TIBCO Service Gateway™ for IDMS/DB

Installing and Operating

Software Release 6.0
July 2012

two-second advantage™

TIBCO®
The Power of Now®

Important Information

SOME TIBCO SOFTWARE EMBEDS OR BUNDLES OTHER TIBCO SOFTWARE. USE OF SUCH EMBEDDED OR BUNDLED TIBCO SOFTWARE IS SOLELY TO ENABLE THE FUNCTIONALITY (OR PROVIDE LIMITED ADD-ON FUNCTIONALITY) OF THE LICENSED TIBCO SOFTWARE. THE EMBEDDED OR BUNDLED SOFTWARE IS NOT LICENSED TO BE USED OR ACCESSED BY ANY OTHER TIBCO SOFTWARE OR FOR ANY OTHER PURPOSE.

USE OF TIBCO SOFTWARE AND THIS DOCUMENT IS SUBJECT TO THE TERMS AND CONDITIONS OF A LICENSE AGREEMENT FOUND IN EITHER A SEPARATELY EXECUTED SOFTWARE LICENSE AGREEMENT, OR, IF THERE IS NO SUCH SEPARATE AGREEMENT, THE CLICKWRAP END USER LICENSE AGREEMENT WHICH IS DISPLAYED DURING DOWNLOAD OR INSTALLATION OF THE SOFTWARE (AND WHICH IS DUPLICATED IN THE LICENSE FILE) OR IF THERE IS NO SUCH SOFTWARE LICENSE AGREEMENT OR CLICKWRAP END USER LICENSE AGREEMENT, THE LICENSE(S) LOCATED IN THE "LICENSE" FILE(S) OF THE SOFTWARE. USE OF THIS DOCUMENT IS SUBJECT TO THOSE TERMS AND CONDITIONS, AND YOUR USE HEREOF SHALL CONSTITUTE ACCEPTANCE OF AND AN AGREEMENT TO BE BOUND BY THE SAME.

This document contains confidential information that is subject to U.S. and international copyright laws and treaties. No part of this document may be reproduced in any form without the written authorization of TIBCO Software Inc.

TIBCO, The Power of Now, TIBCO Object Service Broker, and and TIBCO Service Gateway are either registered trademarks or trademarks of TIBCO Software Inc. in the United States and/or other countries.

All other product and company names and marks mentioned in this document are the property of their respective owners and are mentioned for identification purposes only.

THIS SOFTWARE MAY BE AVAILABLE ON MULTIPLE OPERATING SYSTEMS. HOWEVER, NOT ALL OPERATING SYSTEM PLATFORMS FOR A SPECIFIC SOFTWARE VERSION ARE RELEASED AT THE SAME TIME. SEE THE README FILE FOR THE AVAILABILITY OF THIS SOFTWARE VERSION ON A SPECIFIC OPERATING SYSTEM PLATFORM.

THIS DOCUMENT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

THIS DOCUMENT COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THIS DOCUMENT. TIBCO SOFTWARE INC. MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S) AND/OR THE PROGRAM(S) DESCRIBED IN THIS DOCUMENT AT ANY TIME.

THE CONTENTS OF THIS DOCUMENT MAY BE MODIFIED AND/OR QUALIFIED, DIRECTLY OR INDIRECTLY, BY OTHER DOCUMENTATION WHICH ACCOMPANIES THIS SOFTWARE, INCLUDING BUT NOT LIMITED TO ANY RELEASE NOTES AND "READ ME" FILES.

The TIBCO Object Service Broker technologies described herein are protected under the following patent numbers:

Australia:	-	-	671137	671138	673682	646408
Canada:	2284250	-	-	2284245	2284248	2066724
Europe:	-	-	0588446	0588445	0588447	0489861
Japan:	-	-	-	-	-	2-513420
USA:	5584026	5586329	5586330	5594899	5596752	5682535

Contents

Preface

This software may be available on multiple operating systems. However, not all operating system platforms for a specific software version are released at the same time. Please see the readme file for the availability of this software version on a specific operating system platform.

TIBCO® Object Service Broker is an application development environment and integration broker that bridges legacy and non-legacy applications and data. You can use TIBCO Object Service Broker to access external CA-IDMS data and define TIBCO Object Service Broker tables based on this data. This manual describes the TIBCO Object Service Broker interface to CA-IDMS data; the interface is known as the TIBCO Service Gateway for IDMS/DB.

Topics

Related Documentation

This section lists documentation resources you may find useful.

TIBCO Object Service Broker Documentation

The following documents form the TIBCO Object Service Broker documentation set:

Fundamental Information

The following manuals provide fundamental information about TIBCO Object Service Broker:

- *TIBCO Object Service Broker Getting Started* Provides the basic concepts and principles of TIBCO Object Service Broker and introduces its components and capabilities. It also describes how to use the default developer's workbench and includes a basic tutorial of how to build an application using the product. A product glossary is also included in the manual.
- *TIBCO Object Service Broker Messages with Identifiers* Provides a listing of the TIBCO Object Service Broker messages that are issued with alphanumeric identifiers. The description of each message includes the source and explanation of the message and recommended action to take.
- *TIBCO Object Service Broker Messages without Identifiers* Provides a listing of the TIBCO Object Service Broker messages that are issued without a message identifier. These messages use the percent symbol (%) or the number symbol (#) to represent such variable information as a rules name or the number of occurrences in a table. The description of each message includes the source and explanation of the message and recommended action to take.
- *TIBCO Object Service Broker Quick Reference* Presents summary information for use in the TIBCO Object Service Broker application development environment.
- *TIBCO Object Service Broker Shareable Tools* Lists and describes the TIBCO Object Service Broker shareable tools. Shareable tools are programs supplied with TIBCO Object Service Broker that facilitate rules language programming and application development.
- *TIBCO Object Service Broker Release Notes* Read the release notes for a list of new and changed features. This document also contains lists of known issues and closed issues for this release.

Application Development and Management

The following manuals provide information about application development and management:

- *TIBCO Object Service Broker Application Administration* Provides information required to administer the TIBCO Object Service Broker application development environment. It describes how to use the administrator's workbench, set up the development environment, and optimize access to the database. It also describes how to manage the Pagestore, which is the native TIBCO Object Service Broker data store.
- *TIBCO Object Service Broker Managing Data* Describes how to define, manipulate, and manage data required for a TIBCO Object Service Broker application.
- *TIBCO Object Service Broker Managing External Data* Describes the TIBCO Object Service Broker interface to external files (not data in external databases) and describes how to define TIBCO Object Service Broker tables based on these files and how to access their data.
- *TIBCO Object Service Broker National Language Support* Provides information about implementing the National Language Support in a TIBCO Object Service Broker environment.
- *TIBCO Object Service Broker Object Integration Gateway* Provides information about installing and using the Object Integration Gateway which is the interface for TIBCO Object Service Broker to XML, J2EE, .NET and COM.
- *TIBCO Object Service Broker for Open Systems External Environments* Provides information on interfacing TIBCO Object Service Broker with the Windows and Solaris environments. It includes how to use SDK (C/C++) and SDK (Java) to access TIBCO Object Service Broker data, how to interface to TIBCO Enterprise Messaging Service (EMS), how to use the TIBCO Service Gateway for WMQ, how to use the Adapter for JDBC-ODBC, and how to access programs written in external programming languages from within TIBCO Object Service Broker.
- *TIBCO Object Service Broker for z/OS External Environments* Provides information on interfacing TIBCO Object Service Broker to various external environments within a TIBCO Object Service Broker z/OS environment. It also includes information on how to access TIBCO Object Service Broker from different terminal managers, how to write programs in external programming languages to access TIBCO Object Service Broker data, how to interface to TIBCO Enterprise Messaging Service (EMS), how to use the TIBCO Service Gateway for WMQ, and how to access programs written in external programming languages from within TIBCO Object Service Broker.

- *TIBCO Object Service Broker Parameters* Lists the TIBCO Object Service Broker Execution Environment and Data Object Broker parameters and describes their usage.
- *TIBCO Object Service Broker Programming in Rules* Explains how to use the TIBCO Object Service Broker rules language to create and modify application code. The rules language is the programming language used to access the TIBCO Object Service Broker database and create applications. The manual also explains how to edit, execute, and debug rules.
- *TIBCO Object Service Broker Managing Deployment* Describes how to submit, maintain, and manage promotion requests in the TIBCO Object Service Broker application development environment.
- *TIBCO Object Service Broker Defining Reports* Explains how to create both simple and complex reports using the reporting tools provided with TIBCO Object Service Broker. It explains how to create reports with simple features using the Report Generator and how to create reports with more complex features using the Report Definer.
- *TIBCO Object Service Broker Managing Security* Describes how to set up, use, and administer the security required for an TIBCO Object Service Broker application development environment.
- *TIBCO Object Service Broker Defining Screens and Menus* Provides the basic information to define screens, screen tables, and menus using TIBCO Object Service Broker facilities.
- *TIBCO Service Gateway for Files SDK* Describes how to use the SDK provided with the TIBCO Service Gateway for Files to create applications to access Adabas, CA Datacom, and VSAM LDS data.

System Administration on the z/OS Platform

The following manuals describe system administration on the z/OS platform:

- *TIBCO Object Service Broker for z/OS Installing and Operating* Describes how to install, migrate, update, maintain, and operate TIBCO Object Service Broker in a z/OS environment. It also describes the Execution Environment and Data Object Broker parameters used by TIBCO Object Service Broker.
- *TIBCO Object Service Broker for z/OS Managing Backup and Recovery* Explains the backup and recovery features of OSB for z/OS. It describes the key components of TIBCO Object Service Broker systems and describes how you can back up your data and recover from errors. You can use this information, along with assistance from TIBCO Support, to develop the best customized solution for your unique backup and recovery requirements.

- *TIBCO Object Service Broker for z/OS Monitoring Performance* Explains how to obtain and analyze performance statistics using TIBCO Object Service Broker tools and SMF records
- *TIBCO Object Service Broker for z/OS Utilities* Contains an alphabetically ordered listing of TIBCO Object Service Broker utilities for z/OS systems. These are TIBCO Object Service Broker administrator utilities that are typically run with JCL.

System Administration on Open Systems

The following manuals describe system administration on open systems such as Windows or UNIX:

- *TIBCO Object Service Broker for Open Systems Installing and Operating* Describes how to install, migrate, update, maintain, and operate TIBCO Object Service Broker in Windows and Solaris environments.
- *TIBCO Object Service Broker for Open Systems Managing Backup and Recovery* Explains the backup and recovery features of TIBCO Object Service Broker for Open Systems. It describes the key components of a TIBCO Object Service Broker system and describes how to back up your data and recover from errors. Use this information to develop a customized solution for your unique backup and recovery requirements.
- *TIBCO Object Service Broker for Open Systems Utilities* Contains an alphabetically ordered listing of TIBCO Object Service Broker utilities for Windows and Solaris systems. These TIBCO Object Service Broker administrator utilities are typically executed from the command line.

External Database Gateways

The following manuals describe external database gateways:

- *TIBCO Service Gateway for DB2 Installing and Operating* Describes the TIBCO Object Service Broker interface to DB2 data. Using this interface, you can access external DB2 data and define TIBCO Object Service Broker tables based on this data.
- *TIBCO Service Gateway for IDMS/DB Installing and Operating* Describes the TIBCO Object Service Broker interface to CA-IDMS data. Using this interface, you can access external CA-IDMS data and define TIBCO Object Service Broker tables based on this data.
- *TIBCO Service Gateway for IMS/DB Installing and Operating* Describes the TIBCO Object Service Broker interface to IMS/DB and DB2 data. Using this interface, you can access external IMS data and define TIBCO Object Service Broker tables based on it.

- *TIBCO Service Gateway for ODBC and for Oracle Installing and Operating*
  Describes the TIBCO Object Service Broker ODBC Gateway and the TIBCO Object Service Broker Oracle Gateway interfaces to external DBMS data. Using this interface, you can access external DBMS data and define TIBCO Object Service Broker tables based on this data.

Typographical Conventions

The following typographical conventions are used in this manual.

Table 1 General Typographical Conventions

Convention	Use
TIBCO_HOME *OSB_HOME*	By default, all TIBCO products are installed into a folder referenced in the documentation as *TIBCO_HOME*. On open systems, TIBCO Object Service Broker installs by default into a directory within *TIBCO_HOME*. This directory is referenced in documentation as *OSB_HOME*. The default value of *OSB_HOME* depends on the operating system. For example on Windows systems, the default value is `C:\tibco\OSB`. Similarly, all TIBCO Service Gateways on open systems install by default into a directory in *TIBCO_HOME*. For example on Windows systems, the default value is `C:\tibco\OSBgateways\6.0`. On z/OS, no default installation directories exist.
`code font`	Code font identifies commands, code examples, filenames, pathnames, and output displayed in a command window. For example: Use `MyCommand` to start the foo process.
`bold code font`	Bold code font is used in the following ways: • In procedures, to indicate what a user types. For example: Type **`admin`**. • In large code samples, to indicate the parts of the sample that are of particular interest. • In command syntax, to indicate the default parameter for a command. For example, if no parameter is specified, `MyCommand` is enabled: `MyCommand [`**`enable`**` | disable]`
italic font	Italic font is used in the following ways: • To indicate a document title. For example: See *TIBCO ActiveMatrix BusinessWorks Concepts*. • To introduce new terms For example: A portal page may contain several portlets. *Portlets* are mini-applications that run in a portal. • To indicate a variable in a command or code syntax that you must replace. For example: `MyCommand` *PathName*

Table 1 General Typographical Conventions (Cont'd)

Convention	Use
Key combinations	Key name separated by a plus sign indicate keys pressed simultaneously. For example: Ctrl+C. Key names separated by a comma and space indicate keys pressed one after the other. For example: Esc, Ctrl+Q.
	The note icon indicates information that is of special interest or importance, for example, an additional action required only in certain circumstances.
	The tip icon indicates an idea that could be useful, for example, a way to apply the information provided in the current section to achieve a specific result.
	The warning icon indicates the potential for a damaging situation, for example, data loss or corruption if certain steps are taken or not taken.

Table 2 Syntax Typographical Conventions

Convention	Use
`[ ]`	An optional item in a command or code syntax. For example: `MyCommand [optional_parameter] required_parameter`
`|`	A logical `OR` that separates multiple items of which only one may be chosen. For example, you can select only one of the following parameters: `MyCommand para1 | param2 | param3`

Table 2 Syntax Typographical Conventions

Convention	Use

`{ }`	A logical group of items in a command. Other syntax notations may appear within each logical group.
	
	For example, the following command requires two parameters, which can be either the pair `param1` and `param2`, or the pair `param3` and `param4`.
	
	`MyCommand {param1 param2} | {param3 param4}`
	
	In the next example, the command requires two parameters. The first parameter can be either `param1` or `param2` and the second can be either `param3` or `param4`:
	
	`MyCommand {param1 | param2} {param3 | param4}`
	
	In the next example, the command can accept either two or three parameters. The first parameter must be `param1`. You can optionally include `param2` as the second parameter. And the last parameter is either `param3` or `param4`.
	
	`MyCommand param1 [param2] {param3 | param4}`

Connecting with TIBCO Resources

How to Join TIBCOmmunity

TIBCOmmunity is an online destination for TIBCO customers, partners, and resident experts, a place to share and access the collective experience of the TIBCO community. TIBCOmmunity offers forums, blogs, and access to a variety of resources. To register, go to http://www.tibcommunity.com.

How to Access All TIBCO Documentation

You can access TIBCO documentation here:

http://docs.tibco.com

How to Contact TIBCO Support

For comments or problems with this manual or the software it addresses, please contact TIBCO Support as follows.

- For an overview of TIBCO Support, and information about getting started with TIBCO Support, visit this site:

  http://www.tibco.com/services/support

- If you already have a valid maintenance or support contract, visit this site:

  https://support.tibco.com

  Entry to this site requires a user name and password. If you do not have a user name, you can request one.

Chapter 1 Installing Service Gateway for IDMS/DB

This chapter introduces Service Gateway for IDMS/DB and describes how to install, configure, and customize the software.

Topics

Introducing Service Gateway for IDMS/DB

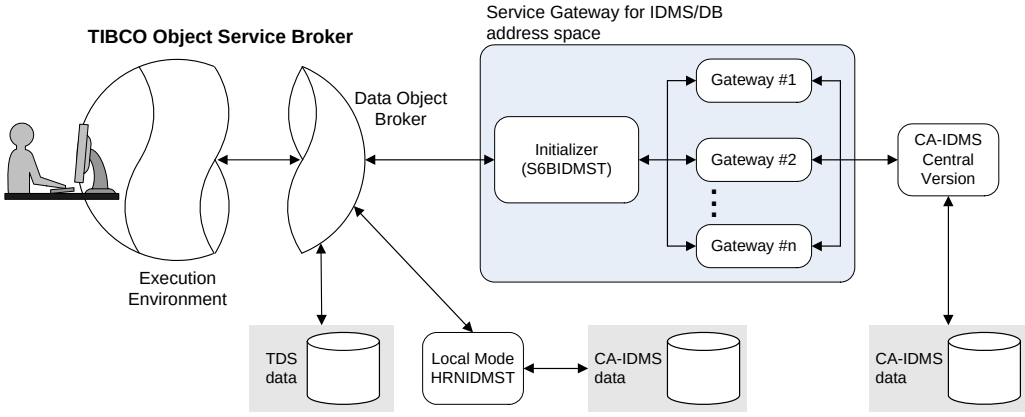
Service Gateway for IDMS/DB is a server interface used to access CA-IDMS data from within TIBCO Object Service Broker. It uses CA-IDMS services to provide retrieval and update access and to ensure data integrity. It ensures data is presented to TIBCO Object Service Broker rules in a manner consistent with TIBCO Object Service Broker behavior. The navigation of the `CA-IDMS` database is defined in the IDM table definition and is therefore not required in each application that accesses `CA-IDMS` data. You can get concurrent real-time access to CA-IDMS data using either `CA-IDMS` Central Version or Local mode.

The Service Gateway for IDMS/DB interface consists of the following components:

- TIBCO Object Service Broker **CA-IDMS Data Dictionary (IDD) extract program** – extracts subschema-specific data and creates data sets to be loaded into TIBCO Object Service Broker control tables
- TIBCO Object Service Broker **tool** – manages the definition of CA-IDMS data as follows:
  - Loads extracted subschema information created by the extract program
  - Lists subschema definitions loaded into TIBCO Object Service Broker
  - Lists IDM tables defined in TIBCO Object Service Broker
  - Deletes subschema definitions from TIBCO Object Service Broker
- **Table Definer** – defines IDM tables based on CA-IDMS subschema information.
- Service Gateway for IDMS/DB – formats TIBCO Object Service Broker requests into CA-IDMS Data Manipulation Language (DML) calls to retrieve, insert, replace, or delete CA-IDMS records.

