFile parameter, in CAIDDSI.INI file, C-5
Logging
 error messages, 5-9
 options, C-5
LogOptions parameter, in CAIDDSI.INI file, C-5
LU2 communication protocol
 and CA90s Services, 3-1
 and tracing, 5-9

M

Maintaining data sources, 5-5
Messages, in log file, A-1
Minimizing server requests, B-4
Minimum SQL Grammar conformance level, A-5

N

- Navigational DML database access
 - setting up for, 3-5
 - using in an application, B-1
 - using SQL statements, 3-6
- Navigational DML service session, ending, B-4
- Network Windows environment, installing in, 4-3
- Non-SQL database record
 - accessing using SQL, 3-6
 - element name
 - length, 3-7
 - transforming, 3-7
 - name, transforming, 3-6
- Non-SQL database set name, transforming, 3-6

O

- OCCURS DEPENDING ON record elements, 3-7
- ODBC Administrator tool
 - data source
 - adding, 5-2
 - maintaining, 5-5
 - options, 5-4, 5-7 to 5-8
 - testing connections, 5-5
 - icon, 4-2
 - starting, 4-2
- ODBC driver
 - conformance levels, A-2
 - data type mapping, A-8
 - overview of, A-1
 - supported API functions, A-3, A-4
- ODBC.INI file, 4-2, C-1
- ODBCINST.INI file, 4-2
- Options Section
 - in CAIDDSI.INI file, C-4
 - specifying default options for new data sources in, C-5
 - specifying global options in, C-5
- Options, for data sources, 5-4
 - default, 5-7
 - in CAIDDSI.INI file, C-4
 - setting, 5-7 to 5-8

P

- Page groups, and database access, 3-5
- Performance considerations
 - for Cache SQL Tables option, 5-7
 - for data conversion, B-4
 - for Enable Ensure option, 5-7
 - for minimizing server requests, B-4
 - for Use Accessible Table View option, 5-7
- Personal computer
 - code pages, 5-13
 - hardware requirements, 2-2
 - software requirements, 2-2, E-1
- Precompiling applications
 - error codes, B-8
 - protocols, B-2
- Protocol
 - communication, 3-1
 - IDMSOFF user session call, B-3
 - IDMSUSER user session call, B-2
 - used by precompiler, B-2
- Pseudoconversational
 - applications, writing, B-5
 - processing, A-2

R

- RAM requirement, 2-2
- Read Committed/Uncommitted option, C-7
- READ ONLY/WRITE access mode, C-7
- ReadOnly parameter, in CAIDDSI.INI file, C-7
- Record
 - accessing using SQL, 3-6
 - name, transforming, 3-6
- Record element name
 - length, 3-7
 - transforming, 3-7
- REDEFINES clause
 - and access by SQL, 3-7
 - with control keys, 3-8
- Resource name
 - defining, 5-10
- RHDCNP3S task code, 3-3

Row, bulk fetch, C-6

S

Sample CA-Visual Realia COBOL program, B-10

Saving translation tables, 5-14

Server

- setting up, 5-9 to 5-11
- task code, 5-10

Server Name Section, in CAIDDSI.INI file, C-4

Servers Section, in CAIDDSI.INI file, C-2, C-3

Service session. *See also* CA-IDMS, Navigational DML

service session, SQL service session

- definition, B-2
- ending, B-4
- starting, B-4
- types, B-3

Set name, transforming, 3-6

Set up for

- CA90s Services, 3-1
- CAICCI/PC, 5-11
- CA-IDMS Server on the mainframe, 2-3
- CASERVER task, 4-1
- database access, 3-4
- servers, 5-9 to 5-11
- SQL to non-SQL database access, 3-6

Setting default data source options, 5-7

Software requirements

- CA90s Services, 2-3
- CA-IDMS, 2-3, E-1
- personal computer, 2-2, E-1

Sort key and SQL access, 3-8

SQL conformance levels

- Core SQL Grammar, A-6
- Extended SQL Grammar, A-7
- Minimum SQL Grammar, A-5

SQL Data Sources dialog box, 6-1, 6-2

SQL database access

- sample CA-Visual Realia COBOL program, D-1
- setting up for, 3-5, 3-6
- used in application, B-2

SQL service session

- ending**, B-4
- starting, B-4

SQL synonym, for non-SQL records, 3-7

Starting

- ODBC Administrator tool, 4-2
- service sessions, B-4

SYSCA.ACCESSIBLE_TABLES view, 3-5, C-7

T

Tables, accessible, 3-5, C-7

Task code

- defining, 3-3
- for set up, 4-1
- in CAIDDSI.INI file, C-4
- specifying, 5-10

TCP/IP communication protocol, and CA90s Services, 3-1

Testing connections, for ODBC data sources, 5-5

Trace flags, in CAIDDSI.INI file Options section, C-6

Trace options

- for applications, 5-9
- for CAICCI/PC, 5-9

Tracing, session tasks, 5-8

Transaction isolation levels, A-12

Translation editor tool, 5-12

- keystrokes, 5-14

Translation table, 5-12 to 5-14, C-5 to C-6

- CECP, 5-13
- customizing, 5-13
- DBCS processing, 5-15
- for character-based (DBCS) languages, 5-14 to 5-15
- name in CAIDDSI.INI file, C-6
- saving, 5-14

TxnIsolation parameter, in CAIDDSI.INI file, C-7

U

Use Accessible Table View option, 3-5, C-7

User session

- calls, B-2
- debugging, A-1
- definition, B-2

ending, B-3

V

Via Node attribute, 5-10

View, SYSCA_ACCESSIBLE_TABLES, 3-5, C-7

W

Workstation installation, 4-3

Writing

 applications, B-1 to B-6

 pseudoconversational applications, B-5