User’s Guide for Transaction Router Services

Mainframe Connect
DirectConnect™ for z/OS Option

12.6

[ Microsoft Windows and UNIX ]
Contents

About This Book ........................................................................................................................... ix

CHAPTER 1 Introducing Transaction Router Service ................................................................. 1
  DirectConnect for z/OS overview ......................................................................................... 1
  DirectConnect architecture ............................................................................................... 1
  DirectConnect components ............................................................................................... 2
  Related products ................................................................................................................ 6
  DirectConnect and TRS environment .............................................................................. 8
  How TRS differs from a DB2 access service ......................................................... 9

CHAPTER 2 Creating a TRS .................................................................................................... 11
  Configuring a TRS library .............................................................................................. 11
  Sample TRS configuration file ...................................................................................... 12
  TRS configuration file format ....................................................................................... 13
  Modifying property values .............................................................................................. 14
  Creating additional TRS configurations ....................................................................... 15
  Creating additional TRS libraries .................................................................................. 15
  Creating additional TRS services .................................................................................. 17
  Service library configuration properties ....................................................................... 18
    AccountFile ................................................................................................................... 21
    Accounting ................................................................................................................... 22
    ConnInfoFile ............................................................................................................... 22
    ConQTimeout (LU 6.2 only) ......................................................................................... 23
    DeactCon (LU 6.2 only) ............................................................................................... 23
    Description .................................................................................................................... 24
    DirectPrevent ............................................................................................................... 24
    LogInfoFile .................................................................................................................... 24
    LogTRS .......................................................................................................................... 25
    MaxConnections .......................................................................................................... 25
    PEMDest ....................................................................................................................... 26
    PEMDestType ............................................................................................................... 26
    ProcessExitEnabled ..................................................................................................... 27
    ProcessExitFile ............................................................................................................. 27
## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ProtocolTraceFile</td>
<td>27</td>
</tr>
<tr>
<td>RegionInfoFile</td>
<td>28</td>
</tr>
<tr>
<td>RPCInfoFile</td>
<td>29</td>
</tr>
<tr>
<td>Security</td>
<td>29</td>
</tr>
<tr>
<td>Send5701</td>
<td>30</td>
</tr>
<tr>
<td>TDSTraceFile</td>
<td>30</td>
</tr>
<tr>
<td>TraceProcessUserExits</td>
<td>31</td>
</tr>
<tr>
<td>TraceProtocol</td>
<td>31</td>
</tr>
<tr>
<td>TraceTRS</td>
<td>31</td>
</tr>
<tr>
<td>TruncateLV</td>
<td>32</td>
</tr>
<tr>
<td>UpgradePassword</td>
<td>32</td>
</tr>
<tr>
<td>UpperCase</td>
<td>33</td>
</tr>
<tr>
<td>UseDBRPC</td>
<td>33</td>
</tr>
<tr>
<td>Service configuration properties</td>
<td>34</td>
</tr>
<tr>
<td>ClientIdleTimeout</td>
<td>34</td>
</tr>
<tr>
<td>Description</td>
<td>35</td>
</tr>
<tr>
<td>EnableAtStartup</td>
<td>35</td>
</tr>
<tr>
<td>XNLChar</td>
<td>36</td>
</tr>
<tr>
<td>XNLngChar</td>
<td>36</td>
</tr>
</tbody>
</table>

### CHAPTER 3 Configuring a TRS ................................................................. 37

- Using TRS administration procedures ........................................ 37
- Command conventions .............................................................. 37
- Viewing command results ....................................................... 38
- Quick reference to TRS administration procedures .................... 39
- Help procedure ........................................................................ 39
- Procedure tables ..................................................................... 39
- Configuration quick-start ..................................................... 43
- Configuring service communications ...................................... 45
  - Configuring connections for LU 6.2 ..................................... 45
  - Configuring regions with TCP/IP .......................................... 48
  - Defining regions to TRS ..................................................... 48
  - Dropping a region ................................................................ 49
- Configuring RPCs .................................................................... 49
  - Defining RPCs to TRS ........................................................ 50
  - Adding an RPC ....................................................................... 50
  - Dropping an RPC ................................................................... 53
- Configuring a default SQL language handler for TRS .................... 53
  - Defining a default SQL language handler ............................... 53
- Using Catalog Stored Procedures (CSPs) .................................... 56
  - CSP scripts ......................................................................... 58
  - Installing CSPs .................................................................... 58
  - Testing CSPs ........................................................................ 59
  - Dropping CSPs ...................................................................... 59
CHAPTER 4  Accessing Catalog Information with CSPs ................................. 61
  Using CSPs ........................................................................................................ 61
    Why use CSPs? .......................................................................................... 62
    Coding instructions .................................................................................. 62
    Wildcard-character search patterns .......................................................... 64
    Escape character .......................................................................................... 65
  Supported CSPs ............................................................................................. 65
  sp_capabilities .............................................................................................. 66
  sp_column_privileges .................................................................................... 68
  sp_columns ....................................................................................................... 70
  sp_databases .................................................................................................... 73
  sp_datatype_info .......................................................................................... 74
  sp_fkeys ........................................................................................................... 76
  sp_pkeys .......................................................................................................... 78
  sp_server_info ............................................................................................... 80
  sp_special_columns ....................................................................................... 81
  sp_sproc_columns .......................................................................................... 83
  sp_statistics ..................................................................................................... 85
  sp_stored_procedures .................................................................................... 87
  sp_table_privileges ....................................................................................... 88
  sp_tables ......................................................................................................... 90
  sp_thread_props ............................................................................................ 92