Accessing CA-IDMS Data

The figure below shows how you can access CA-IDMS data through either the CA-IDMS Central Version or in Local mode, while still having access to TDS data, which is TIBCO Object Service Broker's native data type.



Initializing a Gateway

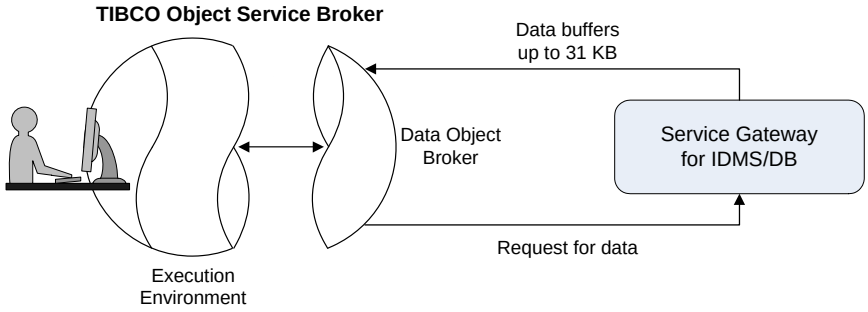
The initializer program (`S6BIDMST`) is used only to start an instance of Service Gateway for IDMS/DB (a Data Object Broker must be active before the Gateway can be started). The initializer program establishes communication to the Data Object Broker using the TIBCO Object Service Broker Communication Subsystem (OCS), interprets the gateway startup parameters, and attaches the number of Gateway tasks requested in the startup parameters.

For CA-IDMS Release 14.0 and above, the following occurs:

- The Gateway uses the standard `CA-IDMS` batch interface for accessing data through the Central Version or in Local mode.
- When accessing data through the Central Version, multiple tasks can run in a single Service Gateway for IDMS/DB address space.
- When accessing data in Local mode, only one task is initiated in the Gateway address space.

Processing CA-IDMS Data

The figure below shows how data is sent to the Data Object Broker in variable length buffers up to a maximum of 31 KB. If a row is less then 4KB, the maximum buffer size of 4 KB is used; otherwise, a 31 KB buffer is used. If a single request requires more rows than a buffer will hold, multiple buffers are sent until the request is complete.



Deployment

You can configure the Data Object Broker and the Service Gateway for IDMS/DB to reside on different hosts and/or operating systems (z/OS, Windows, or Solaris). Service Gateway for IDMS/DB must be in the same domain as the CA-IDMS database system. Refer to Connecting the Gateway to a Windows or Solaris Data Object Broker on page 53 for additional information.

If all components reside in the same domain and in authorized libraries, Cross Memory Services is used for communications. If the components do not all reside in authorized libraries, you can still get the Gateway to use Cross Memory Services; refer to Connecting the Gateway Using Cross Memory Services on page 56.

See Also

The *TIBCO Object Service Broker for z/OS Installing and Operating* manual or the *TIBCO Object Service Broker for Open Systems Installing and Operating* manual for more information on communications requirements.

Preparing for Installation

If you are installing this product on a remote host in relation to the Data Object Broker, where access to the product is only via a network, see Installing on a Remote Host on page 11.

Before installing Service Gateway for IDMS/DB, review the following:

- **TIBCO Object Service Broker Base Component** – You must install and ACCEPT (using SMP/E) the TIBCO Object Service Broker base component before installing Service Gateway for IDMS/DB. You must also have the *HLQ*.INSTALL data set that was created during that installation.
- **Supported Versions of CA-IDMS** – Refer to the Late Breaking News at http://support.tibco.com/ for the most current information about the levels, versions, and releases of CA-IDMS that Service Gateway for IDMS/DB supports.
- **Customizing the OSEMOD IDMS/DB Variables** – Member OSEMOD in the CLIST data set is an ISPF edit macro used to customize members in the IDMS.JOBS, CLIST, CNTL, and JCL data sets. Refer to *TIBCO Object Service Broker for z/OS Installing and Operating* for the IDMS/DB installation variables in OSEMOD that you should customize as required.

Distribution Media and Contents

This section describes how to obtain the software, and the installation file that comprises the distribution media. Similar to the TIBCO Object Service Broker base component, the Service Gateway for IDMS/DB software is distributed in `.xml` format within a ZIP file.

Distribution File Format

The file is in a format compatible with IBM System Modification Program/Extended (SMP/E) naming conventions. The product is packaged in SMP/E txlib format.

Installation Media

As with the TIBCO Object Service Broker base component, you can download the software from the TIBCO site by following these steps:

1. Contact TIBCO Software Inc. for a password, directory information, etc.
2. Connect to the TIBCO site with the required information.
3. Download the appropriate ZIP file.

Installation Files

The following ZIP file comprises the distribution media:

```
TIB_srvcgw-idms_6.0.0_zos.zip
```

Uploading the Software

If you have acquired Service Gateway for IDMS/DB by downloading it from the TIBCO Software site, you must upload the software to the z/OS host system.

Preparing and Uploading the Product File

1. Download or copy the `TIB_srvcgw-idms_6.0.0_zos.zip` file to a PC that can connect to the z/OS host system.
2. Unzip the file to a temporary location on the PC. The ZIP file contains multiple files; of these, the following file is the only file used in this installation:

   `idms.xm1` – compressed file containing Service Gateway for IDMS/DB

The `srvcgw_idms.xm1`, `install.bin`, `ostarrec.bin`, `property.bin`, and `OSTAREDC` files are not used in this procedure.

3. Preallocate the following sequential data set on the z/OS host system:

   HLQ`.IDMS.XM1` (Size 17 KB)

   Use the same *HLQ* that you specified when you uploaded the base component. Below is sample JCL to allocate this data set. Provide a JOB card and submit the JCL.

```
//ALLOC EXEC PGM=IEFBR14
//DD1   DD DSN=HLQ.IDMS.XM1,
//      DISP=(,CATLG,DELETE),UNIT=SYSDA,
//      DCB=(RECFM=FB,LRECL=1024,BLKSIZE=0,DSORG=PS),
//      SPACE=(TRK,(2,1))
```

4. FTP the `idms.xm1` file in BIN mode to the *HLQ*`.IDMS.XM1` data set.

Installing the Software

You must perform the installation under an ISPF environment only.

This section describes the procedure for installing the Service Gateway for IDMS/DB.

You can start the installation if you have the following data sets ready:

- *HLQ*.INSTALL
- *HLQ*.IDMS.XM1

You must use the *HLQ*.INSTALL data set that was created during the installation of the TIBCO Object Service Broker base component.

Installation Overview

To install Service Gateway for IDMS/DB, perform the following:

1. Edit the properties file by specifying the keywords for installing this component.
2. Install the software.

Edit the Properties File

Edit the PROPERTY member in *HLQ*.INSTALL as follows:

```
INSTALL=IDMS
IDMSDBL=fully qualified IDMS DBA load library
IDMSLIB=fully qualified IDMS system load library
IDMSCTL=fully qualified IDMS system SYSCTL file
```

Initial Installation

STEP 1: **Execute File Tailoring EXEC to start installation.**

Member in: *HLQ*.INSTALL

Member: INSTALL (EX member)

The IDMS.JCL data set is created at the successful completion of this step.

STEP 2: **Run Job IDMS.JCL.**

This batch job will uncompress the IDMS.XM1 file to produce the distribution library.

JCL in: *HLQ*.IDMS.JCL (Edit J0B card to your site's standards)

Data Set: *HLQ*.IDMS.JCL (SUB data set)

Uncompressing *HLQ*.IDMS.XM1 produces the distribution library *HLQ*.IDMS.FILEI.

STEP 3: **Create and customize work copies of data sets.**

Member in: *HLQ*.IDMS.FILEI

Member: S6E1CUST (EX member)

The following work copies are created and customized with values specified by OSEMOD variables:

Customized copy – Library Description

- *HLQNONV*.*INSTVER*.JCL – Sample JCL
- *HLQNONV*.*INSTVER*.IDMS.JOBS – Install jobs for IDMS/DB

STEP 4: **Initiate install jobs.**

Member in: *HLQNONV*.*INSTVER*.IDMS.JOBS

Member: S6E2RUNJ (EX member)

SEND messages are directed to the userid specified in the NOTIFY parameter of each job submitted, informing user of submission and normal completion or abnormal termination. The successful completion of the final job in JOBSE list is accompanied by the message ALL MEMBERS PROCESSED.

This completes the installation process for IDMS/DB.

You can modify the STATUS of any job as per your requirement. For example, if your shop normally ACCEPTs the product FMID at some future time, then change the status of S6E4ACPT from INSTALL to FUTURE. Note that you must ACCEPT the Service Gateway for IDMS/DB component before applying any hotfix maintenance using SMP/E.

Preparing for Subschema Access

The subschemas to be used to access CA-IDMS data from TIBCO Object Service Broker must reside in either the CA-IDMS primary dictionary load area (DDLDCLOD) or the CA-IDMS CDMSLIB load library defined in the CA-IDMS Central Version.

See Also The *TIBCO Object Service Broker for z/OS Installing and Operating* manual for more information on installing the base component.

Installing on a Remote Host

This section describes the procedure for installing the software on a remote host in relation to the Data Object Broker installation.

Distribution Media and Contents

This software is distributed in `.xml` format within a ZIP file. The file is in a format compatible with IBM System Modification Program/ Extended (SMP/E) naming conventions. The software is packaged in SMP/E `txlib` format.

Installation Media

You can download the software from the TIBCO site by following these steps:

1. Contact TIBCO Software Inc. for a password, directory information, etc.
2. Connect to the TIBCO site with the required information.
3. Download the appropriate ZIP file.

Installation Files

The following ZIP file comprises the distribution media:

```
TIB_srvcgw-idms_6.0.0_zos.zip
```

Uploading of the Software

If you have acquired the software by downloading it from the TIBCO site, you must upload the software to the z/OS host system.

Preparing and Uploading the Product File

1. Download or copy the `TIB_srvcgw-idms_6.0.0_zos.zip` file to a PC that can connect to the z/OS host system.

2. Unzip the file to a temporary location on the PC. The ZIP file contains multiple files; of these, the following files are the only files used in this installation:
   - `srvcgw_idms.xm1` – Compressed file containing Service Gateway for IDMS/DB for installation on a remote host
   - `install.bin` – The `REXX EXEC` to perform the installation
   - `ostarrec.bin` – The `REXX EXEC` to uncompress the `.xm1` file
   - `property.bin` – A template of mandatory install variables required for product installation
   - `OSTAREDC` – A load module to improve the performance of `OSTARREC`

The `idms.xm1` file is not used in this procedure.

3. Preallocate a PDS, fixed block data set on the z/OS host system with the following name:

   HLQ`.INSTALL`

   where *HLQ* is any valid high-level qualifier. Note that this *HLQ* will be used during the installation. See the sample JCL in the next step.

4. Pre-allocate the following sequential data set on the z/OS host system:

   HLQ`.OS.IDMS.XM1 (size 46,220 KB)`

   Use the same *HLQ* as the previous data set. Below is sample JCL to allocate these data sets. Provide a JOB card and submit the JCL.

```
//ALLOC EXEC PGM=IEFBR14
//DD1   DD DSN=HLQ.INSTALL,
//      DISP=(,CATLG,DELETE),UNIT=SYSDA,
//      DCB=(RECFM=FB,LRECL=80,BLKSIZE=0),
//      SPACE=(TRK,(5,15,100))
//DD2   DD DSN=HLQ.OS.IDMS.XM1,
//      DISP=(,CATLG,DELETE),UNIT=SYSDA,
//      DCB=(RECFM=FB,LRECL=1024,BLKSIZE=0,DSORG=PS),
//      SPACE=(TRK,(1000,50))
```

5. FTP `install.bin`, `property.bin` and `ostarrec.bin` to your z/OS system in BIN mode to the *HLQ*`.INSTALL` data set. Name these utilities `INSTALL`, `PROPERTY` and `OSTARREC`, respectively.
6. FTP the `srvcgw_idms.xm1` file in `BIN` mode to the *HLQ*`.OS.IDMS.XM1` data set.

Installing the OSTAREDC Program

1. Upload the OSTAREDC file to z/OS in binary format to a data set with LRECL=80 and RECFM=FB.
2. In ISPF 3.4, against this data set, type the following:

   "RECEIVE INDA(/)"

   When prompted, specify DA('*HLQ*.INSTLOAD' as the name of the load library where you want the OSTAREDC program restored, using the following syntax:

   DA('datasetname')
3. Edit OSTARREC as follows:
   - Issue the command "FIND OSTAREDC 1".
   - Change the constant after the equal sign to contain the full data set name of the program. The string must start with a double quote and a single quote, and end with a single quote and a double quote (the double quotes delimit the string and the single quotes tell TSO that the data set name is fully qualified). For example, change the following:

     OSTAREDC = "'*HLQ*.INSTLOAD(OSTAREDC)'"

     to

     OSTAREDC = "'*your.load.library*(OSTAREDC)'"

     where *your.load.library* is the name of the library referenced in Step 2.

Installation Process

You can start the installation if you have the following data sets ready:

- *HLQ*.INSTALL
- *HLQ*.OS.IDMS.XM1

The *HLQ* referenced throughout this chapter is the high-level qualifier you specified when you uploaded the product software. This is the value of the INSTALL and XM1 files you specified. It will be used as the default value for all distribution files created when an XM1 is uncompressed. It is equivalent to the value of symbolic parameter HLQ, as described in OSEMOD.

To install Service Gateway for IDMS/DB:

1. Determine your system-environment values listed in System Environment Checklist.
2. Input Updates to the Properties File according to the values determined in Step 1.

3. Following the procedure in Installation Steps.

System Environment Checklist

Before you begin the installation, review the system environment information described in Table 3 and determine whether you will use the default value or provide your own value.

Table 3 `OSEMOD` *Variables*

Description	OSEMOD Variable	Default Value	Your Value
High-level qualifier for uploaded data sets `INSTALL` and `OS.IDMS.XM1`	`$HLQ$`	Specified on upload	
High-level qualifier for non-VSAM and VSAM data sets you are authorized to create	`$HLQNONV$` `$HLQVSAM$`	`TIBCO.TESTNV` `TIBCO.TESTVS`	
Second-level qualifier for install files	`$INSTVER$`	`INS60`	
Second-level qualifier for TIBCO Service Gateway system files	`$SLQ$`	`OSB60`	
Second-level qualifier for SMP/E libraries	`$SMP$`	`SMP60`	
For SMS Shops – `managementclass`, `dataclass`, and `storageclass`, if required			

For new non-VSAM data sets	`$NMGTCLAS`
	`$NDATCLAS$`
	`$NSTOCLAS$`
		`STANDARD`
		`STANDARD`
		`S6BNONV`	

For new VSAM data sets	`$VMGTCLAS`
	`$VDATCLAS$`
	`$VSTOCLAS$`
		`STANDARD`
		`STANDARD`
		`S6BVSAM`	

High-level qualifier of Language Environment libraries for `SCEELKED` and `SCEEBIND`	`$CEELIB$`	`CEE`	

Table 3 OSEMOD Variables

Description	OSEMOD Variable	Default Value	Your Value
High-level qualifier of IBM's Callable Services library CSSLIB	$CSSLIB$	SYS1	
IDMS DBA system load library	$IDMSDBL$	CA.IDMS.R170.DBA.LOADLIB	
IDMS load library	$IDMSLIB$	CA.IDMS.LOADLIB	
IDMS SYSCTL file	$IDMSCTL$	CA.IDMS.SYSCTL	

For details, refer to the *TIBCO Object Service Broker for z/OS Installing and Operating* manual.

Updates to the Properties File

Use the PROPERTY member in *HLQ*.INSTALL as a template, and modify to suit your requirements. Table 4 describes keywords in the properties file that correspond to the system environment variables in System Environment Checklist.

Table 4 Properties File Keywords

Keyword	Description
INSTALL=	To install Service Gateway for IDMS/DB, specify REMOTEGATEWAY: INSTALL=REMOTEGATEWAY
SERVICEGATEWAY=	To install Service Gateway for IDMS/DB, specify IDMS: SERVICEGATEWAY=IDMS
HLQNONV=	High-level qualifier for non-VSAM data sets.
HLQVSAM=	High-level qualifier for VSAM data sets.
INSTVER=	Second-level qualifier for install files.
SLQ=	Second-level qualifier for TIBCO Object Service Broker system files.
SMP=	Second-level qualifier for SMP/E libraries.

Table 4 Properties File Keywords

Keyword	Description
SMS=	YES for SMS site, NO for non-SMS site. **Warning:** If you select the SMS=YES option, be sure to specify SMS-managed data-set names. The SMS automatic class selection (ACS) rules at your site determine whether a data-set name is eligible for SMS management. If the answer is yes, SMS manages that name. Otherwise, the result is unpredictable.
COMPAT=	Use if SMS=YES. Valid values: YES for SMS compatible data set name classes; NO for SMS non-compatible data set name classes. If COMPAT=NO, specify the following: • NMGTCLAS – MANAGEMENTCLASS for non-VSAM data sets • NDATCLAS – DATACLASS for non-VSAM data sets • NSTOCLAS – STORAGECLASS for non-VSAM data sets • VMGTCLAS – MANAGEMENTCLASS for VSAM data sets • VDATCLAS – DATACLASS for VSAM data sets • VSTOCLAS – STORAGECLASS for VSAM data sets
VOLSER=	If SMS=YES, specify one DASD volume for VSAM data set allocation. Default is USER01. If SMS=NO, specify three DASD volumes separated by commas. The defaults are OSBS06, OSBD18, and OSBB02. • vol1 – DASD volser for temp work files • vol2 – DASD volser for install files • vol3 – DASD volser for TIBCO Object Service Broker system files
CEELIB=	High-level qualifier of Language Environment libraries
CSSLIB=	High-level qualifier of IBM's Callable Services library CSSLIB
IDMSDBL=	Fully qualified IDMS DBA system load library
IDMSLIB=	Fully qualified IDMS system load library
IDMSCTL=	Fully qualified IDMS system SYSCTL file

Installation Steps

To exit the interactive session at any time after executing the REXX exec INSTALL, do the following:

1. Press PA1.
2. Enter hi.
3. Press ENTER twice.

Step 1: **Execute File Tailoring EXEC to start installation.**

Member in: *HLQ*.INSTALL

Member: INSTALL (EX member)

STEP 1 will verify that files can be allocated successfully using the values provided in the PROPERTY file. Test files of type sequential, PDS, PDSE, and VSAM will be allocated then deleted. Installation will stop if any test allocation fails. You should investigate the cause, correct the condition and repeat STEP 1.

The IDSM.JCL data set is created at the successful completion of this step.

Step 2: **Edit the Job card to your site's standards and run Job IDMS.JCL.**

JCL in: *HLQ*.IDMS.JCL (Edit Job Card to your site's standards.)