CHAPTER 5  Configuring a TRS Library for Security ...................................... 93
  Security overview ........................................................................................... 93
    Security features ........................................................................................ 94
    Security considerations ............................................................................... 94
    Security responsibilities ............................................................................ 95
  Security quick-start ........................................................................................ 96
  TRS Administrator’s security tasks .............................................................. 98
    Overriding security .................................................................................... 99
    User IDs ....................................................................................................... 99
    System Administrator’s account ............................................................... 99
    Defining logins to TRS ............................................................................... 100
  User-level security ........................................................................................ 100
    Displaying current logins .......................................................................... 101
    Adding a login ............................................................................................. 101
    Changing user passwords and logins ........................................................ 102
    Changing passwords .................................................................................. 103
    Changing logins ........................................................................................ 103
    Deleting a user definition .......................................................................... 103
  Conversation-level security .......................................................................... 104
    When to forward login information .......................................................... 104
    What login information to forward .......................................................... 105
Contents

Connection-level security (LU 6.2 only) ........................................ 105
Connection groups ........................................................................ 105
Transaction-level security ............................................................ 108
Assigning transaction groups .................................................... 108
Defining a default SQL language handler ................................ 108
Defining group logins ................................................................ 109
Specifying login ID levels .......................................................... 109
Transaction group procedures ............................................... 109

CHAPTER 6 Using Password Expiration Management (PEM) with TRS...... 117
What is PEM? .............................................................................. 117
PEM server capabilities ............................................................. 118
Starting a host transaction.......................................................... 118
Changing the host password.................................................... 118
Implementing PEM functionality for LU 6.2 TRS.................... 119
CICS SIT table property ............................................................ 119
Obtaining information about passwords................................. 120
User password information..................................................... 120
Group password........................................................................ 120
Changing passwords............................................................... 121
Changing an individual password......................................... 121
Changing a group’s password................................................. 122
Setting up new users............................................................... 123

CHAPTER 7 Controlling a TRS .......................................................... 125
Controlling connections (LU 6.2 only) ...................................... 125
Activating connections............................................................. 125
Deactivating a connection......................................................... 127
Controlling regions (TCP/IP Only) ........................................ 128
Activating regions ..................................................................... 128
Deactivating a region............................................................... 128
Disconnecting a client............................................................... 129
Controlling RPCs................................................................. 129
Activating an RPC ................................................................. 129
Deactivating an RPC .............................................................. 130
Controlling tracing .................................................................... 130
Starting tracing ......................................................................... 131
Stopping tracing ......................................................................... 132
Controlling accounting............................................................. 132
Activating and deactivating accounting................................. 133
Reading the accounting log..................................................... 133
Stopping TRS........................................................................ 133
### CHAPTER 8 Monitoring a TRS

- Monitoring the status of TRS: 135
- Monitoring clients: 136
- Monitoring connections (LU 6.2 only): 137
- Monitoring regions (TCP/IP only): 138
- Monitoring RPCs: 139
- Displaying TRS configuration properties: 140
- Requesting trace information: 141
- Summary of clients in each listed state: 142

### APPENDIX A Sending Requests to TRS

- Description of request types: 143
  - Size of requests to AMD2: 144
  - Sending SQL statements to DB2 UDB: 144
  - Accessing DB2 UDB data: 144
  - Sending RPCs to TRS: 144
- Unsupported calls: 146
  - DB-Library calls: 146
  - Client-Library calls: 147

### APPENDIX B Testing a TRS Installation with Sample Programs

- When to test your installation: 149
- Where to find the sample programs: 149
- How to test your TRS installation: 150
  - Starting TRS: 150
  - Defining the connection for Windows (LU 6.2 only): 150
  - Defining the test region (TCP/IP only): 151
  - Defining the test RPC: 152
  - Running the sample: 154
  - Checking for error messages: 154
  - Looking at additional sample programs: 155
  - Looking at catalog RPC scripts: 156

### APPENDIX C Localization

- What is localization?: 157
- How servers handle conversions: 158
- Environment variables for localization: 159
- Localization files: 160
  - Where localization files come from: 160
  - Location of localization files: 161
  - *.loc files: 162
  - Character set files: 162
About This Book

This guide describes how to configure, control, monitor, and use Transaction Router Service (TRS). TRS allows Sybase® clients to access data stored on a mainframe computer. TRS supports mainframe connections for LU 6.2 or TCP/IP networks.

Audience

This book is written for:

- Application Programmers, who develop organization-specific programs using the major features of DirectConnect.
- System Administrators, who install and test DirectConnect. When DirectConnect is running, System Administrators provide ongoing administration support, disaster recovery, and troubleshooting support.
- System Programmers, who install and test DirectConnect. System Programmers also provide product administration, troubleshooting, and disaster recovery.

How to use this book

This guide describes a set of tasks, with each chapter representing a task. The following table shows how this book is organized.

<table>
<thead>
<tr>
<th>See</th>
<th>When you are ready to...</th>
</tr>
</thead>
</table>
| Chapter 1, “Introducing Transaction Router Service” | • Understand Enterprise Connect products.  
• Understand how DirectConnect and TRS work together. |
| Chapter 2, “Creating a TRS” | • Set up a TRS configuration file.  
• Define the properties in that file.  
• Use the file to establish a TRS. |
| Chapter 3, “Configuring a TRS” | • Configure TRS.  
• Perform administration procedures.  
• Use the TRS elements that require administration.  
• Review quick reference tables for the procedures. |
| Chapter 4, “Accessing Catalog Information with CSPs” | Use CSPs to access the DB2 UDB catalog |
### Related documents

To install DirectConnect products, use the ECDA *Installation Guide* for your database system.

To configure and administer DirectConnect for z/OS access services, use the Mainframe Connect DirectConnect for z/OS Option *User’s Guide for DB2 Access Services*.

To install and administer mainframe products, use the following documents:

- MainframeConnect Client Option *Installation and Administration Guide* for CICS
- Mainframe Connect Client Option *Installation and Administration Guide* for IMS and MVS

---

<table>
<thead>
<tr>
<th>See</th>
<th>When you are ready to.....</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chapter 5, “Configuring a TRS Library for Security”</td>
<td>Control client access to TRS, to specific host connections, and to mainframe transactions.</td>
</tr>
<tr>
<td>Chapter 6, “Using Password Expiration Management (PEM) with TRS”</td>
<td>Implement and use the Advanced Program-to-Program Communications (APPC) Password Expiration Management (PEM) function with TRS.</td>
</tr>
<tr>
<td>Chapter 7, “Controlling a TRS”</td>
<td>Use the controlling administration tasks that TRS may need while it is running.</td>
</tr>
</tbody>
</table>
| Chapter 8, “Monitoring a TRS” | • Find information about TRS users, connections, regions, and Remote Procedure Calls (RPCs).  
• Find information about the trace status of TRS.  
• Determine the options specified when TRS started. |
| Appendix A, “Sending Requests to TRS” | For clients using the Enterprise Connect product set, use TRS to access mainframe data. |
| Appendix B, “Testing a TRS Installation with Sample Programs” | Test a TRS installation by running mainframe access products sample programs. |
| Appendix C, “Localization” | Set up an application to run in a particular national language environment. |
| Appendix D, “TRS Process User Exits” | Implement processing user exits. |
| Appendix E, “Compatibility with MDI Database Gateway and Net-Gateway” | Require information regarding the compatibility between DirectConnect and both the MDI Database Gateway™ and Net-Gateway. |
About This Book

- Mainframe Connect Server Option *Installation and Administration Guide* for IMS and MVS
- MainframeConnect™ Server Option *Installation and Administration Guide* for CICS
- Open Client™ and Open Server™ *Common Libraries Reference Manual*
- Open Client *Client-Library/C Programmer’s Guide*
- Open Client *Client-Library/C Reference Manual*
- Open Server *Server-Library/C Reference Manual*

**Other sources of information**

Use the Sybase Getting Started CD, the Sybase Technical Library CD, and the Technical Library Product Manuals Web site to learn more about your product:

- The Getting Started CD contains release bulletins and installation guides in PDF format, and may also contain other documents or updated information not included on the Technical Library CD. It is included with your software. To read or print documents on the Getting Started CD, you need Adobe Acrobat Reader (downloadable at no charge from the Adobe Web site, using a link provided on the CD).