Data Set: *HLQ*.IDMS.JCL (SUB data set)

This batch job will uncompress the OS.IDMS.XM1 file to produce the distribution libraries. If you modify the job name, make sure it does not exceed seven characters. The job should successfully complete with a return code of 0.

Step 3: **Edit OSEMOD. (Optional)**

If you wish to make additional changes to the values of OSEMOD variables, make the changes now.

Member in: *HLQ*.FILECLS

Member: OSEMOD

Step 4: **Create and customize work copies of data sets.**

Member in: *HLQ*.OS.IDMS.FILEI

Member: S6R1CUST (EX member)

The following work copies are created and customized with values specified by OSEMOD variables:

Customized copy – Library Description

- *HLQNONV*.*INSTVER*.CLIST – CLIST
- *HLQNONV*.*INSTVER*.CNTL – CNTL
- *HLQNONV*.*INSTVER*.JCL – Sample JCL
- *HLQNONV*.*INSTVER*.OS.IDMS.JOBS – Install jobs for remote Service Gateway for IDMS/DB

Step 5: **Modify STATUS of installation jobs, as required.**

Member in: *HLQNONV*.*INSTVER*.OS.IDMS.JOBS

Member: JOBSR (EDIT member)

Jobs in MEMBER are evaluated in the order they are listed and are submitted based upon their specified STATUS. The next job is submitted only if the previous one completed with its expected return code RC.

Valid status: INSTALL (run the job), FUTURE/OPTIONAL (skip the job), DONE (job already completed).

Status is modified from INSTALL to DONE only if the job's completion code is equal to or less than the stated return code.

You can modify the STATUS of any job as per your requirement. For example, if your shop normally ACCEPTs the product FMID at some future time, then change the status of S6R4ACPT from INSTALL to FUTURE. Note that you must ACCEPT the remote Service Gateway for IDMS/DB FMID before applying any hotfix maintenance using SMP/E.

Skip this step if the default STATUS of all the jobs is acceptable.

Step 6: **Initiate install jobs.**

Member in: *HLQNONV*.*INSTVER*.OS.IDMS.JOBS

Member: S6R2RUNJ (EX member)

SEND messages are directed to the userid specified in the NOTIFY parameter of each job submitted, informing user of submission and normal completion or abnormal termination. The successful completion of the final job in JOBSR list is accompanied by the message ALL MEMBERS PROCESSED.

This completes the installation process.

Verification of Installation

The Installation Verification Procedure (IVP) for an external DBMS enables you to quickly verify that the installation of a TIBCO Object Service Broker DOB and one or more DBMS Service Gateways was successful. The IVP verifies that the communication between the DOB and a Service Gateway and that between a Service Gateway and the DBMS is functioning properly.

The IVP is split between two elements:

- Member IVPIDMSP, which was placed in *HLQNONV*.*INSTVER*.JCL by the installation procedure for the Service Gateway for IDMS, starts the gateway.
- A batch file, shell script, or JCL member, which is run in the DOB environment, loads dictionary information for the IDMS sample table and invokes the Gateway to ensure that the IDMS data is accessible.

If the gateway was installed locally, that is, inside an existing OSB installation, then both JCL members are in that installation's JCL data set.

You can perform each of the steps manually. That is, you can start the Service Gateway for IDMS normally rather than by running IVPIDMSP. You can also browse data in the external database via the Gateway without using the batch file, shell script, or JCL.

Requirements

The IVP requires the following:

- The complete installation of a TIBCO Object Service Broker DOB on z/OS, Windows, or UNIX.

- The installation of a Service Gateway. Recall that it must run from an APF-authorized library and that, if the DOB is running on z/OS, you must properly configure the Gateway in Resource Management.
- The sample record EMPLOYEE in the IDMS database.
- Configuration of the communications path between the Service Gateway and the DOB.

Configurations

Before running the IVP, you must configure the communication path through which the Service Gateway will use to communicate with the DOB. The details of this configuration depend upon the DOB's installation platform (z/OS or Open Systems) and upon which the communications protocol.

If the DOB is on Windows or UNIX and the communications protocol is TCP/IP, then the communications configuration for the Service Gateway is determined by the Relay Parameter File, which is placed in *HLQNONV*.*SLQ*.RELAYCFG by the installer. A complete discussion of how to configure that file is beyond the scope of this manual; see the section Configuring TCP/IP in the *TIBCO Object Service Broker for z/OS Installing and Operating* manual.

For the purpose of the IVP only, a simplified file will suffice. The following example illustrates a Relay Parameter File that configures a connection to a remote DOB on Windows/UNIX.

```
<?xml version="1.0"?>
<relay xmlns="http://www.tibco.com/OSB/relayparms.xsd">
   <tcpipparms tcbnum="3" maxtcbsockets="50" />
    <directory>
    <node name="WINDOB">
         <tcpip host="192.168.1.1" port="12000" />
    </node>
    </directory>
</relay>
```

Substitute the name, host, and port number for your DOB into this template. Their values are in the HURON.DIR file in the database directory of the OSB installation to which you would like to connect.

The data set *HLQNONV*.*INSTVER*.JCL(IVPIDMSP) contains the job steps required for preparing the Service Gateway for verification. Once you have met all the requirements, customize the JCL and run it. Note that if the Service Gateway has already been configured and started, omit that step.

To customize the JCL, replace each instance of `$TDS$` in the JCL with the DOB node name that you placed in the Relay Parameter File.

Recall that the IVP itself is a batch file, shell script, or JCL member in the DOB environment. Depending on your operating system, do the following:

- **On Windows** Locate the batch file `ivpidms.bat` in the `bin` directory of your OSB installation. Follow the directions in the file to customize the variables within that file, including the settings for where to place the results.
- **On Solaris** Locate the (`ksh`) shell script `ivpidms.ksh` in the `bin` directory of your OSB installation. Follow the directions in the script to customize the shell and environment variables within the script, including the settings for where to place the results.
- **On z/OS** Locate the JCL member `IVPDIMS` in the JCL data set associated with the DOB and configure the values for `TDS`, `USERID`, and `PASSWORD`. You can view the results in the job's output with `SDSF`.

Verification

Follow these steps for verification:

1. On the gateway side, submit the JCL to start the Service Gateway for IDMS manually or by running `IVPIDMSP`.
2. On the DOB side, run the `IVPIDMSP` JCL member, `ivpdims.bat`, or `ivpdims.ksh` to access the predefined DBMS sample table.
3. Check the results.

   If the verification procedure succeeded, the sample data from the database is in the output.

Uninstalling on a Remote Host

To uninstall the software, perform the following:

STEP 1: **Run the IDMS/DB cleanup job.**

Member in: *HLQNONV*.*INSTVER*.OS.IDMS.JOBS

Member: S6R9CLEN (Edit JOBCARD and SUB member)

STEP 2: **Manually delete the following data sets (in the specified sequence):**

1. HLQ.FILECLS
2. HLQ.FILECTL
3. HLQ.FILEEM1
4. HLQ.FILEEM2
5. HLQ.FILEJCL
6. HLQ.FILEOBJ
7. HLQ.FILETRK
8. HLQ.FILEXML
9. HLQ.MACRO
10. HLQ.OS.IDMS.FILEI
11. $HLQNONV$.$INSTVER$.CLIST
12. $HLQNONV$.$INSTVER$.CNTL
13. $HLQNONV$.$INSTVER$.JCL
14. $HLQNONV$.$INSTVER$.OS.IDMS.JOBS

At this point, you should only have the uploaded data sets:

- HLQ.INSTALL
- HLQ.OS.IDMS.XM1

Manually delete the data sets.

Implementing Security

Use one of the following methods to secure access to CA-IDMS objects:

- Program registration
- An external security package

Using Program Registration

Program registration verifies the CA-IDMS application program is authorized to access the requested subschema. TIBCO Service Gateway for IDMS/DB uses one of the following identifiers as the program ID in the CA-IDMS BIND RUN UNIT:

1. HURON
2. The TIBCO Object Service Broker session ID
3. The current TIBCO Object Service Broker security group name

CA-IDMS Security Requirements if Using Program Registration

Depending on which identifier you want the Gateway to use as the program ID in the CA-IDMS BIND RUN UNIT, you must specify one of three combinations of the SECLEVEL, SECURITY, and EXTERNALUSERID parameters at the Gateway startup time. The following table illustrates these three combinations. Refer to Supplying Service Gateway for IDMS/DB Startup Parameters on page 40 for descriptions of these parameters.

Program ID in CA-IDMS BIND RUN UNIT	SECLEVEL	SECURITY	EXTERNALUSERID
HURON	0	Ignored if specified.	Ignored if specified.
TIBCO Object Service Broker session ID.	1	Do not include this parameter.	USERID
Current TIBCO Object Service Broker security group name.	1	Do not include this parameter.	GROUP

External Security Package

The external security package you are using verifies that the user ID is authorized to access the requested subschema. When the external security package is invoked, one of the following three identifiers is passed as the user ID:

- The user ID that started the Gateway (or started task name for started task Gateways)
- The TIBCO Object Service Broker session ID
- The current TIBCO Object Service Broker security group name

You *must* install the TIBCO Object Service Broker external security interface macros to pass the TIBCO Object Service Broker session ID or the current TIBCO Object Service Broker security group as the user ID. Depending on which identifier you want to pass as the user ID, you must specify one of four combinations of the `SECLEVEL`, `SECURITY`, and `EXTERNALUSERID` parameters at startup. The following table illustrates the four combinations. For details, see Supplying Service Gateway for IDMS/DB Startup Parameters on page 40 and Implementing External Security on page 49.

Identifier Passed to CA-IDMS	SECLEVEL	SECURITY	EXTERNALUSERID
STN [a] / USJ [b]	0	Ignored if specified.	Ignored if specified.
STN / USJ	1	The TIBCO Object Service Broker external security interface macros are not installed; do not include this parameter.	Ignored if specified.
TIBCO Object Service Broker session ID	1	`EXTERNAL`	`USERID`
Current TIBCO Object Service Broker security group name	1	`EXTERNAL`	`GROUP`

a. The started task name if the Gateway is running as a started task.

b. The user ID that submits the Gateway job if the Gateway is running as a batch job.

TIBCO Object Service Broker Security

To restrict the ability to define IDM tables from within TIBCO Object Service Broker, restrict read access to the TIBCO Object Service Broker control tables `@IDMSRECORDS`, `@IDMSELEMENTS`, `@IDMSSETS`, and `@IDMSINDEXES`.

Fail Safe Processing

To guarantee consistency when updating both TDS and CA-IDMS data from a single instance of TIBCO Service Gateway for IDMS/DB in a single transaction, you must use Fail Safe level 1 processing. For more information, refer to Implementing Fail Safe Processing on page 50.

See Also The *TIBCO Object Service Broker Managing Security* manual for more information on restricting table access.

Specifying Service Gateway for IDMS/DB Parameters

You must specify the gateway parameters in the SYSIN member associated with DDname IDMSSRV0. You can include the gateway parameters in any order, one per record, beginning in column one, and ending with a blank or a comma. An asterisk (*) in column one indicates a comment record.

The SERVERID parameter can be overridden at runtime. Refer to Dynamically Changing the Gateway Parameters on page 47 for more information.

Gateway input parameters are listed below; those you must set at installation time are shown as Required=Y, those you can add or modify later are shown as Required=N. For additional details, see Supplying Service Gateway for IDMS/DB Startup Parameters on page 40. A sample input parameter data set is provided in member XIDMCNTL of the CNTL data set distributed with TIBCO Object Service Broker.

Parameter	Default	Required
DEBUG	Not applicable—include parameter or omit	N
EXTERNALSECURITY	EXTERNAL	N
EXTERNALUSERID	USERID	N
FSLEVEL	0	N
FSTABLENAME	@IDMFSTRXDB	N
IDPREFIX	IDMS	Y
MDL	HRNnnn	Y
POOLSIZE	256 KB	Y
SECLEVEL	0	Y
SERVERID	DEFAULT	Y
SERVERS	3	Y
TDS	None	Y
TRXDB	S6BSS001	N

Understanding the Startup Prerequisites

Prerequisites

Before you can start TIBCO Service Gateway for IDMS/DB, you must do one of the following:

- If the Data Object Broker is on z/OS, the Gateway must be identified to the Data Object Broker. To do this, define the Gateway resources to the Data Object Broker's resource management repository file.
- If the Data Object Broker is on Windows, set up National Language Support, if necessary. Refer to the *TIBCO Object Service Broker National Language Support* manual for setup and configuration information.

Default Resource Settings (z/OS only)

Use the Resource Management option (option 3) available from the Administration control group of the TIBCO Object Service Broker Administration Menu. You need to use the Resource Details (`PF5`) and the Resource Schedules (`PF10`) screens to specify the connection attributes. To get to the Resource Detail screen, you must first specify a type (`SERVERTYPE`) and group (`SERVERID`) on the Resource Type screen.

The following table illustrates the attributes for `SERVERID=DEFAULT`:

Resource Details				Resource Schedule
Intermediate Rollbk	Early Release	Last User Reuse	Commit Level	Online Only
N	Y	N	0	N

Default Settings

The default settings of the fields are affected by the values of the RESPONSEMODE and FSLEVEL gateway parameters:

Depending on the values you set in Specifying Service Gateway for IDMS/DB Parameters on page 26, you must specify the following defaults:

If Gateway Parameter is …	Then …
RESPONSEMODE=SYNC	EARLY RELEASE must be set to N.
RESPONSEMODE=ASYNC	EARLY RELEASE must be set to Y.
FSLEVEL=1	COMMIT LEVEL must be set to 1.

See Also *TIBCO Object Service Broker for z/OS Installing and Operating* manual for more information on defining and managing resources, and the Administration Menu.

Starting the Gateway

You can run the Gateway as either a batch job or a started task. For information on adding the Gateway tasks, refer to Adding the Gateway Tasks on page 58.

Running the Gateway as a Batch Job

If the Gateway is running as a batch job, you can access data through either the Central Version or in Local mode.

Central Version

A sample batch JCL to run the Gateway accessing data through the Central Version is located in member `IDMSSJCL` of the JCL data set. Customize the JCL to run with CA-IDMS and submit it.

Local Mode

A sample batch JCL to run the Gateway in Local mode is located in member `IDMSLJCL` of the JCL data set. You must define each CA-IDMS database to be used during Local mode processing in the Gateway's JCL. If running the Gateway in Local mode, only one task is available. Customize the JCL to run with CA-IDMS and submit it.

Running the Gateway as a Started Task

If you are running the Gateway as a started task, you can access data either through the Central Version or in Local mode as follows:

Central Version	Modify the JCL member IDMSSJCL of the JCL data set to be a JCL procedure and move it to your site `PROCLIB` (for example, `SYS1.PROCLIB`). You can then start the Gateway from the z/OS operator console using the z/OS **`START`** command.
Local Mode	Modify the JCL member IDMSLJCL of the JCL data set to be a JCL procedure and move it to your site PROCLIB (for example, `SYS1.PROCLIB`). You can then start the Gateway from the z/OS operator console using the z/OS **`START`** command.

See Also *TIBCO Object Service Broker Messages With Identifiers* for information on messages produced by the Gateway.

Shutting Down the Gateway

You can shut down a Gateway using the **MODIFY** operator command from the z/OS operator console, or by using the RESOURCE MANAGEMENT option from the Administration menu.

MODIFY Operator Command

The format of the **MODIFY** command is as follows:

MODIFY *dob_jobname*,**STOPSERVER=***idmsserv*

dob_jobname	Name of the batch job or the started task name under which the Data Object Broker is running.
idmsserv	Unique name of the Gateway. The Gateway creates this name by appending a two-digit number to the IDPREFIX parameter specified in the Gateway startup JCL. The default is IDMS. To view the unique name assigned to existing an instance of TIBCO Service Gateway for IDMS/DB, select the RESOURCE MANAGEMENT option from the Administration menu.

Additional MODIFY Commands

You can also use one of three variations of the **MODIFY** operator command to shut down a group of instances of TIBCO Service Gateway for IDMS/DB:

- Shut down all the Gateways:

  MODIFY *dob_jobname*,**STOPSERVER=ALLIDMS**

- Shut down all the Gateways with a common IDPREFIX:

  MODIFY *dob_jobname*,**STOPSERVER=***idprefix**

- Shut down all the Gateways with a common SERVERID:

  MODIFY *dob_jobname*,**STOPSERVER=SRVID***serverid*

RESOURCE MANAGEMENT Option

Use the RESOURCE MANAGEMENT option from the Administration menu to shut down either one Gateway or groups of Gateways. For details on the Administration menu, see the *TIBCO Object Service Broker for z/OS Installing and Operating* manual.

Chapter 2 Operating Service Gateway for IDMS/DB

This chapter describes how to operate Service Gateway for IDMS/DB.

Topics

Defining CA-IDMS Databases to TIBCO Object Service Broker

There are two methods you can use to define a CA-IDMS database to TIBCO Object Service Broker, depending on the location of your CA-IDMS data. Both methods produce the same results:

- Using IDMSLOAD in the JCL data set
- Using the S6BIDUTL or the S6BIDU12 program, and the IDMS TIBCO Object Service Broker tool

Using IDMSLOAD

IDMSLOAD in the JCL data set provides a sample JCL to extract a CA-IDMS subschema definition from the IDD and load that definition into TIBCO Object Service Broker in a single job. IDMSLOAD performs the following functions:

1. Extracts a CA-IDMS subschema definition into four data sets (RECORD, ELEMENT, SET, and INDEX) that it automatically creates
2. Executes the LOAD_IDMS_DEF rule to load the four data sets extracted in the previous step into TIBCO Object Service Broker tables
3. Deletes the four data sets created

Using S6BIDU12 and the IDMS Tool

To use S6BIDU12 and the IDMS tool to define a CA-IDMS database to TIBCO Object Service Broker, complete the following tasks:

1. Extracting CA-IDMS data dictionary information on page 32
2. Loading a CA-IDMS subschema definition into TIBCO Object Service Broker on page 34

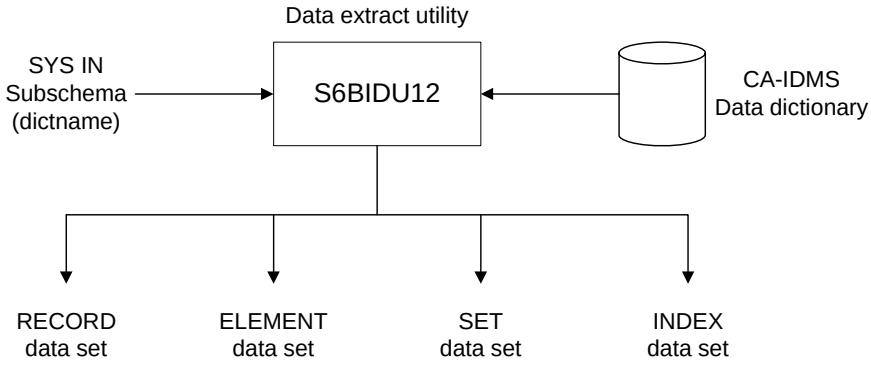
These tasks are described in the following sections.