- The Technical Library CD contains product manuals and is included with your software. The DynaText reader (included on the Technical Library CD) allows you to access technical information about your product in an easy-to-use format.

Refer to the *Technical Library Installation Guide* in your documentation package for instructions on installing and starting the Technical Library.

- The Technical Library Product Manuals Web site is an HTML version of the Technical Library CD that you can access using a standard Web browser. In addition to product manuals, you will find links to EBFs/Maintenance, Technical Documents, Case Management, Solved Cases, newsgroups, and the Sybase Developer Network.


**Sybase certifications on the Web**

Technical documentation at the Sybase Web site is updated frequently.

- **Finding the latest information on product certifications**
  2. Select Products from the navigation bar on the left.
3 Select a product name from the product list and click Go.

4 Select the Certification Report filter, specify a time frame, and click Go.

5 Click a Certification Report title to display the report.

❖ **Creating a personalized view of the Sybase Web site (including support pages)**

Set up a MySybase profile. MySybase is a free service that allows you to create a personalized view of Sybase Web pages.

1 Point your Web browser to Technical Documents at http://www.sybase.com/support/techdocs/.

2 Click MySybase and create a MySybase profile.

Sybase EBFs and software maintenance

❖ **Finding the latest information on EBFs and software maintenance**


2 Select EBFs/Maintenance. If prompted, enter your MySybase user name and password.

3 Select a product.

4 Specify a time frame and click Go. A list of EBF/Maintenance releases is displayed.

   Padlock icons indicate that you do not have download authorization for certain EBF/Maintenance releases because you are not registered as a Technical Support Contact. If you have not registered, but have valid information provided by your Sybase representative or through your support contract, click Edit Roles to add the “Technical Support Contact” role to your MySybase profile.

5 Click the Info icon to display the EBF/Maintenance report, or click the product description to download the software.

Style conventions

This book uses the following style conventions:

<table>
<thead>
<tr>
<th>This type of information</th>
<th>Looks like this</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gateway Library function names</td>
<td>TDINIT, TDCANCEL</td>
</tr>
<tr>
<td>Client–Library™ function names</td>
<td>CTBINIT, CTBCANCEL</td>
</tr>
</tbody>
</table>
### Syntax conventions

Syntax statements that display options for a command look like this:

```sql
sp_columns table_name [, table_owner] 
[ , table_qualifier [ , column_name]]
```

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Convention</th>
</tr>
</thead>
<tbody>
<tr>
<td>()</td>
<td>Parentheses are part of the command.</td>
</tr>
<tr>
<td>{ }</td>
<td>Braces indicate that you must choose at least one of the enclosed options. Do not type the braces when you type the option.</td>
</tr>
<tr>
<td>[ ]</td>
<td>Brackets indicate that you can choose one or more of the enclosed options, or none. Do not type the brackets when you type the options.</td>
</tr>
<tr>
<td></td>
<td>The vertical bar indicates that you can select only one of the options shown. Do not type the bar in your command.</td>
</tr>
<tr>
<td>,</td>
<td>The comma indicates that you can choose one or more of the options shown. Separate each choice by using a comma as part of the command.</td>
</tr>
</tbody>
</table>

The following table explains the syntax conventions used in this guide.

<table>
<thead>
<tr>
<th>This type of information</th>
<th>Looks like this</th>
</tr>
</thead>
<tbody>
<tr>
<td>Other executables (DB- Library™ routines, SQL commands) in text</td>
<td>The dbrpcparam routine, a select statement</td>
</tr>
<tr>
<td>Directory names, path names, and file names</td>
<td>/usr/bin directory, interfaces file</td>
</tr>
<tr>
<td>Variables</td>
<td>n bytes</td>
</tr>
<tr>
<td>SQL Server™ datatypes</td>
<td>datetime, float</td>
</tr>
<tr>
<td>Sample code</td>
<td>01 BUFFER PIC S9(9) COMP SYNC</td>
</tr>
<tr>
<td>User input</td>
<td>01 BUFFER PIC X(n)</td>
</tr>
<tr>
<td>Client-Library and Gateway Library function argument names</td>
<td>BUFFER, RETCODE</td>
</tr>
<tr>
<td>Names of objects stored on the mainframe</td>
<td>SYCTSA5</td>
</tr>
<tr>
<td>Symbolic values used with function arguments, properties, and structure fields</td>
<td>CS-UNUSED, FMT-NAME, CS-SV-FATAL</td>
</tr>
<tr>
<td>Client-Library property names</td>
<td>CS-PASSWORD, CS-USERNAME</td>
</tr>
<tr>
<td>Client-Library and Gateway Library datatypes</td>
<td>CS-CHAR, TDSCHAR</td>
</tr>
</tbody>
</table>

All other names and terms are in regular typeface.
If you need help

Each Sybase installation that has purchased a support contract has one or more designated people who are authorized to contact Sybase Technical Support. If you cannot resolve a problem using the manuals or online help, please have the designated person contact Sybase Technical Support or the Sybase subsidiary in your area.
CHAPTER 1

Introducing Transaction Router Service

This chapter provides an overview of DirectConnect for z/OS, of which Transaction Router Service (TRS) is a component. DirectConnect for z/OS combined with other Sybase products provides access and integration of mainframe data. For more information, see the Overview Guide for Mainframe Connect.

This chapter contains the following topics:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>DirectConnect for z/OS overview</td>
<td>1</td>
</tr>
<tr>
<td>How TRS differs from a DB2 access service</td>
<td>9</td>
</tr>
</tbody>
</table>

DirectConnect for z/OS overview

This section describes DirectConnect for z/OS and other Sybase products that DirectConnect and TRS interact with. This section covers the following topics:

- DirectConnect architecture
- DirectConnect components
- Related products
- DirectConnect and TRS environment

DirectConnect architecture

DirectConnect for z/OS is Open Server-based software that supports DB-Library, CT-Library, and Open Database Connectivity (ODBC) application programming interfaces (APIs).
DirectConnect serves as a fundamental building block for highly-scalable database middleware applications. DirectConnect products are local area network (LAN)-based middleware gateways and servers that provide access to non-Sybase data and applications.

In addition, DirectConnect can be used with other Sybase products, such as Adaptive Server®, ASE/CIS, and Replication Server.

DirectConnect for z/OS consists of:

- A server, which provides the framework in which service libraries can operate
- One or more service libraries (DB2 UDB and TRS), which provide the framework in which access services can operate
- One or more access services for each service library (DB2 UDB and TRS), which are the logical points of connection for DirectConnect clients.

The following subsections describe each of these components.

**DirectConnect components**

This section describes the following DirectConnect components:

- DirectConnect server
- DirectConnect service libraries
- DirectConnect services

**Note** Mainframe Client Connect (MCC) is no longer being provided or supported. Sybase recommends that you migrate from a three-tier (gateway-enabled) environment to a two-tier (gateway-less) environment using TCP.
DirectConnect server

The DirectConnect server provides management and support functions for DirectConnect service libraries, such as:

- Routing client connections to the appropriate access service based on user ID, requesting application, and access service name.
- Providing a single log file and a trace file for access services. TRS has its own Tabular Data Stream™ (TDS) trace file, LU 6.2 protocol trace file and TCP/IP protocol trace file.
- Logging server, access service, and client messages.
- Tracing server, access service, and client events.
- Providing configuration management of all installed services.

For detailed information about configuring and starting the server, see the DirectConnect Server Administration Guide.
DirectConnect service libraries

Residing on the DirectConnect server, a service library is a set of configuration properties that describes how its access services will function. The following service libraries reside on the DirectConnect server:

- Transaction Router Service Library
- Access Service Library
- Administrative Service Library

DirectConnect services

An access service is the client connection point for a DirectConnect server. It is the pairing of a service library with a set of specific values for the configuration properties.

DB2 UDB access services

A DB2 access service works with MainframeConnect for DB2 UDB to allow clients to access DB2 data.

Each access service is a specific set of configuration properties that:

- Transforms SQL
- Convert datatypes
- Supports remote stored procedures (RPCs)
- Transfers data between DB2 UDB and other servers accessible through Open Client
- Supports Catalog Stored Procedures (CSPs) and system stored procedures
- Supports RSPs and host-resident requests

For more information about the DirectConnect for z/OS DB2 UDB Access Service Library, see the Mainframe Connect DirectConnect for z/OS Option User’s Guide for DB2 Access Services.

Transaction Router Service (TRS)

Each TRS library contains a TRS that provides access to DB2 data and supports Open ServerConnect™ mainframe applications, defined to TRS as (RPCs).
The TRS access service routes requests from remote LAN-based clients to Open ServerConnect transactions. Optionally, it can also route requests to MainframeConnect™ for DB2 UDB and return results to the client.