Task A Extracting CA-IDMS data dictionary information

To extract CA-IDMS element, record, set, and index information from the IDD, use the S6BIDU12 program for subschemas for Release 12.0 and greater. The SYSIN member identifies the subschema that you want to define to TIBCO Object Service Broker and the dictionary where the subschema resides. You can request specific records to be unloaded if you do not require all records defined in the subschema. Only one subschema can be extracted at a time.

Extracting information from the IDD using the extract program

The extract program navigates the IDD database to create RECORD, ELEMENT, SET, and INDEX data sets, as shown in the illustration below. These data sets are defined to TIBCO Object Service Broker as import files and are loaded into TIBCO Object Service Broker control tables. This process is described in Task B, Loading a CA-IDMS subschema definition into TIBCO Object Service Broker, on page 34.



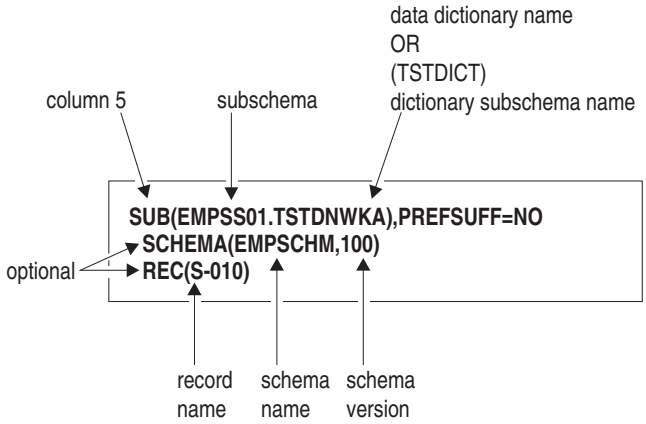
The sample JCL required to run the extract program exists in member IDMSUJCL of the JCL data set. This member extracts the CA-IDMS subschema definition.

You must extract and load only the subschema that contains all the records defined in the application database. You can define all IDM tables using this subschema. When defining the table, you can specify which Run Time Subschema accesses the data.

Syntax For Identifying the CA-IDMS Subschema

The illustration below shows a sample SYSIN member identifying the CA-IDMS subschema in the extract program. To select specific records from a subschema, include as many REC lines as required.

The PREFSUFF=NO parameter removes the CA-IDMS prefix or suffix added to element names from the TIBCO Object Service Broker control table entries. The default specification for this parameter is YES.



Task B Loading a CA-IDMS subschema definition into TIBCO Object Service Broker

To load the information extracted from the IDD, as described in Extracting CA-IDMS data dictionary information on page 32, complete the following steps:

1. Edit the definitions for the @IDMS_GEN_ELE, @IDMS_GEN_REC, @IDMS_GEN_SET, and @IDMS_GEN_NDX tables.

   For each of these tables, enter the name of the corresponding data set created by the extract program in the **FILENAME** field.

   Notes: To edit these import tables you must have MODIFY_DEFN access to them. *Only* a level-7 system administrator can assign MODIFY_DEFN for the import tables. For the @IDMS_GEN_REC layout, it is recommended that you allow for large record sizes >9999 bytes long by re-extracting the subschemas and reloading them into TIBCO Object Service Broker.

   The IDMS load program fails when a second attempt is made to reload definitions for a given subschema. IDMS utilities use $CLRTAB prior to loading the definition, which requires the user to have MOD_DEFN permission on the following system tables: @IDMSELEMENTS, @IDMSRECORDS, @IDMSINDEXES, and @IDMSSETTS (this applies to level-1 users).

2. Execute the IDMS tool

   You can execute the tool from the workbench using either the EX execute rule option or the primary command field.

The screen below shows the Manager Utilities for IDMS Data main menu. To select a menu option, type an **S** in the line command field beside it and press Enter.

```
 MANAGER UTILITIES FOR IDMS DATA
---------------------------------

Enter an "S" to select a function

_  List IDMS Tables

_  List IDMS Subschemas

_  Load an IDMS Subschema

_  Delete IDMS Subschemas




PFKEYS: 1=HELP 3=EXIT 12=EXIT
```

3. Select the Load an IDMS Subschema option.

   A screen similar to the one below appears. Verify that the data set names on the screen correspond to those created by the extract program.

```
MANAGER UTILITIES FOR IDMS DATA:  DEFINE SUBSCHEMA
Command ===>

------------------------------------------------------------------------------
     About to define SUBSCHEMA: EMPSS01   from IMPORTFILES listed
------------------------------------------------------------------------------


   IDMS Records .....      ...> USR00.IDMS.REC.DATA
   IDMS Elements.....      ...> USR00.IDMS.ELE.DATA
   IDMS Sets ........      ...> USR00.IDMS.SET.DATA
   IDMS Indexes .....      ...> USR00.IDMS.NDX.DATA




PFKEYS: 1=HELP 3=EXIT 12=EXIT 4=CONFIRM
```

4. Press PF4 to confirm that you want to load the subschema definition that appears on the screen.

   If the subschema definition currently exists, it is overwritten. The IDM table definitions using this subschema still exist; however, they contain no subschema changes. The sample screen below shows a count of the number of records, elements, sets, and indexes.

```
MANAGER UTILITIES FOR IDMS DATA:  DEFINE SUBSCHEMA
Command ===>

------------------------------------------------------------------------------
     About to define SUBSCHEMA: EMPSS01   from IMPORTFILES listed
------------------------------------------------------------------------------


   IDMS Records .....   14 ...> USR00.IDMS.REC.DATA
   IDMS Elements.....  391 ...> USR00.IDMS.ELE.DATA
   IDMS Sets ........   24 ...> USR00.IDMS.SET.DATA
   IDMS Indexes .....   19 ...> USR00.IDMS.NDX.DATA




PFKEYS: 1=HELP 3=EXIT 12=EXIT 4=CONFIRM
```

If your subschema definition changes, re-extract the subschema with the extract program described in Extracting CA-IDMS data dictionary information on page 32, and reload the subschema using the IDMS tool.

Determining When to Modify Table Definitions

You do not have to modify existing IDM table definitions unless one of the following changes occur:

- Elements represented in an IDM table are removed from a record. You must modify all IDM tables defined on that record.
- Element definitions changed. You must modify the IDM tables that contain those elements.
- A record is deleted. An IDM table that accesses that record must be modified to exclude the deleted record.

See Also *TIBCO Object Service Broker Managing Security* for more information on security access levels.

Conversions

Conversion of Record and Element Names

The conversions that take place when loading CA-IDMS subschema definitions into TIBCO Object Service Broker are as follows:

- Element names are truncated to 16 characters.
- Hyphens in element names are changed to underscores.
- A sequence number is appended to element names of FILLER to make the element names within a record unique. For example, in the case of FILLER_33, FILLER is the 33rd element in the record.
- Group names are not stored in the TIBCO Object Service Broker control tables.
- OCCURS and OCCURS DEPENDING ON are supported up to a maximum of 8999 occurrences. Since occurring fields are not part of a relational database structure, a prefix is added to each field occurrence. For example, if the element `AMOUNT` occurs three times, the TIBCO Object Service Broker definition is `I001_AMOUNT`, `I002_AMOUNT`, and `I003_AMOUNT`.
- Redefined elements are stored in the TIBCO Object Service Broker control tables.

Conversion of Data Types

The following table shows the default mapping of CA-IDMS data types to TIBCO Object Service Broker data types. This mapping is imposed by the Table Definer. When CA-IDMS data is modified by a TIBCO Object Service Broker transaction, it is converted back to its original data type before being sent to CA-IDMS.

CA-IDMS Data Type	External Syntax	TIBCO Object Service Broker Syntax	TIBCO Object Service Broker Length
CHARACTER	C	C	Same
ALPHABETIC	A	C	Same
ZONED NUMERIC UNSIGNED (length=4)	M	B	2

CA-IDMS Data Type	External Syntax	TIBCO Object Service Broker Syntax	TIBCO Object Service Broker Length
ZONED NUMERIC UNSIGNED (length=8)	M	B	4
ZONED NUMERIC UNSIGNED (length=9)	M	B	4
ZONED NUMERIC UNSIGNED (length not=4, 8, or 9)	M	P	Absolute value of (length/2 + 1)
ZONED NUMERIC SIGNED	N	P	length/2+1
PACKED NUMERIC UNSIGNED	U	P	Same
PACKED NUMERIC SIGNED	P	P	Same
BINARY (2, 4, 8) UNSIGNED	K	P	3, 5, 8 [a]
BINARY (2, 4, 8) SIGNED	B	P	3, 5, 8 [a]
BIT	X	Not supported	
FLOATING POINT	F	F	Same

a. If there are more than 31 significant digits in the field, you must assign the field syntax C and equivalent length.

Providing Default Database Names for Users

If a user always accesses data from a specific physical database, you can assign a default database name to the user for each subschema they require.

To assign a default database name to the user for each subschema, edit the @IDMSDBNAME(*subschema*) table, adding the user ID in the **USERID** field and the database name in the **DBNAME** field.

@IDMSDBNAME Table for the EMPSS01 Subschema

```
EDITING TABLE    :    @IDMSDBNAME(EMPSS01)
COMMAND ==>
                                                                SCROLL: P
    USERID    DBNAME
_ -------- --------
_ DGF40     WEST
_ SHR10     EAST
_ USR40     SOUTH
_ USR90     NORTH




PFKEYS: 4=INSERT 16=DELETE 5=FIND NEXT 6=CHG NEXT 18=EXCLUDE 3=SAVE 12=CANCEL
```

If a database name is specified in the Table Definer **DBNAME** field and the **DBNAME** field of the @IDMSDBNAME(*subschema*) table, the entry in @IDMSDBNAME is used when any of the user IDs specified in this table access CA-IDMS data using the IDM table definition.

The @IDMSDBNAME table enables specific users to access different physical databases using the same IDM table definition. You use the default security to access and update this table. If necessary, ensure that users who should not modify this table have the proper security restrictions.

Binding IDM Table Definitions

You can bind an IDM table definition, but not its data. IDM tables for which you request binding are bound to both the Execution Environment and the Gateway when they are accessed from a rule. If you change a definition, it is automatically rebound in the Service Gateway for IDMS/DB.

You can specify the maximum amount of space available to hold all IDM table definitions using the POOLSIZE startup parameter. Refer to Supplying Service Gateway for IDMS/DB Startup Parameters on page 40 for more information.

See Also *TIBCO Object Service Broker Managing Security* for more information on security access levels. See *TIBCO Object Service Broker Application Administion* for information on binding tables.

Supplying Service Gateway for IDMS/DB Startup Parameters

You must specify the gateway parameters in the SYSIN member associated with DDname IDMSSRV0. For more information on how to specify the gateway parameters, refer to Specifying Service Gateway for IDMS/DB Parameters on page 26.

The SERVERID parameter can be overridden at runtime. Refer to Dynamically Changing the Gateway Parameters on page 47 for more information.

Available Parameters

Available parameters are listed below:

DEBUG	Specify that debugging messages (the type of request and the TIBCO Object Service Broker transaction number) should be echoed on the Gateway and placed in the JES job log for batch jobs and appear on the console for started tasks. At normal transaction end, the physical I/O count from the CA-IDMS statistics block also appears.

security	Has meaning only if SECLEVEL=1, an external security package is used to verify user IDs against subschemas and/or areas, and you have the TIBCO Object Service Broker external security interface macros installed. The TIBCO Object Service Broker session ID or the current security group is passed to an external security package (for example, CA-ACF2) to verify access to subschemas and/or areas. The default is EXTERNAL. If CA-ACF2 security is used, the Gateway must be installed and executed from an authorized library. For CA-IDMS Release 12.0 and greater, the TIBCO Object Service Broker external security interface can use any CA-IDMS supported external security package.

EXTERNALUSERID	Has meaning only if SECLEVEL=1. Specifies the ID to be used for external security validation. The default is USERID.

USERID	The TIBCO Object Service Broker session ID.
GROUP	The current TIBCO Object Service Broker security group. **Note** Ensure that all security groups used to verify CA-IDMS resources are between one and eight characters long. A SECURITYFAIL occurs for more than eight characters.

FSLEVEL (FSL)

Use to specify the level of Fail Safe processing. The default value is zero.

Valid values:

1	Activate Fail Safe Level 1. The Gateway informs the Data Object Broker that it can support Fail Safe level-1 processing. If the Gateway is to attach to a z/OS Data Object Broker, the Data Object Broker's connection attribute setting "commit level" must be set to 1. If not, the Gateway connection is rejected. Refer to Implementing Fail Safe Processing on page 50 for more information. You must specify the TRXDB, FSTABLENAME, RECOVERYID, and RECOVERYPASSWORD parameters.
0	De-activate Fail Safe processing. The Gateway informs the Data Object Broker that it does not support Fail Safe level-1 processing. If the Gateway is to attach to a z/OS Data Object Broker, the Data Object Broker's connection attribute setting "commit level" must be set to 0. If not, the Gateway connection is rejected. Refer to Implementing Fail Safe Processing on page 50 for more information.

FSTABLENAME

Required only if FSLEVEL=1. Specifies the name of the IDM table that maps to the CA-IDMS transaction file. The default is @IDMFSTRXDB.

IDPREFIX	Each instance of TIBCO Service Gateway for IDMS/DB must have a unique IDPREFIX. The Gateway appends two decimal digits to the prefix to create a unique name for each instance of TIBCO Service Gateway for IDMS/DB, and uses the result to log in to TIBCO Object Service Broker. This parameter must have four characters; the default is IDMS.
MIXEDSSCDETECTION	Only one subschema can be actively accessed per transaction. If multiple subschemas are found to be referenced in processing multiple IDM type tables, the gateway takes the following action, depending on the value specified or defaulted for this parameter. The values are as follows: • **I** – Ignore this condition. • **W** – Display a message in the gateway log on the discrepancy but allow the transaction to continue. This value is the default. • **E** – Display the message in the gateway log and end the transaction with an error.
MDL	The pattern for selecting the VTAM ACB name that the Gateway uses for communications. If not specified, the Gateway uses the TDS parameter as the pattern. If you do not specify the MDL parameter, ensure that the TDS parameter is a valid VTAM ACB model. For example, if TDS=HRN001 and you do not include an MDL parameter, the model HRN*nnn* is used.
POOLSIZE	Set the amount of space (in kilobytes) to hold IDM table definitions in each instance of TIBCO Service Gateway for IDMS/DB. This parameter can be up to a maximum of 16384 KB. The default value is 256 KB. An estimate of the number of IDM tables that can be accessed in a single transaction is POOLSIZE divided by CTABLESIZE. Refer to Estimating the CTABLESIZE Parameter on page 45 for more information.

RESPONSEMODE	The mode that the Gateway uses to respond to requests. You can use this mode to free up a Gateway more quickly for a new transaction in read-only situations. For use only with a Data Object Broker on z/OS. The default is ASYNC. Valid entries:

ASYNC	The Data Object Broker releases the Gateway from the TIBCO Object Service Broker transaction when the end-of-transaction request is sent, provided no updates are requested. The next transaction can then be scheduled before the previous transaction is completed. ASYNC is recommended for online Gateways. If SYNC is used, performance can be affected.
SYNC	Causes the Data Object Broker to wait for the Gateway to complete end-of-transaction processing. SYNC is recommended for Gateways running in batch mode.

SECLEVEL	Specify the level of authorization to use when accessing CA-IDMS data. The default value is 0.

0	If program registration is used to verify access to subschemas, HURON is used as the program ID. If an external security package is used to verify user IDs against subschemas and/or areas: • The user ID that submits the Gateway job is used for the Gateways running as batch jobs. • The started task name is used for the Gateways running as started tasks.
1	If program registration is used to verify access to subschemas, the TIBCO Object Service Broker session ID or current security group name is used as the program ID. If an external security package is used to verify user IDs against subschemas and/or areas and the TIBCO Object Service Broker external security interface macros are not installed: The user ID that submits the Gateway job is used for the Gateways running as batch jobs. The started task name is used for the Gateways running as started tasks. If an external security package is used to verify user IDs against subschemas and/or areas, and the TIBCO Object Service Broker external security interface macros are installed, you must specify EXTERNALSECURITY=EXTERNAL; this means the TIBCO Object Service Broker session ID or current security group name is used as the user ID.

SERVERID	Identify a pool of the Gateways with common characteristics. If you have multiple TIBCO Object Service Broker IDMS/DB initializer programs with the same SERVERID, ensure that the FSLEVEL and FSTABLENAME or TRXDB gateway parameters have the same values for each initializer program. This parameter can be up to eight characters; the default is DEFAULT. This parameter can be overridden at runtime. Refer to Dynamically Changing the Gateway Parameters on page 47 for more information.

SERVERS	The number of the Gateway tasks that the TIBCO Object Service Broker IDMS/DB initializer program should attach to the Gateway address space at startup. If running the Gateway in Local mode, only one task is available. If accessing data through the Central Version, this number can be from 1 to a value less than or equal to the value set in the **Maximum Connection Count** field in your Network Configuration. The default value is 3. Refer to Adding the Gateway Tasks on page 58 or the *TIBCO Object Service Broker for z/OS Installing and Operating* manual for more information.

TDS	Supply the Communications Identifier of the Data Object Broker with which the Gateway communicates.

TRXDB	Required only if FSLEVEL=1. Specifies the name of the CA-IDMS subschema that defines the CA-IDMS transaction ID database. The default is HRNSS001.

If you have multiple TIBCO Object Service Broker IDMS/DB initializer programs with the same SERVERID, the FSLEVEL and FSTABLENAME or TRXDB parameters must have the same values for each initializer program.

Estimating the CTABLESIZE Parameter

When you select CA-IDMS elements as TIBCO Object Service Broker IDMS/DB fields, the number of fields you can access using an IDM table definition is dependent upon the CTABLESIZE Data Object Broker parameter. To estimate the number of bytes required to support a specified number of fields, invoke the ESTIMATETBLDFN shareable tool from the workbench as follows:

```
ESTIMATETBLDFN(num_fields)
```

You must supply one argument, the maximum number of fields accessed by an IDM table in your system. The rule returns an estimate of the maximum CTABLESIZE required (for each TIBCO Object Service Broker table type) to support this number of fields.