Security can also be configured on a transaction or user basis.

There are two TRS service libraries:
- *TRSLU62* service library, which uses the LU 6.2 communications protocol to talk to Mainframe Connect or any Open ServerConnect application running in CICS
- *TRSTCP* service library, which uses the Transmission Control Protocol/Internet Protocol (TCP/IP) communications protocol to talk to MainframeConnect for DB2 UDB or any Open ServerConnect application running in CICS

Having multiple instances of a TRS library on a server results in different physical copies of the shared library files that constitute the TRS component.

An explanation of the TRS components of DirectConnect can be found in the DirectConnect *Transaction Router Service User’s Guide*.

**Administrative Service Library**

The Administrative Service Library provides specific administrative services for all DirectConnect libraries, including writing to logs and allowing remote configuration of DirectConnect services (for example, through DirectConnect Manager).

**DirectConnect Manager**

DirectConnect Manager is a graphical user interface (GUI) systems management tool for administering DirectConnect. It allows you to:
- Manage DirectConnect servers on multiple platforms
- Change configuration properties of DirectConnect servers, service libraries, and services
- Create and copy services
- Create new servers using DCDirector
- Start and stop existing servers using DCDirector
- Start, stop, and delete services
- Test the availability of a data source by creating a connection to it
DirectConnect for z/OS overview

- Retrieve a DirectConnect server log file or a subset of the log, and view log file messages with a text editor
- Update DirectConnect server connection information
- View the status of a service and data source on the desktop

Related products

This section describes products that DirectConnect interacts with to provide mainframe access for LAN client requests.

MainframeConnect for DB2 UDB

MainframeConnect for DB2 UDB is a CICS transaction that works with DirectConnect for z/OS to provide access to mainframe data. It performs the following functions:

- Supports full read-write, dynamic SQL access to data
- Allows applications to use cursors for flexible and efficient result set processing
- Permits the use of long-running transactions against mainframe databases
- Allows applications to use dynamic events to map SQL to a static plan

DirectConnect invokes MainframeConnect to access mainframe data on behalf of its Open Client-based clients, such as:

- ASE/CIS
- ASE through RPCs
- Enterprise Application Server
- JDBC or ODBC applications
- Replication Server

Note MainframeConnect for DB2 UDB is available only for z/OS CICS environments.
Open ServerConnect

Open ServerConnect is a programming environment that lets you create mainframe transactions that are accessible to Sybase client applications. To provide this access, Open ServerConnect uses the following basic interfaces:

- Traditional Open Server programming environment (for new customers and Sybase-heritage customers using new applications)
- RSP programming environment (only for MDI-heritage customers using their legacy applications)

These transactions provide access to virtually any MVS data source and are used for a variety of functions, including:

- Accessing existing mainframe applications
- Initiating mainframe batch jobs
- Providing source data for data transfer operations
- Providing data mapped to a table within ASE/CIS thus allowing results to be accessed or joined with data from other targets

LAN-side client applications access Open ServerConnect transactions directly through DirectConnect or indirectly through ASE/CIS or a Sybase Adaptive Server RPC.

Open ClientConnect

Open ClientConnect is a programming environment that lets you create mainframe applications that access:

- LAN data residing on a Sybase Adaptive Server or other supported data sources
- Other CICS regions

It allows you to treat the mainframe as if it were just another node on a LAN.

Open ClientConnect uses the following APIs:

- Traditional Open Client programming environment (for new customers and Sybase-heritage customers using new applications)
- Client Services Application (CSA) programming environment (only for MDI-heritage customers using their legacy applications)
DirectConnect and TRS environment

The following figure shows the relationship of the DB2 UDB access service library and TRS library with various components of the client workstation, LAN, and mainframe environments.

*Figure 1-2: DirectConnect for z/OS environment*

As shown, the request from a client application goes over the LAN to the DirectConnect server. From there, either TRS or a DB2 access service routes the request to the appropriate CICS region. Then, the request accesses data on the UDB database.
For more information on how to create multiple TRS libraries, see “Creating additional TRS configurations” on page 15.

How TRS differs from a DB2 access service

Like the DB2 UDB service library component of DirectConnect for z/OS, TRS allows users access to DB2 UDB data. They both perform protocol translation, route client requests and server results, and allow remote mainframe password management.

A DB2 UDB access service allows the client application to access data stored in a DB2 UDB database running on z/OS through MainframeConnect for DB2 UDB, a CICS transaction; however, the DB2 UDB access service cannot invoke other CICS transactions. In direct contrast, TRS allows the client through MFC, to invoke CICS, IMS, and MVS transactions that are based on Open ServerConnect APIs.

In addition, TRS provides:

- Additional security control on a user or transaction basis
- Access to IMS and MVS data
- Access to multiple Open ServerConnect-based CICS transactions in multiple CICS regions, including any MainframeConnect running in the defined regions.

A DB2 UDB access service provides:

- Access to remote stored procedure (RSP) programs (TRS does not)
- DB2 and SQL datatype transformation
- Access to bidirectional transfer functionality
- Advanced datatype conversion

The following figure shows how a single client connection through TRS can access many CICS transactions.
How TRS differs from a DB2 access service

Figure 1-3: TRS accessing many CICS transactions

Basically, you use TRS when:

- Your applications invoke Open ServerConnect-based mainframe transactions.
- You use client applications written for the TRS predecessor, Net-Gateway.
Creating a TRS

This chapter describes how to create and customize a DirectConnect Transaction Router Service (TRS) library and name a service associated with that service library.

This chapter contains the following topics:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuring a TRS library</td>
<td>11</td>
</tr>
<tr>
<td>Creating additional TRS configurations</td>
<td>15</td>
</tr>
<tr>
<td>Service library configuration properties</td>
<td>18</td>
</tr>
<tr>
<td>Service library configuration properties</td>
<td>34</td>
</tr>
</tbody>
</table>

Note You can use DirectConnect Manager to create and edit a TRS configuration file.

Configuring a TRS library

To create and modify a TRS, you must edit the TRS configuration file. It is a simple text file named srvlibname.cfg, where srvlibname is the base name of the TRS executable file. For example, if you have the sample LU 6.2-based TRS, your default configuration file is called TRSLU62.CFG. You can use a text editor to change any service library property by editing and saving this file. This file is located in one of the following directories:

- For Microsoft Windows:
  %SYBASE%\%SYBASE_ECON%\srvrname\cfg
- For UNIX:
  $SYBASE/$SYBASE_ECON/srvrname/config
Configuring a TRS library

Running DirectConnect with the -N option as part of the initial configuration will create a sample TRS configuration file that you can modify for your site. For information about installing the sample service library, see the Mainframe Connect DirectConnect for z/OS Option Installation Guide.

When you edit the TRS configuration file:

- Enter values for configuration properties that apply to that TRS only. The system ignores properties that are not applicable to your installation. For example, the ConnInfoFile property applies only to LU 6.2 installations and not to TCP/IP installations.

- Ignore properties for which the default value is sufficient. See “Service library configuration properties” on page 18 for the default values of each configuration property.

The TRS library uses some configuration information from the DirectConnect server. For instructions about configuring DirectConnect server properties, see the ECDA and Mainframe Connect Server Administration Guide for DirectConnect.

Sample TRS configuration file

Following is an example of a TRS configuration file:

```
[Service Library]
{Transaction Router Service Property}
PEMDest=CICSQA
RPCInfoFile=d:\newpath\SYBASE\DC-12_0\srvrname\cfg\trslu62.rpc
LogInfoFile=d:\newpath\SYBASE\DC-12_0\srvrname\log\trslu62.grp
TDSTraceFile=d:\newpath\SYBASE\DC-12_0\srvrname\log\trslu62.tds
AccountFile=d:\newpath\SYBASE\DC-12_0\srvrname\log\trslu62.act
MaxConnections=100
TraceTRS=short
Security=no
DirectPrevent=yes
Accounting=yes
UseDBRPC=no
TruncateLV=no
UpperCase=no
ConnInfoFile=c:\newpath\SYBASE\DC-12_0\srvrname\cfg\trslu62.cid
ConQTimeout=120
DeactCon=no
[ServiceA]
EnableAtStartup=yes
```
TRS configuration file format

The following principles apply to TRS configuration properties:

- Service library properties apply to the service library as a whole. TRS configuration properties are service library properties, except the service-level properties Description, EnableAtStartup and ClientIdleTimeout. How a TRS service operates is affected by the values of its parent service library.

- Configuration properties are not case sensitive. In this guide, property names appear in mixed case for easier reading.

A TRS configuration file consists of the following lines. (For a sample file, see “Sample TRS configuration file” on page 12.)

- The name of the TRS service library is shown in brackets on the first line of the file. This character string must appear at the top of the file.

- The subsection name, Transaction Router Service Property, is shown in braces on the next line. This character string must appear under the service library line. There are no other subsections.