The following screen illustrates the result of executing the rule for 50 fields.

```
------------------------------ INFORMATION LOG ---------------------------------
COMMAND ===>                                                    SCROLL ===> P
DATE: Mar 28,2007 REPORT ON ESTIMATE CTABLESIZE
                          FOR "50" FIELDS

Table Type       CTablesize(K)
----------       -------------

ADA                 5
DAT                 7
DB2                 5
IDM                 6
IMS                 6
MAP                 4
SLK                 4
204                 6
TDS                 3




PFKEYS: 2=NEXT LOG 3=EXIT 5=REPEAT FIND 12=EXIT 13=PRINT 9=RECALL
```

If the CTABLESIZE parameter is set to its maximum of 31 KB, you can access approximately 380 fields using one IDM table definition, providing the data does not exceed 31744 bytes per row.

See Also

- *TIBCO Object Service Broker Shareable Tools* for more information on ESTIMATETBLDEFN.
- *TIBCO Object Service Broker Parameters* for more information about the CTABLESIZE Data Object Broker parameter.

Dynamically Changing the Gateway Parameters

When a table is defined, attributes specific to external DBMS table types are held in the @SERVERPARMS TIBCO Object Service Broker control table, which is parameterized by table type. Each occurrence in the table specifies a value for the table with regard to the external environment, such as SERVERID or SERVERTYPE.

Sample @SERVERPARMS Control Table for Table Type IDM

```
BROWSING TABLE   :  @SERVERPARMS(IDM)
COMMAND ==>

    NUMBER          NAME         TYPE  SYNTAX   LENGTH   DECIMAL       DEFAULT
_ ---------- --------------      -     -      ------   -------  -------------------
_           1 SERVERID             S     C           8         0  DEFAULT
_           2 SERVERTYPE           S     C           3         0  IDM
_           3 SUBSCHEMA            S     C           8         0
_           4 READY_MODE           S     C           2         0  SR
_           5 DBNAME               S     C           8         0
_           6 OPTIMIZEUPDATE       S     C           1         0  N
_           7 USERSUBSCHEMA        S     C           8         0
_           8 CODEPAGE             S     C          32         0
_           9 VERSION              Q     V           8         0  0

PFKEYS: 1=HELP 5=FIND NEXT 9=RECALL 18=EXCLUDE 13=PRINT 3=END 14=EXPAND
```

Modifying the SERVERID Startup Parameter

At runtime, you can dynamically modify the SERVERID startup parameter using the SETXPARM and RESETXPARM shareable tools. This reduces the number of table definitions required to define the external data. The changes are stored in either of two session tables:

@SRVRPRMS_TYP	Manages global changes to the table type

@SRVRPRMS_TBL	Manages specific changes to an individual table

The changes are in effect for the duration of the session or until SETXPARM is invoked again or the overrides are reset.

The Gateway Parameters That Can Be Overridden at Runtime

The following table lists the gateway parameters and Table Definer fields that can be dynamically changed with SETXPARM and RESETXPARM:

Name	Parameter or Field	Default Value	Maximum Length (Bytes)
SERVERID	Parameter	DEFAULT	8
DBNAME	Field		8
READY_MODE	Field	SR	2
OPTIMIZEUPDATE	Field	N	1
USERSUBSCHEMA	Field		8

Examples Using SETXPARM and RESETXPARM

The following example sets the SERVERID for all IDM tables to TORONTO:

```
CALL SETXPARM('TABLETYPE', 'IDM', 'SERVERID', 'TORONTO', '');
```

This example sets the value of the **Optimize Update** field for the EMPLOYEE table to Y:

```
CALL SETXPARM('TABLENAME', 'EMPLOYEE', 'OPTIMIZEUPDATE', 'Y', '');
```

This example resets the SERVERID for IDM tables to the Table Definer default value.

```
CALL RESETXPARM ('TABLETYPE', 'IDM', 'SERVERID', '');
```

See Also *TIBCO Object Service Broker Shareable Tools* for detailed descriptions of SETXPARM and RESETXPARM.

Implementing External Security

The TIBCO Object Service Broker external security interface is used only if:

- An external security package such as CA-ACF2 is used to secure access to CA-IDMS objects
- The SECLEVEL=1 gateway startup parameter
- The TIBCO Object Service Broker external interface macros are installed

The TIBCO Object Service Broker external security interface macros for CA-IDMS Release 12.0 and greater support any external security package used with CA-IDMS.

Security Process

When the Gateway issues a CA-IDMS BIND RUN UNIT, CA-IDMS Exit 14 is initiated. Depending on the setting of the EXTERNALUSERID gateway startup parameter, Exit 14 passes the TIBCO Object Service Broker session ID or the current security group as the user ID to the external security package. Refer to Supplying Service Gateway for IDMS/DB Startup Parameters on page 40 for more information. The external security package verifies that the user ID is authorized to access the requested subschema or area.

If the TIBCO Object Service Broker external security interface macros are not installed and an external security package (for example, CA-ACF2) is used to verify user IDs against subschemas and/or areas, the user ID that submits the Gateway job requires access (for batch jobs) or the started task name requires access (for started tasks).

Installing the External Security Interface

This process assumes an external security package (such as the CA-ACF2/IDMS interface) is installed and that Exit 14 (in RHDCUXIT) is activated.

CA-IDMS Release 12 and Greater

If you are installing an external security interface for CA-IDMS Release 12 or greater, the HRNSEC12 and IDMSS12X TIBCO Object Service Broker macros are located in the MACRO data set. These data sets are distributed with TIBCO Object Service Broker. The HRNSEC12 macro contains instructions for installing both HRNSEC12 and IDMSS12X. IDMSS12X replaces the existing SVC user exit supplied by CA-IDMS.

Implementing Fail Safe Processing

Fail Safe level-1 processing guarantees consistency when updating both TDS and CA-IDMS data from a single instance of TIBCO Service Gateway for IDMS/DB in a single transaction. At the end of a transaction, the Data Object Broker requests that the Gateway commit outstanding updates. As part of CA-IDMS commit processing, the Gateway initiates another run unit for the transaction database, updates the database to record the successful commit, issues a FINISH for the transaction database run unit, and then issues a FINISH for the original run unit. If the Gateway does not respond to the Data Object Broker in a reasonable amount of time, the transaction is flagged as being in doubt. Locks held on TDS data remain in place until the problem is resolved.

When a connection is re-established between the Data Object Broker and an instance of TIBCO Service Gateway for IDMS/DB with the same configuration as the one that failed, the Data Object Broker asks the Gateway if the in-doubt transaction completed. The Gateway checks the CA-IDMS transaction database to determine this. If CA-IDMS has completed the update, the TDS updates are applied and the locks are released.

You can resolve in-doubt transactions only by starting an instance of TIBCO Service Gateway for IDMS/DB with exactly the same parameter settings as the Gateway in use when the transaction was placed in doubt.

See Also *TIBCO Object Service Broker for z/OS Managing Backup and Recovery* for more information on Fail Safe processing.

Procedure

To implement Fail Safe processing, you must complete the following tasks:

1. Define a transaction database, page 51
2. Extract the CA-IDMS transaction database, page 51
3. Load the CA-IDMS subschema definition into TIBCO Object Service Broker, page 51
4. Define a TIBCO Object Service Broker IDMS/DB transaction table, page 51
5. Define TIBCO Service Gateway for IDMS/DB Fail Safe startup parameters, page 52

These tasks are described in the following sections.

Task A Define a transaction database

To implement Fail Safe level-1 processing, you must define a CA-IDMS transaction database. The DDL required to define the database is located in the CNTL data set, in members XIDMTRX1, XIDMTRX2, XIDMTRX3, and XIDMTRX4.

These members contain definitions to create the following

- Fail Safe record
- Schema
- Subschema
- Physical database

Do not change the CA-IDMS record name; however, you can change the schema and subschema names. If the subschema name is changed, ensure you include the TRXDB parameter (specifying the new subschema name) when the Gateway is started. This transaction table holds a maximum of one record for each combination of TIBCO Service Gateway for IDMS/DB and Data Object Broker.

Task B Extract the CA-IDMS transaction database

Use the CA-IDMS extract program S6BIDUTL or S6BIDU12 to extract the CA-IDMS element, record, set, and index information from the IDD that you defined in Define a transaction database on page 51. For more information, refer to Task A, Extracting CA-IDMS data dictionary information, on page 32.

Task C Load the CA-IDMS subschema definition into TIBCO Object Service Broker

After extracting the information from the IDD, you must load the CA-IDMS subschema definition into TIBCO Object Service Broker. For more information, refer to Task B, Loading a CA-IDMS subschema definition into TIBCO Object Service Broker, on page 34.

Task D Define a TIBCO Object Service Broker IDMS/DB transaction table

After loading the CA-IDMS subschema definition into TIBCO Object Service Broker, you must define a TIBCO Object Service Broker IDMS/DB transaction table to contain the CA-IDMS data in these fields. By default this table is called @IDMFSTRXDB and its fields must match the fields in the CA-IDMS transaction database.

The transaction table can be managed in TIBCO Object Service Broker like any other IDM table. For example, you can write a TIBCO Object Service Broker rule to clean up the Fail Safe database on shutting down the Gateway.

Task E Define TIBCO Service Gateway for IDMS/DB Fail Safe startup parameters

Ensure that the following TIBCO Service Gateway for IDMS/DB startup parameters are included in your startup JCL:

- FSLEVEL=1
- FSTABLENAME=@IDMFSTRXDB

For more information on these TIBCO Service Gateway for IDMS/DB startup parameters, refer to Supplying Service Gateway for IDMS/DB Startup Parameters on page 40.

This default name of the TIBCO Object Service Broker IDMS/DB transaction table (@IDMFSTRXDB) could have changed when you defined it in Define a TIBCO Object Service Broker IDMS/DB transaction table on page 51.

At startup, the Gateway asks the Data Object Broker for the TIBCO Object Service Broker IDMS/DB transaction table definition specified in the FSTABLENAME startup parameter. This table definition is bound for the life of the Gateway and is used at transaction end and at intermediate COMMITs, to update the CA-IDMS transaction database.

Connecting the Gateway to a Windows or Solaris Data Object Broker

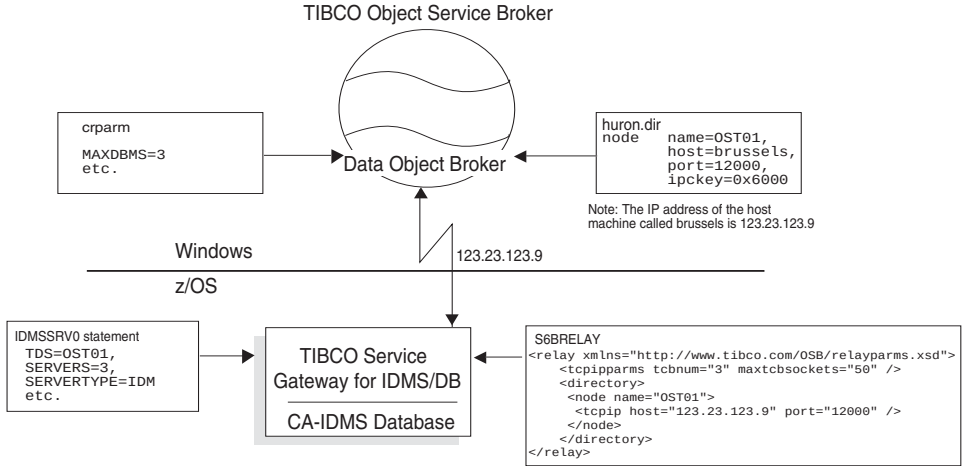
You can configure the Data Object Broker and the Gateway to reside on different domains and operating systems (z/OS, Windows, or Solaris). The Gateway must be in the same domain as the CA-IDMS database system.

The following configuration steps are required to access a Data Object Broker from a different operating environment than your TIBCO Service Gateway for IDMS/DB:

- Configure the TCP/IP connection on the z/OS system where your TIBCO Service Gateway for IDMS/DB and CA-IDMS database reside.
- Configure the TCP/IP connection on the machine where your TIBCO Object Service Broker for Open Systems resides.
- Specify the number of Gateways that can connect to the Data Object Broker.
- Specify the appropriate TIBCO Service Gateway for IDMS/DB parameters.

Sample Configuration

The following diagram shows a sample configuration:



Configure the TCP/IP Connection on z/OS

Prepare the TIBCO Object Service Broker relay file (RELAYCFG member in the CNTL data set). This file associates the TIBCO Object Service Broker communications identifier with the TCP/IP application addressing information.

Sample Relay File assigned to DDNAME S6BRELAY

```
<relay xmlns="http://www.tibco.com/OSB/relayparms.xsd">
    <tcpipparms tcbnum="3" maxtcbsockets="50" />
    <directory>
     <node name="OST01">
      <tcpip host="123.23.123.9" port="12000" />
     </node>
    </directory>
</relay>
```

The element and attribute names in the relay file are case sensitive.

See Also

- *TIBCO Object Service Broker for z/OS Installing and Operating* for detailed information about preparing the TIBCO Object Service Broker relay file.
- *TIBCO Object Service Broker Parameters* for details about the parameters used in the relay file and how to specify them.

Configure the TIBCO Object Service Broker TCP/IP Environment

Add the following parameters for the TCP/IP connection to the Data Object Broker directory file, huron.dir:

name	This must be the same value as the node name set in the relay file described in Configure the TCP/IP Connection on z/OS on page 54.
host	The name of the host machine where the TIBCO Object Service Broker monitor process listens for connections.
port	The number of the TIBCO Object Service Broker monitor socket port.
ipckey	The value of the IPC key.

Specify the Number of Instances of TIBCO Service Gateway for IDMS/DB Connecting to the Data Object Broker

Specify the following value for the MAXDBMS parameter in the crparm file for your Data Object Broker.

MAXDBMS	Used during problem analysis to determine which portion of the Gateway code is in error. Valid values are listed below. The default is 0.

This must be equal to or greater than the value specified in the SERVERS= parameter IDMSSRV0 statement in the startup JCL or IDMSSRV0 SYSIN file.

Specify the Gateway Parameter

Specify the following value for the TDS parameter in the IDMSSRV0 statement in the startup JCL or IDMSSRV0 SYSIN file. Refer to Supplying Service Gateway for IDMS/DB Startup Parameters on page 40 for more details about specifying the gateway parameters.

TDS	This must be the same value as the node name set in the relay file described in Configure the TCP/IP Connection on z/OS on page 54.

See Also

- *TIBCO Object Service Broker for z/OS Installing and Operating* for detailed information about preparing the relay file.
- *TIBCO Object Service Broker Parameters* for details about the parameters used in the relay file and how to specify them.

Connecting the Gateway Using Cross Memory Services

Standard Operation

To run the Gateway and have it use Cross Memory Services:

- The Gateway and the Data Object Broker must be running in the same z/OS image
- The execution libraries must be APF-authorized

  The interface between the Data Object Broker and IDMS/DB requires these libraries to be part of the STEPLIB concatenation:

```
//stepname EXEC PGM=S6BIDMST,PARM=IDMS
//STEPLIB  DD   DISP=SHR,DSN=OSB.V520.AUTH            <-- Authorized
//         DD   DISP=SHR,DSN=CAI.IDMS.R150.LOADLIB       <-- Authorized
//         DD   DISP=SHR,DSN=IDMS.R150.DBA.LOADLIB       <-- Authorized
```

Execution via HRNLIB

In some IDMS/DB environments, the DBA LOADLIB is not authorized. Including the un-authorized library makes the whole job step un-authorized, and communication uses VTAM instead of Cross Memory Services.

If you want to use Cross Memory Services in this situation, supply a HRNLIB DD statement to the Gateway JCL using the procedure shown here:

1. In the EXEC statement, change the PGM= to execute S6BSRV00 (instead of S6BIDMST), with PARM=IDMS.
2. Add a HRNLIB DD statement pointing to your TIBCO Object Service Broker LOAD library. This library must be APF Authorized.

Here is an example of the resulting JCL:

```
//stepname EXEC PGM=S6BSRV00,PARM=IDMS
//STEPLIB  DD   DISP=SHR,DSN=OSB.V520.AUTH            <-- Authorized
//         DD   DISP=SHR,DSN=CAI.IDMS.R150.LOADLIB       <-- Authorized
//         DD   DISP=SHR,DSN=IDMS.R150.DBA.LOADLIB       <-- Not authorized
//HRNLIB   DD   DISP=SHR,DSN=OSB.V520.AUTH            <-- Authorized
```

Verification

To verify that the Gateway is using Cross Memory Services, check the Data Object Broker log for a server-available message. If the message indicates `COMM=XMS`, as shown here, the job is using Cross Memory Services:

```
S6BKC018I-TEST IDMS01   SERVER AVAILABLE, TYPE=IDM, SERVER ID=IDMSSRVR  COMM=XMS
```

Other Operational Procedures

Adding the Gateway Tasks

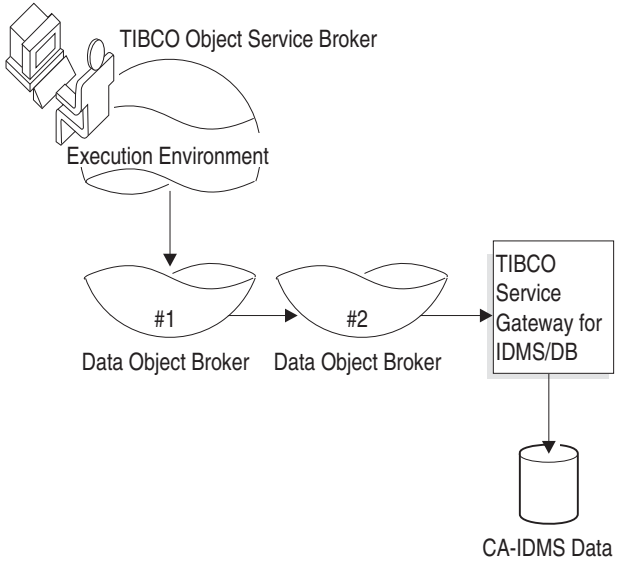
The number of Gateways attached to the Gateway address space is specified in the Gateway startup JCL. You can use one of the following methods to add additional Gateways:

- Shut down the Gateway and start it again with an increased number of Gateways (SERVERS).
- Start another instance of TIBCO Service Gateway for IDMS/DB with the same SERVERID and a different IDPREFIX.

Using Distributed Data with TIBCO Service Gateway for IDMS/DB

Distributed access between TIBCO Object Service Broker and CA-IDMS is permitted subject to requirements of all distributed access. Refer to *TIBCO Object Service Broker Application Administion* and *TIBCO Object Service Broker for z/OS Installing and Operating* for more information on distributed access.

This illustration shows a sample TIBCO Object Service Broker IDMS/DB distributed-data scenario:



Displaying the Status of the Instances of TIBCO Service Gateway for IDMS/DB

You can display the status of all instances of TIBCO Service Gateway for IDMS/DB in an address space using the RESOURCE MANAGEMENT option from the Administration menu. The following screen shows an example of the type of information displayed for resource type IDMS.

```
S6BADM33  HTSTSRV      RESOURCE DETAIL FOR IDMS IDMSSRV    2000JAN02 16:40:52

INTERMEDIATE ROLLBK  N EARLY RELEASE    Y LAST USER REUSE    N COMMIT LEVEL  0
RETRY INTERVAL       0 TP NAME            USER ID PREFIX       FAILURES      0
NODE  MMMSYSTEMA01     INDOUBTS          N                    DELETE

                      CONNECTIONS      IN-USE         TRX          MESSAGE
                     CUR  MAX  LMT   CUR  MAX       COUNT          COUNT
          ONLINE       0    0    0     0    0          0              0
          COMMON       0    0   25     0    0          0              0

SCHEDULE NAME IMSIMSDRASV

         APPLICABLE DAYS          EXCEPTION  START  ONLINE  CONNECTIONS
   MON TUE WED THR FRI SAT SUN    DATE        TIME   ONLY       MAX
    Y   Y   Y   Y   Y   Y                    00:00    N          25




 ENTER-PATHS PF2-TYPE PF4-GROUP PF5-PEER PF9-START PF10-SCHEDULES PF11-UPDATE
THERE ARE NO ACTIVE PATHS TO BE DISPLAYED, REQUEST IGNORED
```

Debugging Rules and Applications

The Rule Debugger helps you identify and fix errors within your application. You can also make and test certain ad hoc changes to your rules. The debugger stops the rules execution at events specified from within the Debug screen.

For CA-IDMS Release 12.0 and greater, you can determine the DML statements being sent to CA-IDMS by your TIBCO Object Service Broker rules by adding a SYSLIST DD NAME for the output in the Gateway startup JCL, and adding the line DMLTRACE=ON to the SYSIDMS SYSIN member specified in the Gateway startup JCL. When you use this parameter, DML statements for all IDMS/DB table accesses are included in the trace. To view only DML statements created by your rules, you require a dedicated Gateway. You can do this by using the SERVERID parameter. To change the server ID of an IDM table definition to an

alternate dedicated server ID, invoke the CHANGE_SERVERID tool from the workbench as follows:

```
CHANGE_SERVERID(table_name,old_serverid,new_serverid)
```

Reporting Problems

Refer to How to Contact TIBCO Support on page xvi for information about reporting problems with Service Gateway for IDMS/DB to TIBCO Support. Have the following information available when reporting a problem:

- CA-IDMS Release and Maintenance Level you are running
- IDM table definitions and sample data
- CA-IDMS schema and subschema definition (section containing CA-IDMS records selected in your IDM tables)
- Bachman diagram for CA-IDMS records accessed by your IDM tables
- TIBCO Service Gateway for IDMS/DB job log and the gateway startup parameters
- DML/TRACE output if running CA-IDMS 12.0

Also, have a listing available of all TIBCO Object Service Broker IDMS/DB control tables:

- @IDMSACCESS(table)
- @IDMSDBNAME(subschema)
- @IDMSELEMENTS(subschema,record)
- @IDMSFIELDS(table)
- @IDMSINDEXES(subschema,set)
- @IDMSRECORDS(subschema)
- @IDMSSETS(subschema)
- @IDMSTABLES (section containing table(s) in question)

See Also

- *TIBCO Object Service Broker Application Administion* and *TIBCO Object Service Broker for z/OS Installing and Operating* for more information on distributed access
- *TIBCO Object Service Broker for z/OS Installing and Operating* for more information on the RESOURCE MANAGEMENT option
- *TIBCO Object Service Broker Programming in Rules* for more information on debugging rules and applications

Chapter 3 Managing IDMS/DB Data Definitions

This chapter provides information to manage TIBCO Object Service Broker IDMS/DB Data Definitions and to use the TIBCO Object Service Broker IDM table.

Topics

Accessing CA-IDMS Data

To access CA-IDMS data from TIBCO Object Service Broker, you must define a TIBCO Object Service Broker table of type IDM. An IDM table can have up to four parameters, up to 16 composite primary keys, one or more fields, and an optional location parameter.

Prerequisite

Before you can define an IDM table, the CA-IDMS subschema must be described to TIBCO Object Service Broker. Refer to Task A, Extracting CA-IDMS data dictionary information, on page 32 for more information.

To define an IDM table, do the following tasks:

1. Task A Invoke the Table Definer, page 63
2. Task B Select a CA-IDMS Subschema, page 67
3. Task C Specify IDM Table Specifications, page 68
4. Task D Select CA-IDMS Records, page 71
5. Task E Specify Record Connections, page 72
6. Task F Define CA-IDMS Elements, page 76
7. Task G Verify the Access Path, page 83

These tasks are described in the sections below.

Task A Invoke the Table Definer

The Table Definer can be invoked from the workbench using either the DT define table option or the primary command field. You can access an existing definition or define a new IDM table.

Accessing Existing Tables

You can display the definition of an existing IDM table from the workbench in one of three ways:

- Type the name of an existing table beside the DT define table option and press Enter to display its definition.
- Type the name of an existing table in the primary command field, for example:

  DT EMPLOYEE_DENTAL<Enter>
- Move the cursor to the DT define table option and press Enter. This displays the Object Manager screen, which contains a list of existing tables.

  Scroll through this list to see which table you require. To select a table, type **S** beside the name and press Enter.

Defining a New Table

To define a new table, complete the following steps:

1. Type the name of a new IDM table beside the DT define table option or in the primary command field.

   Valid entries are a character string of up to 16 characters beginning with a letter (A - Z) or a special character ($ or #), and with more letters, special characters, digits (0 - 9), or underscore characters (_). A table name starting with an @ symbol denotes a table supplied by TIBCO Object Service Broker.

   This displays a TDS definition template.
2. Change the Type field at the top of the screen from TDS to IDM and press Enter.

The initial IDM table definition screen appears, similar to the one shown below:

```
COMMAND==>                          TABLE DEFINITION

      Table: EMPLOYEE_DENTAL  Type: IDM   Unit: USR40


   IDM Table Specification

  Defining SUBSCHEMA:            Run Time SUBSCHEMA:
     IDMS Ready Mode: SR            Optimize Update: N
           Server ID: DEFAULT                DBName:

--------------------------------------------------------------------------------
     Show?         Lvl Record Name       Get Forall Set Name         Opt
--------------- -  --- ---------------   ---  ---   ---------------- --




PFKEYS: 1=HELP 2=DOC 3=SAVE 5=ELEMENTS 6=RECORDS 12=CANCEL 9=SAVE MIN 22=DEL
```

Definition Screen Segments

The initial IDM table definition screen is divided into the following four segments:

Header segment	Specifies the IDM table name, type, and unit. Refer to Modifying the Header Segment on page 65 for information.
Table Specification segment	Creates specifications for the IDM table and specifies IDMS/DB parameters. Refer to , Task B Select a CA-IDMS Subschema, on page 67 for more information.

Record segment	Displays the records that make up the chosen access path for your table definition. These records are for display purposes only; they can be changed only from the CA-IDMS Record and the Set Connections screens. Refer to Task D Select CA-IDMS Records on page 71 and Task E Specify Record Connections on page 72 for more information.
PF Keys segment	Displays the PF keys available from this screen. Refer to Modifying the Header Segment on page 65 for information.

Modifying the Header Segment

The header segment of the initial IDM table definitions screen contains the following fields:

Table	Displays the table name specified when you invoked the Table Definer. You can type in a new name to save the definition of the current table under a new name. Valid Entries: A character string of up to 16 characters beginning with a letter (A - Z) or a special character ($ or #), and continuing with more letters, special characters, digits (0 - 9), or underscore characters (_).
Type	Displays the table type IDM, which you changed in Task A Invoke the Table Definer on page 63
Unit	Displays the user unit associated with the table. The unit marks the table as belonging to a particular application or to a logical unit such as utilities, accounting, or network control. The default unit for your user ID is specified in your TIBCO Object Service Broker user profile. Valid Entries: A character string up to a maximum length of eight characters. These can be provided by your system administrator, for example, ACC.

PF Keys and Primary Commands

You can use the following PF keys or their corresponding primary commands (or their abbreviations) from the initial IDM table definition screen:

PF Key	Primary Command	Description
1	-	Displays corresponding help for the field or screen where your cursor is placed.
2	**DOC**	Displays the Documentation screen, where you can document the table definition. Refer to Appendix A, Documenting IDM Tables, on page 105 for more information.
3	**SAVE**	Saves the definition of the IDM table and returns you to the workbench.
5	**ELEMENTS**	Displays the Elements screen, where you can choose elements for each record in the access path.
6	**RECORDS**	Displays the Records screen, where you can choose records to be part of the access path.
9	**MINIMAL**	Displays the Elements screen so that you can create a minimal definition.
12	**CANCEL**	Cancels changes to the definition and returns you to the workbench.
13	**PRINT**	Prints the existing table definition. You remain in the Table Definer.
22	**DELETE**	Deletes the definition of the IDM table. When you press Enter, you are prompted to confirm the deletion.

See Also

- *TIBCO Object Service Broker Getting Started* for more information on the Object Manager
- *TIBCO Object Service Broker Shareable Tools* for more information on copying TIBCO Object Service Broker objects

Task B Select a CA-IDMS Subschema

To select a CA-IDMS subschema from the initial IDM table definition screen, place your cursor beside the **DEFINING SUBSCHEMA** field and press PF1. This displays a list with information on all CA-IDMS subschemas loaded into TIBCO Object Service Broker from the CA-IDMS Data Dictionary. Refer to Task A, Extracting CA-IDMS data dictionary information, on page 32 for more information.

Sample CA-IDMS Subschema

```
------------------------------  FIELD  Level Help  --------------------------
COMMAND ==>                                                      Scroll:

DEFINING  SUBSCHEMA:   This is the name of the IDMS subschema that is used to
define  the  table.   It  is  required  for  all  IDMS  tables except minimal
definitions.


   SUBSCHEMA DATE_LOADED NUM_RECORDS NUM_SETS NUM_INDEXES NUM_ELEMENTS
_ EMPOMSUB   1998-01-14           1        4           6            5
_ EMPSS01    1998-07-11          14       24          18          391
_ EMULSUBS   1997-09-08           1        0           0           10
_ FS6000DD   1997-09-08          13       15          10          384
_ FS6000SS   1996-09-08          40       57          63          833
_ HRNSS001   1996-10-19           1        2           2            6
_ IDMSNWKA   1995-08-17         166      306          65         3384
_ OEX001V0   1995-09-08          60       89         569          782
_ OMSUBHRN   1993-03-22           2        3           1           77


S=Select
PFKEYS: 3=RETURN VALUE 12=CANCEL
```

To base your IDM table on any one of these subschemas, type an **S** beside the desired subschema and press PF3. The initial IDM table definition screen appears again with the chosen subschema.

You do not require a subschema for a minimal definition. Refer to Specifying a Location Parameter on page 77 for more information.

Task C Specify IDM Table Specifications

Providing Field Values

After selecting a CA-IDMS subschema, you can provide values to the IDM table specification segment. Provide the necessary entries for the fields listed below. After completing the necessary fields in the table specification segment, press Enter to validate the entries.

IDMS Ready Mode

The CA-IDMS ready mode to use when you access this table in a TIBCO Object Service Broker transaction running in update mode. If the transaction is running in browse mode, shared retrieval (SR) mode is used, regardless of what is specified in this field. The default mode is SR.

The Gateway READYs only those AREAs associated with the CA-IDMS records specified in the IDM table definition.

For subsequent accesses to other tables with different AREA names within the same transaction, a new READY in the same mode as the first READY is issued.

If a CA-IDMS INDEX is used to access a record, and the Gateway receives a '0301' CA-IDMS status code indicating the AREA where the INDEX resides is not READY'd, the Gateway issues a READY for the INDEX's AREA and continues processing. You can choose from the following modes:

SR	Shared Retrieval
SU	Shared Update
PU	Protected Update
EU	Exclusive Update

The value of this field can be overridden at runtime. Refer to Dynamically Changing the Gateway Parameters on page 47 for more information.

Server ID

The ID for the instance of, or group of instances of, TIBCO Service Gateway for IDMS/DB to use when accessing the table you are defining. This identifies a group of the Gateways with common characteristics, and must match the SERVERID parameter specified in the Gateway JCL. Refer to Supplying Service Gateway for IDMS/DB Startup Parameters on page 40 for more information. This is a required field for all IDM table definitions, except minimal definitions. The default is DEFAULT. Valid entries are character strings of up to eight characters.

The value of this field and the SERVERID parameter can be overridden at runtime. Refer to Dynamically Changing the Gateway Parameters on page 47 for more information.

Run Time SUBSCHEMA

Provides a different subschema to use at runtime for more efficient access. The runtime subschema does not have to be loaded into TIBCO Object Service Broker. If you do not specify a runtime subschema, the same subschema used for the definition is used by default.

The value of this field can be overridden at runtime. Refer to Dynamically Changing the Gateway Parameters on page 47 for more information.

Optimize Update

When you request CA-IDMS data with a FORALL statement, the Gateway retrieves all valid record occurrences and sends them to the Data Object Broker in variable length buffers of up to 4 KB each until the request is complete.

When this field is set to N, if REPLACE or DELETE statements for the same IDM table are embedded in the FORALL loop, the Gateway uses the DBKEY of each record in the definition to re-establish currency on the correct record and then REPLACE and/or DELETE it.

When this field is set to Y, the Gateway returns one occurrence at a time to the Data Object Broker instead of 4 KB buffers, removing the need to re-establish currency.

While this option removes the need to re-read the record occurrences to be updated, it significantly increases message traffic between the Gateway, the Data Object Broker, and the Execution Environment, and therefore must be carefully considered.

The value of this field can be overridden at runtime. Refer to Dynamically Changing the Gateway Parameters on page 47 for more information.

You should set Optimize Update to Y only if the Gateway, the Data Object Broker, and the Execution Environment communicate using Cross Memory Services.

DB Name

Type the physical database name to use when accessing this table in a TIBCO Object Service Broker transaction. Your system administrator can define a default database name for your user ID, described in Providing Default Database Names for Users on page 38. This name overrides entries in this field for your user ID. The database name for the first IDM table accessed in a transaction is used for all subsequent accesses to CA-IDMS for the life of the TIBCO Object Service Broker transaction.

The value of this field can be overridden at runtime. Refer to Dynamically Changing the Gateway Parameters on page 47 for more information.

Task D Select CA-IDMS Records

After completing the necessary fields in the IDM table specification segment, you must select CA-IDMS records for your definition. Press PF6 to display the CA-IDMS RECORDS screen, shown below. The sample screen below lists all records defined for the selected subschema EMPSS01.

```
COMMAND==>                          TABLE DEFINITION

      Table: EMPLOYEE_DENTAL  Type: IDM   Unit: USR40

--------------------------------------------------------------------------------
Select ONE starting record from SUBSCHEMA > EMPSS01
--------------------------------------------------------------------------------
N <- Select all fields from this record?
_ COVERAGE          (VIA)  _ DENTAL-CLAIM     (VIA)  _ DEPARTMENT       (CALC)
_ EMPLOYEE          (CALC) _ EMPOSITION       (VIA)  _ EXPERTISE        (VIA)
_ HOSPITAL-CLAIM    (VIA)  _ HRNTRXDB         (CALC) _ INSURANCE-PLAN   (CALC)
_ JOB               (CALC) _ NON-HOSP-CLAIM   (VIA)  _ OFFICE           (CALC)
_ DENTAL            (CALC) _ STRUCTURE        (VIA)



S-Select starting record
PFKEYS: 1=HELP ENTER=SELECT 12=RETURN 4=CANCEL
```

Type an **S** beside the record that you want as the starting record in your access path and press Enter. The Set Connections screen for that record appears. For example, if you choose the EMPLOYEE record as the starting record, a screen similar to the one shown in Task E Specify Record Connections on page 72 appears.

You can also type a Y in the **Select all fields from this record?** field. This automatically includes all elements from the record in the table definition. The default is N.

Task E Specify Record Connections

Set Connections Screen

You use the first display of the Set Connections screen, shown below, to choose how to access the first record in the access path. The sample screen below displays four possible access choices via:

- The EMP-NAME-NDX Index
- The DEPT-EMPLOYEE set
- The OFFICE-EMPLOYEE set
- An area sweep

```
COMMAND==>                          TABLE DEFINITION

      Table: EMPLOYEE_DENTAL  Type: IDM   Unit: USR40
-------------------------------------------------------------------------------
   Record(s) connected to >
-------------------------------------------------------------------------------
   Record Name       Set Name         Options
   ---------------  ----------------   --     -------
_  SR7               EMP-NAME-NDX       OA     INDEX
_  DEPARTMENT        DEPT-EMPLOYEE      OA     OWNER
_  OFFICE            OFFICE-EMPLOYEE    OA     OWNER
_  ** AREA SWEEP **                             AREA

Y <- Select all fields from this record?
-------------------------------------------------------------------------------
     Show?         Lvl Record Name       Get Forall Set Name        Opt
--------------- -  --- ---------------   ---  ---   ---------------- --
                    1  EMPLOYEE          OC   OFA

PFKEYS: 1=HELP ENTER=SELECT 5=ELEMENTS 4=CANCEL 18=-RECORD 9=CONNECT 12=RETURN
Select INDEX, SET or AREA you want to navigate
```

When accessing the first record via a set (second and third options), ensure you established the proper record currencies before accessing this table.

Procedure

Type an **S** beside the access method you want to use to access the first record in the access path, for example, DEPARTMENT, and press Enter. The **Set Name** field (access method) for the first record in the access path appears. The subsequent screens show the next set of records that can be added to your access path. You have the following two options:

- Add as many records as required.
- Stop at one record pressing PF5.

After adding the required number of records, press PF5 to move to the Elements screen. Refer to Task F Define CA-IDMS Elements on page 76 for more information on selecting elements.

Field Values

The Set Connection screen is divided into two sections. The top section contains the following fields:

Record Name	The name of the CA-IDMS record attached to the current record.
Set Name	The name of an index for a record or the name of the set that connects two records. If the first record in the access path has more than one index, all index names are listed.
Options	The Gateway uses this field to determine whether explicit **CONNECT** or **DISCONNECT** commands are required. Possible set membership options are:
	MA Mandatory Automatic OA Optional Automatic MM Mandatory Manual OM Optional Manual
Owner or Member Field	Lists whether the record is an index, owner, area, or a member of the current record.
Select all fields from this record?	The default Y automatically includes all elements from the record in the table definition. If set to N, you can selectively include elements from the record.

The bottom section of the screen displays records that you chose previously as part of the access path. The last record in the list is the current record. This section contains the following fields:

Show?	Contains one of the following entries if there is a possibility of empty or duplicate records. If you type a Y beside this entry, the field means the following: Show if Set Empty: Include the record even if the set is empty. The fields associated with the empty set have NULL values. Show if No Owner: Include the record even if there is no owner. The fields associated with the owner have NULL values. Show if Duplicates: Include duplicate records.
Lvl	The order in which the record is accessed.
Record Name	Name of the CA-IDMS record. The last name in the list is the current record.
Get	The CA-IDMS directive code used to retrieve the CA-IDMS record when a TIBCO Object Service Broker GET is issued. The codes are: OC OBTAIN CALC OFS OBTAIN FIRST IN SET OFA OBTAIN FIRST IN AREA OO OBTAIN OWNER
Forall	The CA-IDMS directive code that is used to retrieve the CA-IDMS record when a TIBCO Object Service Broker FORALL is issued. The codes are: OC OBTAIN CALC OFS OBTAIN FIRST IN SET OFA OBTAIN FIRST IN AREA OLA OBTAIN LAST IN AREA OLS OBTAIN LAST IN SET OO OBTAIN OWNER

Set Name	Displays the name of the set of the record.
Opt	Displays the membership options for the listed records.

PF Keys

You can use the following keys from the IDMS/DB Set Connections screen:

Enter	Selects your chosen records.
PF1	Displays the corresponding help for the field or screen where your cursor is placed.
PF3	Returns you to the previous screen.
PF4	Cancels changes to the definition and returns you to the workbench.
PF5	Saves the currently selected access path and displays the Elements screen.
PF9	Displays optional sets that are available.
PF12	Returns you to the previous screen.
PF18	Removes the record where the cursor is placed from the access path, moving you up one level in the access path.

Task F Define CA-IDMS Elements

After choosing CA-IDMS records, you must define elements for each of the CA-IDMS records. This is done from the IDMS/DB Elements screen, shown in the following illustration. You access this screen pressing PF5 from the IDMS/DB Set Connections screen.

This Table Definer screen displays a list of IDMS/DB elements under Field Name:

```
COMMAND==>                          TABLE DEFINITION

      Table: EMPLOYEESS        Type: IDM   Unit: PHG00

  Location Parm    Typ Syn Len Dec Default          '      Event Rule   Typ Acc
  ---------------- --- --- --- -- ----------------  '   ---------------- -   -
  LOCATION          I   C  16  0                    ' _
                                                    ' _
                  |------------- Metadata Definition -------------- | IDMS Def >
     Field Name    T S   Len  Dc Reference        Default           |Ky Ix S  Le
- ---------------- - -- ----- -- ---------------- ------------------|-- -- - ---
_ EMPLOYEE                                                         |
K EMP_ID           B      2  0                                     | 1    M
_ EMP_FIRST_NAME   C     10  0                                     |    2 C
_ EMP_LAST_NAME    C     15  0                                     |    1 C
_ EMP_STREET       C     20  0                                     |      C
_ EMP_CITY         C     15  0                                     |      C
_ EMP_STATE        C      2  0                                     |      C
_ EMP_ZIP_FIRST_FI C      5  0                                     |      C
_ EMP_ZIP_LAST_FOU C      4  0                                     |      C
_ EMP_PHONE        P      6  0                                     |      M
K-Key  P-Parm  S-Select
PFKEYS: 1=HELP 3=SAVE 12=RETURN 4=CANCEL
```

The IDMS/DB Elements screen is divided into three segments:

Header segment	Displays the IDM table name, type, and unit that you specified in Modifying the Header Segment on page 65.
Location Parm & Event Rule segment	Provides an area to specify location parameters and event rules for the IDM table. Refer to Specifying a Location Parameter on page 77 and Specifying Event Rules on page 78 for more information.
Element segment	Provides an area to select elements to be changed from the CA-IDMS Record and the Set Connections screens. Refer to Selecting Elements on page 79 for more information.

The procedures for the Location Parm, Event Rule, and Element segments are described below.

Specifying a Location Parameter

You use this section of the Table Definition screen to define a location parameter for an IDM table. You use a location parameter to access CA-IDMS data through a peer Gateway associated with another Data Object Broker (remote node). If you do not need to access remote data, blank out the location parameter name to delete the parameter. If you always access CA-IDMS data remotely, the node from which you request the access can have either a minimal or a full definition.

Minimal Definition

A minimal definition consists of the following:

- The table name, which must be the same at both locations.
- The location parameter, which must be the same at both locations. The name of the remote node where the full definition is located must be supplied in the **Default** field, **Src** field, or **Src** and **Sourcename** field.

The table type specified in a minimal definition does not have to match the table type of the full definition on the remote node. A minimal definition with a location parameter means you always access data at a remote node.

Full Definition

A full table definition with a location parameter means you can access data at either the local or the remote node. Define data parameters on the full definition, not a minimal definition.

The table type of the full definition must match the data on the local node. For example, a full definition of type TDS used to access data on the local node can also be used to access an IDM table with the same name on a remote node.

See Also *TIBCO Object Service Broker Managing Data* for more information on location parameters and minimal table definitions.

Specifying Event Rules

Event rules enable you to validate data and automatically trigger other events based on specific update and/or retrieval access to IDM tables. Event rules are always called when the table is accessed in the type of access specified. All rules applying to a specific access are executed in the order they appear in the scrollable Event Rule segment of the TIBCO Object Service Broker IDMS/DB Elements screen.

Defining Fields

You define the fields in the Event Rule segment as follows:

Typ Specifies the type of the event rule as follows:

V Validation Rule

No database updates are allowed during the validation process. The rule must be a function that returns Y (yes) if the validation is successful, N (no) if the validation is not successful, or a message explaining why it is not successful.

T Trigger Rule

There are no restrictions on coding, other than the rule must not be a function, it cannot change the contents of the triggering row, and it cannot use the TRANSFERCALL statement. Nested triggers are possible.

Acc The type of access, or manipulation, to be performed on the data, causing the event to be executed.

Validation Rules	Trigger Rules
W - Any write (insert, replace, delete)	W- Any write (insert, replace, delete)
I - Only insert	I - Only insert
R - Only replace	R - Only replace
D - Only delete	D - Only delete
	G - Any retrieval

See Also *TIBCO Object Service Broker Managing Data* for more information on specifying event rules.

Selecting Elements

The Element segment is divided into two areas: the Metadata Definition and the IDMS Definition areas. You can change any of the fields in the Metadata Definition area. The IDMS Definition area is for display purposes only.

Metadata Definition Area

The Metadata Definition area contains the following fields:

Field Name	The default field name. Each CA-IDMS element name is converted to a valid TIBCO Object Service Broker name. Refer to Conversion of Record and Element Names on page 37 for details about how element names are converted to field names. You can type over an entry in this column with a new name to uniquely identify the field within the IDM table. You can use the same name as a field in another table. If you are moving data between this table and another table, giving fields the same names simplifies the process. Valid Entries: A character string (unique to the IDM table definition) of up to 16 characters beginning with a letter (A - Z) or a special character (@, $, or #), and continuing with more letters, special characters, digits (0 - 9), or underscore characters (_).
T	Displays the TIBCO Object Service Broker semantic data type of the element. For valid semantic data types, refer to the *TIBCO Object Service Broker Programming in Rules* manual.
S	Displays the TIBCO Object Service Broker syntax of the element. The Table Definer also automatically assigns the TIBCO Object Service Broker length for each element. Refer to Conversion of Data Types on page 37 for more information.
Len	Displays the maximum number of character positions (including the decimal point, the mantissa, and others) that a value in the field of this element can occupy.
Dc	Displays the number of decimal places of the element.

Reference	Supply the name of a reference table for the field if required. Refer to the *TIBCO Object Service Broker Managing Data* manual for more information.
Default	Supply a default value for the field if required. This default value is used if there is no other value specified for the field.

IDMS/DB Definition Area

The IDMS/DB Definition area contains the following fields for display purposes only:

Ky	Indicates if the field is a single primary key or part of a composite primary key of the record. A single primary key is represented with a 1. Composite primary keys are numbered in the order in which the primary key is composed. A D in this field indicates that the element is a DB key, a unique identifier for every record. This key is automatically appended as the last element of each record, and assigned the same name as the record. The DB key must be included as an element in the record. **Note** You can select this element as the primary key.
IX	Indicates if the element is part of a secondary index. The numbering of these fields indicates the order of the fields that comprise the secondary index.
S	Indicates the syntax of the IDMS/DB element.
Len	Indicates the length of the IDMS/DB element.
Dc	Indicates the number of decimal places in the IDMS/DB element **Note** You must press PF11 to view this field.
BOM	Displays a Y if the element is from the second occurrence of the record in the table of a Bill of Materials structure.
Record Name	Displays the name of the record to which the element belongs.
Element Name	Displays the element name contained in the record listed in the **Record Name** field. The last element of each record is a DB key that is automatically created and used to uniquely identify the record.

Using Line Commands

The following line commands are valid from the IDMS/DB Elements screen:

K	Elements that are to be the primary key fields. Each IDM table definition must have a primary key. The Table Browser supports up to eight composite primary keys. Use rules to access IDM tables with more than eight primary keys. Definitions Without Parameters: The primary key must be defined on elements from the first record in the access path. If the first record in the access path has a CALC key or index, you should select only elements of the CALC key or index as the primary key. If the first record in the access path does not have a CALC key or index, you can select any element as the primary key, up to 16 elements. Definitions with Parameters: You can select only one element from the second-level record in the access path as the primary key. A primary key and a parameter cannot be selected from the same record; therefore, definitions that have only one record cannot be parameterized. **Note** You cannot define a primary key on an element that is in a record accessed using an OBTAIN OWNER (OO) directive.
P	You can select only CALC keys as parameters, and parameters can be defined only on the first record in the access path. Since a parameter and a primary key cannot be selected from the same record, definitions that have only one record cannot be parameterized. CALC keys are identified by a number in the **Ky** field. If the CALC key is composed of more than one element, ensure that you select them all as parameters. You can use up to four parameters for a table definition in TIBCO Object Service Broker; therefore, if the CALC key contains more than four elements, you can select only its elements as primary key fields, or as fields of the table.

S	Elements to be fields in the IDM table. The number of fields you can select is dependent upon the length of the sum of all fields, primary keys, data parameters and control information. This sum must be less than or equal to 3915 bytes. For an explanation of the formula used to calculate the total number of bytes of all fields, refer to the *TIBCO Object Service Broker Programming in Rules* manual. The number of fields you can access is dependent upon the CTABLESIZE Data Object Broker parameter; however, you use the Table Definer to select all IDMS/DB elements as fields. You can use the ESTIMATETBLDFN tool, described in Estimating the CTABLESIZE Parameter on page 45, to estimate the size of this parameter.

You can change existing line commands by typing over them or, if you decide not to include a currently selected IDMS/DB element, you can de-select it by typing a blank over the line command. Press PF3 to return to the initial IDM table definition screen where you can verify your access path.

Elements Screen PF Keys

You can use the following keys from the IDMS/DB Elements screen:

Enter	Validates the data on the screen.
PF1	Displays the corresponding help for the field or screen where your cursor is placed.
PF3	Takes you to the Verify Access screen.
PF4	Cancels changes to the definition and returns you to the workbench.
PF12	Returns you to the previous screen.

Task G Verify the Access Path

IDM Table Definition Screen

When you are done selecting all records and elements, the initial IDM table definition screen, similar to the one shown below, appears for you to verify your access path to the CA-IDMS database.

```
COMMAND==>                          TABLE DEFINITION

      Table: EMPLOYEE_DENTAL  Type: IDM   Unit: USR40


   IDM Table Specification

  Defining SUBSCHEMA: EMPSS01    Run Time SUBSCHEMA: EMPSS01
     IDMS Ready Mode: SR            Optimize Update: N
           Server ID: DEFAULT                DBName:

-------------------------------------------------------------------------------
     Show?          Lvl Record Name       Get Forall Set Name         Opt
--------------- -   --- ---------------   ---  ---   ---------------- --
                     1  EMPLOYEE          OC   OFS   DEPT-EMPLOYEE    OA
                     2  COVERAGE          OFS  OFS   EMP-COVERAGE     MA
IF SET EMPTY -> _    3  DENTAL-CLAIM      OFS  OFS   COVERAGE-CLAIMS  MA



PFKEYS: 1=HELP 2=DOC 3=SAVE 5=ELEMENTS 6=RECORDS 12=CANCEL 9=SAVE MIN 22=DEL
```

Verifying Access Paths

The initial IDM table definition screen also displays the navigation to be used to retrieve CA-IDMS data using this table definition. The only way to change the access path is to change the order of the selected CA-IDMS records and/or change the elements selected as parameters or primary keys. The only fields you can change from this screen are:

- The **Show?** field (described in Field Values on page 73). Refer to Mapping Considerations on page 85 for more information on IDM table definitions.
- The **GET** and **FORALL** fields. Valid values for both fields are:

OFS	OBTAIN FIRST IN SET
ONS	OBTAIN NEXT IN SET
OPS	OBTAIN PREVIOUS IN SET
OLS	OBTAIN LAST IN SET

If you are satisfied with your IDM table definition, press PF3 to save the definition. If you do not want to save your changes, press PF12 to cancel them.

Mapping Considerations

In choosing an optimal access path, it is critical that the person defining IDM table definitions be familiar with the structure of the CA-IDMS database.

Effects of Record Selection

Consider the following points when selecting CA-IDMS records:

- Determine the data you require and an optimal access path to retrieve that data. Create an IDM table definition to match your requirements or review existing definitions for a match. Do not use existing definitions if they contain extra records for the following reasons:

  Unless the **Show?** field on the Set Connections screen is set to Y, when more than one record is defined in an IDM table definition, the Gateway returns data only when all record types exist in the specified sets. For example, if the EMPLOYEE and COVERAGE records are defined in an IDM table, only EMPLOYEEs that have COVERAGE are returned. EMPLOYEEs that do not have COVERAGE cannot be retrieved with this definition. Refer to Task G Verify the Access Path on page 83 for more information on the **Show?** field.

  When the extra member records do exist, they all must be retrieved and returned to TIBCO Object Service Broker since the table definition requested them.

- When the required access path contains two CALC records, be sure to examine your application retrieval needs. If most retrieval requests specify a value for the CALC key of both CALC records, consider which record should come first in the access path.

  Consider a table that has OFFICE (CALC) and EMPLOYEE (CALC) defined in an access path. To find a specific EMPLOYEE in a specific OFFICE, the Gateway CALCs to the OFFICE record and searches the OFFICE-EMPLOYEE set until the specified EMPLOYEE is found. If the EMPLOYEE record were the first in the access path, the Gateway could CALC to the specified EMPLOYEE and do an OBTAIN OWNER to retrieve the OFFICE.

- Ensure VIA records are accessed through CALC owner records to avoid area sweeps. VIA records that have system-owned indexes can be accessed by that index.

Effects of Element Selection

Consider the following points when selecting elements:

- Include IDMS/DB elements that are part of an index or set in the table definition. If selection criteria are specified for this type of field, the Gateway can use the field for positioning and walk the set from that point. Refer to Accessing TIBCO Object Service Broker IDM Tables on page 88 for more information.
- The number of elements that you select as fields from each record in the table definition is used by the Gateway to determine whether a STORE should be done for that record when you issue an INSERT statement. Refer to Insert Processing on page 99 for more information.

Chapter 4 Processing Data

This chapter show you how to access the TIBCO Object Service Broker IDM tables.

Topics

Accessing TIBCO Object Service Broker IDM Tables

When you access CA-IDMS data from TIBCO Object Service Broker, the following occurs:

- TIBCO Object Service Broker requests are translated into CA-IDMS Data Manipulation Language (DML) statements.
- IDMS/DB element data types are translated to the TIBCO Object Service Broker field types defined in the IDM table.

You can access the data using either of the following methods:

- The Table Browser
- The rules language

These methods are discussed in the following sections.

Using the Table Browser

You can use the Table Browser to browse a defined IDM table by typing the table name next to the BR browse table option, for example:

```
OFFICE_EMP('002')<Enter>
```

In this example, `OFFICE_EMP` is the name of the IDM table and `002` is the `OFFICE_CODE` parameter value. The CA-IDMS data is presented to you in the TIBCO Object Service Broker table format, as shown in the following sample screen:

CA-IDMS Data Presented in TIBCO Object Service Broker Table Format

```
BROWSING TABLE   :  OFFICE_EMP(002)
 COMMAND ==>
                                                                  SCROLL: P
   EMP_ID EMP_FIRST_NAME  EMP_LAST_NAME       EMP_STREET          EMP_CITY
_ ------   ----------    --------------- -------------------- ---------------
_     69   JUNE          BLOOMER         14 ZITHER TERR       LEXINGTON
_    119   CHARLES       BOWER           30 RALPH ST          WELLESLEY
_     81   TOM           FITZHUGH        450 THRUWAY ST       MANSFIELD
_     45   GEORGE        FONRAD          99 VERDE ST          STOUGHTON
_     53   ROBIN         GARDNER         68 75TH ST           LOWELL
_     30   HENRIETTA     HENDON          16 HENDON DR         WELLESLEY
_    100   EDWARD        HUTTON          781 CROSS ST         MELROSE
_     51   CYNTHIA       JOHNSON         17 MANIFESTO DR      WALPOLE
_     49   DOUGLAS       KAHALLY         56 SPELLING AVE      READING
_     67   MARIANNE      KIMBALL         561 LEXINGTON AVE    LITTLETON
_    106   DORIS         KING            716 MORRIS ST        MELROSE
_    127   CAROL         MCDOUGALL       19 URITOP DR         WELLESLEY
_    101   BRIAN         NICEMAN         60 FLORENCE AVE      MELROSE
_     91   MADELINE      ORGRATZI        67 RAINBOW DR        KENDON
_    149   LAURA         PENMAN          45 THRUSH LN         WALTHAM

PFKEYS: 1=HELP 5=FIND NEXT 9=RECALL 18=EXCLUDE 13=PRINT 3=END 14=EXPAND
```

Exceptions

You can browse an IDM table in the same way you would browse any other TIBCO Object Service Broker table, with the following exceptions:

- If your table definition contains fields of syntax C or V that are longer than 260 bytes, do the following:

  Use the Single Occurrence Editor (SOE) from the Table Editor to view them.

  Use **`SELECT LIKE`** instead of **`SELECT`** to access fields of this length.

- Do not use the Single Occurrence Editor to edit rows in IDM tables because primary key uniqueness cannot be guaranteed.
- If your transaction is running in update mode, the Ready Mode specified in the IDM table definition, not TIBCO Object Service Broker, determines locking of CA-IDMS data. However, if your transaction is running in browse mode, a shared retrieval (SR) Ready Mode is automatically used, regardless of what is specified in the first IDM table accessed in the transaction. No locks are taken in browse mode and you cannot update the data.

See Also *TIBCO Object Service Broker Managing Data* for more information on using the Table Browser.

Using Rules

Accessing CA-IDMS data using the rules language is similar to accessing TIBCO Object Service Broker data. The main difference is in the way CA-IDMS uses your selection criteria in conjunction with the IDM table definition to interpret the request. Use the following sections in conjunction with *TIBCO Object Service Broker Programming in Rules*, which describes TIBCO Object Service Broker rules language statements and coding.

Transaction Processing

If you issue a TIBCO Object Service Broker EXECUTE statement within a main (parent) transaction, it creates another transaction stream (child transaction), to a maximum of ten streams. The number of streams allowed in a transaction depends on the TRANMAXNUM Execution Environment parameter, which has a default of nine streams. Each transaction stream in TIBCO Object Service Broker accessing CA-IDMS data requires its own task.

Ensure that your system administrator is aware of the number of tasks required to accommodate all transaction streams accessing CA-IDMS data in a single transaction.

Using TRANSFERCALL or DISPLAY & TRANSFERCALL statements in a rule minimizes the Gateway tasks and reduces the possibility of CA-IDMS locking contention.

When a TIBCO Object Service Broker transaction runs in browse mode, a shared retrieval (SR) Ready Mode is automatically used, regardless of what is specified in the IDMS Ready Mode field of the first IDM table accessed in the transaction, and no locks are taken on the data. However, locks are taken on the data for transactions running in update mode. This is determined by the Ready Mode specified in the IDM table definition of the first definition accessed in the transaction.

Transaction Limitations

The number of IDM tables you can access per transaction depends on the POOLSIZE gateway parameter and the CTABLESIZE Data Object Broker parameter. For more information, refer to Supplying Service Gateway for IDMS/DB Startup Parameters on page 40.

If you use the default parameter values, you can access at least 16 IDM tables per transaction; maybe more, depending on the size of the IDM table definitions. Refer to Estimating the CTABLESIZE Parameter on page 45 for more information.

The following sections outline differences encountered while using rules and explain how your rule selection criteria translate into DML access statements.

See Also *TIBCO Object Service Broker Parameters* about the CTABLESIZE Data Object Broker parameter and the TRANMAXNUM Execution Environment parameter.

GET Processing

A GET statement retrieves the first occurrence in the IDM table that satisfies the specified selection criteria.

A GET...ORDERED statement must retrieve all CA-IDMS data that satisfies the selection criteria and SORT it in the Execution Environment before returning the first occurrence that meets the selection criteria.

The first record defined in the access path is retrieved from CA-IDMS based on the selection criteria and whether the table is parameterized. The remaining records defined in the access path are retrieved from CA-IDMS based on the set specified in the table definition and on selection criteria specified for elements of the remaining records.

Retrieving the First Record Defined in an Unparameterized Table

The first record defined in the access path of a non-parameterized table is retrieved as described in the following table:

Selection Criteria Specified	Retrieval Process
Equality operator specified on a CALC key element that is defined as the TIBCO Object Service Broker primary key.	OBTAIN CALC.
Equality operator specified on an index or sort key element.	Direct Access using OBTAIN sort key.
No selection criteria specified.	OBTAIN FIRST IN SET or OBTAIN FIRST IN AREA.
• >, <, >=, or <= operators specified on a sort key element or on part of a composite sort key • Equality operator specified on part of a composite sort key or index	Position within the index or set and walk the index or set until the first record occurrence that satisfies the selection criteria is found.

Selection Criteria Specified	Retrieval Process
• Selection criteria specified with an OR condition • Selection criteria contains the LIKE operator • Selection criteria specified on an element that is not a sort key or index • Selection criteria specified on a CALC key that is not defined as the TIBCO Object Service Broker primary key	Walk the set or sweep the area until the first record occurrence that satisfies the selection criteria is found.

When selection criteria are specified for both non-key IDMS/DB elements and IDMS/DB key elements (that is, CALC, sort key, and index elements), the Gateway does the following:

- Retrieves the record occurrence using the most efficient retrieval process as described in Retrieving Remaining Parameterized Table Records below
- Evaluates the selection criteria for the non-key elements after the record occurrence is retrieved

Retrieving the First Record in a Parameterized Table

A GET request on a parameterized table always results in an OBTAIN CALC request being sent to CA-IDMS to access the first record defined in the IDM table definition.

Retrieving Remaining Parameterized Table Records

The remaining member records defined in the access path are retrieved as described in the following table for both non-parameterized and parameterized tables:

Selection Criteria Specified	Retrieval Process
• No selection criteria specified • Selection criteria specified with an OR condition • Selection criteria specified on an element that is not a sort key • Selection criteria containing the LIKE operator	Walk the set until the first record occurrence that satisfies the selection criteria is found.
Equality operator specified on a sort key element.	Direct Access using OBTAIN sort key.
>, <, >=, or <= operators specified on a sort key element or part of a composite sort key.	Position within the set and walk the set until the first record occurrence that satisfies the selection criteria is found.

When selection criteria are specified for both IDMS/DB sort key elements and non-key IDMS/DB elements, the Gateway does the following:

- Retrieves the record occurrence using the most efficient retrieval process as described in the above table
- Evaluates the selection criteria for the non-key elements after the record occurrence is retrieved

If more than one set is included in the access path, the Gateway positions itself within each set, where possible, based on the points outlined in the above table.

Remaining owner records defined in the access path for both non-parameterized and parameterized tables are retrieved with an OBTAIN OWNER request using the specified set name. Selection criteria specified on elements of the owner record are evaluated after the owner is retrieved.

FORALL Processing

A FORALL statement is a looping construct that processes a set of occurrences. The body of the loop consists of the statements to be executed for each occurrence satisfying the selection criteria. FORALL statements can be nested provided that they refer to different table names.

Rows are returned to TIBCO Object Service Broker in the order in which CA-IDMS passes them. If you require a different order, you must include an ORDERED clause in your FORALL statement. TIBCO Object Service Broker orders only rows specified in the selection criteria.

The first record defined in the access path is retrieved from CA-IDMS based on the selection criteria specified and whether the definition is parameterized. The remaining records defined in the access path are retrieved from CA-IDMS based on the set specified in the table definition and on selection criteria specified for elements of the remaining records.

Non-parameterized Table Access

The first record defined in the access path of a non-parameterized table is retrieved as described in the following table:

Selection Criteria Specified	Retrieval Process
Equality operator specified on a CALC key element defined as the TIBCO Object Service Broker primary key.	OBTAIN CALC.
>, <, >=, <=, or = operators specified on a sort key element or on part of a composite sort key.	Position within the index or set and walk index or set until all record occurrences satisfying selection criteria are found.
• Selection criteria specified with OR condition • Selection criteria containing the LIKE operator • Selection criteria specified on an element that is not a sort key or index • Selection criteria specified on a CALC key that is not defined as the TIBCO Object Service Broker primary key	Walk the set or sweep the area until all record occurrences that satisfy the selection criteria are found.

Selection Criteria Specified	Retrieval Process
No selection criteria specified.	Sweep the area or index.

When selection criteria are specified for both non-key IDMS/DB elements and key IDMS/DB elements (that is, CALC, sort key, and index elements), the Gateway does the following:

- Retrieves all record occurrences using the most efficient retrieval process as described in the above table
- Evaluates the selection criteria for the non-key elements when the record occurrences are retrieved

Parameterized Table Access

A FORALL request on a parameterized table always results in an OBTAIN CALC request being sent to CA-IDMS to access the first record defined in the IDM table definition.

Retrieving the Remaining Records

The remaining member records defined in the access path are retrieved as described in the following table for both non-parameterized and parameterized tables:

Selection Criteria Specified	Retrieval Process
• No selection criteria specified • Selection criteria specified with an OR condition • Selection criteria specified on an element that is not a sort key • Selection criteria containing LIKE operator	Walk the set until all record occurrences that satisfy the selection criteria are found.
>, <, >=, <=, or = operators specified on a sort key element or part of a composite sort key.	Position within set and walk the set until all record occurrences satisfying selection criteria are found.

When selection criteria are specified for both IDMS/DB sort key elements and non-key IDMS/DB elements, the Gateway does the following:

- Retrieves record occurrences using the most efficient retrieval process as described in the above table
- Evaluates the selection criteria for the non-key elements when the record occurrences are retrieved

If there is more than one set in the access path, the Gateway positions itself within each set, where possible, based on the above table.

Remaining owner records defined in the access path for both non-parameterized and parameterized tables are retrieved with an OBTAIN OWNER request using the specified set name. Selection criteria specified on elements of the owner record are evaluated after the owner is retrieved.

Replace (Update) Processing

A REPLACE request against an IDM table modifies the last TIBCO Object Service Broker IDMS/DB row retrieved.

Processing Considerations

You can modify all elements that were selected from MEMBER records, keeping the following points in mind:

- You use CA-IDMS to modify the CALC key of a record providing the modification does not violate the Duplicates not allowed option. However, modifying a CALC key causes CA-IDMS to create a pointer from the new CALC location back to the physical location of the original record.
- An error occurs if a Duplicates not allowed option is violated when modifying a sort, index, or CALC key.

You cannot modify data elements of owner records that are retrieved with an OBTAIN OWNER directive. However, you can DISCONNECT and/or CONNECT CALC manual or optional owner records by modifying the CALC key of the owner record as follows:

- If the **Show if No owner** field in the Verify Access Path screen is set to Y, you can do one of the following:

  DISCONNECT the owner by setting its CALC key to null and issuing a REPLACE.

  CONNECT the owner record by setting the null owner CALC key to a valid owner CALC key, and issuing a REPLACE.
- Regardless of the setting of the **Show if No owner** field, the Gateway DISCONNECTs an owner and CONNECTs another owner when you change the existing owner record's CALC key to another valid owner record's CALC key, and issue a REPLACE.

Insert Processing

An INSERT request always attempts to STORE a CA-IDMS record. CA-IDMS CONNECTs automatic owner records and the Gateway CONNECTs manual owner records included in the IDM table definition.

Processing Considerations

When an IDM table definition contains multiple CA-IDMS records, excluding owner records with an OBTAIN OWNER directive, the number of records STOREd as a result of an INSERT is dependent on the number of elements (excluding FILLER elements) selected from each record. For example:

- If a definition contains all elements from the OFFICE and EMPLOYEE records, the Gateway STOREs and CONNECTs both records.
- If an IDM table definition contains a parameter on the OFFICE record and all elements of the EMPLOYEE record, the Gateway STOREs only the EMPLOYEE record, CONNECTed to the OFFICE record parameter value.

When an IDM table definition contains owner records with an OBTAIN OWNER directive, the result of an INSERT is:

- The Gateway STOREs fully selected member records
- The Gateway CONNECTs that member to the owner specified providing the specified owner record exists and is a CALC record

For a parameterized IDM table, the parameterized record is retrieved and all fully selected member records are STOREd and CONNECTed to the parameterized record when an INSERT is issued.

A CA-IDMS error occurs if you do *not* include all automatic owner records of the member record to be inserted in your IDM table definition. Manual owner records included in the definition are explicitly CONNECTed to the member record you are inserting.

Delete Processing

A DELETE request against an IDM table deletes the lowest-level member record in the access path providing that record does not have any member records. Records that have an OBTAIN OWNER directive are not deleted.

Processing Considerations

The following two methods can be used to delete records, providing the record to be deleted does not have any members:

- Access the record occurrence you want to delete by issuing a GET or FORALL statement, and then issue a DELETE. Only the last member record defined in the access path is deleted. The definition can be parameterized or non-parameterized and can contain any number of records.
- Issue a DELETE statement specifying the TIBCO Object Service Broker primary key value under the following conditions:

  If the IDM table is non-parameterized, the definition can contain only one record. If the element selected as the TIBCO Object Service Broker primary key is not unique, the first record containing the value specified for the primary key is deleted.

  If the IDM table is parameterized, the definition can contain only the parameterized record and one member record. If the element selected as the primary key is not unique within the set, the first record containing the value specified for the primary key, within the specified set, is deleted.

Error Handling and Recovery

This section describes how the Gateway handles TIBCO Object Service Broker requests with respect to the following:

- Synchronization and recovery
- Error handling

Synchronization and Recovery

If your TIBCO Object Service Broker transaction is running in browse mode, the Ready Mode specified in the first IDM table accessed in the transaction determines how CA-IDMS data is locked. If your TIBCO Object Service Broker transaction is running in browse mode, a shared retrieval (SR) Ready Mode is automatically used, regardless of what is specified in the first IDM table accessed in the transaction. No locks are taken in browse mode and you cannot update the data. A CA-IDMS transaction spans the same length of time as a TIBCO Object Service Broker transaction.

Notes on Synchronization and Recovery

- A normal end of transaction results in a DML FINISH being sent to CA-IDMS.
- A COMMIT request sent to the Gateway results in a DML COMMIT being sent to CA-IDMS.
- A ROLLBACK request sent to the Gateway or a transaction failure results in a DML ROLLBACK being sent to CA-IDMS.
- TIBCO Object Service Broker ROLLBACKs *must* be followed by a transaction end.
- TRANSFERCALL and DISPLAY & TRANSFERCALL statements are allowed after a ROLLBACK.

The COMMITLIMIT exception does not apply to IDM tables. Requests to update CA-IDMS data are processed as they are encountered and are not buffered in the Intent List.

Updating CA-IDMS Data Only

Transactions that update only CA-IDMS data are recoverable under CA-IDMS.

Updating TIBCO Object Service Broker and CA-IDMS Data

TIBCO Object Service Broker provides a method of ensuring data integrity when a TIBCO Object Service Broker transaction updates both CA-IDMS and TIBCO Object Service Broker data in the same transaction. This method is referred to as Fail Safe level-1 processing.

If you did not request Fail Safe processing, transactions that update both CA-IDMS and TIBCO Object Service Broker data can result in discrepancies if the Gateway or the Data Object Broker abnormally terminates during transaction end processing. Refer to Implementing Fail Safe Processing on page 50 for more information.

When a ROLLBACK request is sent to the Gateway, a DML ROLLBACK is sent to CA-IDMS and the TDS Intent Lists are discarded.

Error Handling

The TIBCO Object Service Broker runtime environment signals system exceptions to permit an application to recover from an error. A three-level hierarchy of exceptions exists. All errors encountered when accessing CA-IDMS data through the Gateway are trapped under one of the following exceptions:

ERROR	An error is detected and no lower-level exception handler is coded in the application.
ACCESSFAIL	A table access error is detected.

GETFAIL	No occurrence satisfies the selection criteria. The CA-IDMS status codes 0326 or 0307 raise this exception.
DELETEFAIL	The primary key specified for a DELETE statement does not exist. The CA-IDMS status codes 0326 or 0307 raise this exception. **Note** This exception is raised only when a DELETE statement includes a WHERE primary_key = value.
INSERTFAIL	The primary key provided for an INSERT statement already exists. The CA-IDMS status code 1205 raises this exception.

INTEGRITYFAIL	An attempt to violate data integrity is detected.

See Also *TIBCO Object Service Broker Programming in Rules* about exception handling.

@SERVERERROR Tool

You must pass @SERVERERROR the contents of RETURN_MESSAGE, which has the following format:

```
pppIDnnnx serverid serveruserid source: Message
```

The following list describes the variables necessary to pass the RETURN_MESSAGE contents to @SERVERERROR:

ppp	The user-specified 3 character product ID.
nnn	The CA-IDMS external message number.
x	The message severity (E for error, W for warning, and I for information).
serverid	The server ID of the Gateway.
serveruserid	The Gateway user ID (IDPREFIX + ###) of the Gateway.
source	The code portion of the Gateway that trapped the error and returned the message (for example, CSECT, rule, or function).
Message	The actual error message text.

If a specific message from a specific instance of Service Gateway for IDMS/DB has some information that is required to process the error, the table-driven approach to the execution of @SERVERERROR causes a rule (specified for that error by the developer using @SERVERERROR) to execute. The error message is interpreted in the @SERVERERROR processing and put into a temporary table until required.

To customize error handling, you must update data in the @IDMSTATUSCODES control table. The definition of these tables is owned by TIBCO Object Service Broker and must not be modified. The data you update in them is owned by you.

Table Processing

When the SERVERERROR exception is raised and the @SERVERERROR rule is called by your application, the following steps take place:

1. @SERVERERROR reads the @SERVERMSGCNTL table and looks up the specific message identifier handlers.
2. The appropriate message handler looks up the external error codes in the correct control tables.
3. If any codes are found, they call the associated user-written handler.
4. The user-written handler can use other functions and data stored in specific tables to handle specific external error/status codes.

Considerations

@SERVERERROR can be called at any time, although it is useful only for parsing TIBCO Object Service Broker IDMS/DB messages generated due to external CA-IDMS errors. The original message can always be retrieved using @SE_MSG when @SERVERERROR has been called. The information parsed by @SERVERERROR has transaction scope.

You can add your own instances in the @SERVERMSGCNTL table, provided that the OWNER specified begins with letters A to Z and the key values in their instance are message identifiers in the form ID*nnnx* mentioned on page 103.

See Also *TIBCO Object Service Broker Shareable Tools* for more information on the @SERVERERROR and RETURN_MESSAGE tools.

Appendix A Documenting IDM Tables

This appendix details how to document TIBCO Object Service Broker IDM tables.

Topics

Using the Documentation Screen

Each table definition in TIBCO Object Service Broker has a Documentation screen associated with it. You use this screen to create or modify documentation for the table. To display the Documentation screen for an IDM table, press PF2 from the Table Definer. The Documentation screen associated with the OFFICE_EMP table is shown below.

```
DESCRIPTION OF TABLE            OFFICE_EMP                  UNIT: USR40

MODIFIED ON 20 JAN 2000 BY ACC            CREATED ON 02 JAN 2000 BY USR40

KEYWORDS: EMPLOYEE
SUMMARY : IDMS/DB TABLE CONTAINING EMPLOYEE DATA

                               DESCRIPTION
_ - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
_ This table contains all information on current employees.









PFKEYS: 3=END 5=VIEW DOCUMENT 13=PRINT 12=EXIT
```

Field Values

The Table Definer updates some of the fields on this screen. You must maintain the **KEYWORDS**, **SUMMARY**, and **DESCRIPTION** fields. Complete these fields as follows:

KEYWORDS	Type individual words that briefly describe the table. These words are used by the Keyword Search facility in TIBCO Object Service Broker. This field is one line long and can contain multiple entries, separated by commas or blanks.

SUMMARY	Type a one line summary of the **DESCRIPTION** field.
DESCRIPTION	Type information about the table (for example, what its role is, what it does, and how it works) using TIBCO Object Service Broker **SCRIPT** commands. There is no limit to the amount of information you can type in this field.

PF Keys

The following function keys are supported from the Documentation screen:

PF1	Displays corresponding help for the current field or screen.
PF3	Saves changes and returns you to the Table Definer.
PF5	Toggles between browse and edit modes.
PF12	Cancels changes and returns you to the Table Definer.
PF13	Prints the version of the documentation that you are viewing.

See Also *TIBCO Object Service Broker Shareable Tools* for more information on the SCRIPT tool.

Index

Symbols

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

R

S

T

U

V

W

X

Z