- Each configuration property and its value are shown on individual lines. Configuration properties can be listed in any order within their subsection. If a configuration property line is deleted or omitted from the file for any reason, the default value for that property is applied automatically. See “Service library configuration properties” on page 18 for the default values of each configuration property.

- The TRS service name is shown in brackets and must conform to the following rules:
  - Service names must be unique within the first 11 characters in length.
  - The initial character must be an alphabetic character (a–z, A–Z).
  - Subsequent characters can be alphabetic or numeric characters or the underscore (_) character.
  - To add a service named “ServiceA,” the following line must exist in the TRS configuration file:
    
    [ServiceA]
Configuring a TRS library

For a client to successfully connect to a service, the service name must correspond to a query type entry in the client interfaces file. For an explanation of query type entries and how to add them to the interfaces file, see the Mainframe Connect DirectConnect for z/OS Option Installation Guide. When a client connects to DirectConnect, it specifies a service name, as shown in the following isql example:

```
isql -Usa -P -SServiceA
```

go

where ServiceA is the service name (or server name to the client).

You can include comments in the TRS configuration file. Each comment must be on a separate line and begin with a semicolon or “#” symbol.

Modifying property values

Most configuration properties have default values. Some properties require that you supply values for your site. You can change the properties by using a text editor or by using DirectConnect Manager.

Using DirectConnect Manager

You can configure the TRS library properties and the TRS service properties by using the DirectConnect Manager. To configure the TRS library and service properties, refer to the DirectConnect Manager online Help topic, Managing Configuration | Modifying server configuration properties.

**Note** When you use DirectConnect Manager to change Accounting, LogTRS, and TraceTRS properties, changes take effect immediately. Changes to all other TRS library configuration properties take effect when you restart the server.

Using a text editor

To change the TRS service library and services configuration property values:

1. Open the TRS configuration file and change the service library properties as applicable.

2. Open the TRS service file and change the TRS service property values as applicable.

3. Save the file.
Creating additional TRS configurations

This section describes how to create additional TRS libraries and services.

Creating additional TRS libraries

Because many of the server library specific properties affect the major functions of TRS, you may want to create multiple instances of a TRS Library to obtain functionally different configurations. For example, if you want one TRS LU62 Library service that enforces security and one that does not, create two instances of the TRS LU62 Library. This is necessary because the security configuration property operates at the server library level and affects all services in that library. To copy a TRS library, use the trscopy utility, described in the following subsection.

Description of the trscopy utility

The trscopy utility creates a copy of a DirectConnect TRS Library by using an existing TRS as a template to find all of the related source files.

**Note** This utility makes a copy of a TRS service library executable available for all DirectConnect servers defined under the installation area, but sets up a sample service only under the same DirectConnect server as provided on the command line.

The utility copies files with a base file name of the source TRS in the directory tree of the source DirectConnect server to files with a base file name of the target TRS in the directory tree of the target DirectConnect server.

For example, if you are using Windows, to create a new instance of the TRS LU62 Library (named new_trslu62), run the trscopy program with the appropriate arguments. Doing this produces two complete sets of the TRS LU62 files:

- The original trslu62 executable and all its files

4. Stop the server, and then restart it to implement the changes.
Creating additional TRS configurations

- An executable named `new_trslu62` and a copy of all of its files, with a base file name of `new_trslu62`, placed in the destination DirectConnect server subdirectory tree

The following table shows the TRS executable file names based upon the platform:

<table>
<thead>
<tr>
<th>Platform</th>
<th>File name and extension</th>
</tr>
</thead>
<tbody>
<tr>
<td>Windows</td>
<td><code>trslu62.dll</code> and <code>trstcp.dll</code></td>
</tr>
<tr>
<td>HP</td>
<td><code>trslu62.sl</code> and <code>trstcp.sl</code></td>
</tr>
<tr>
<td>AIX</td>
<td><code>trslu62.so</code> and <code>trstcp.so</code></td>
</tr>
<tr>
<td>Solaris</td>
<td><code>trslu62.so</code> and <code>trstcp.so</code></td>
</tr>
</tbody>
</table>

After you run `trscopy`, you must edit the new TRS Library’s configuration file and change the service name. Then, add this new service name to the SYBASE interfaces file on the client. For instructions regarding the interfaces file refer to the Mainframe Connect DirectConnect for z/OS Option Installation Guide.

**Note** The UNIX version of `trscopy` has an option that automatically generates a new TRS Service name in the target TRS Library configuration file and adds that service to the SYBASE interfaces file on the server.

Using the trscopy command

Following is the `trscopy` command for Microsoft Windows:

```
trscopy source_DirectConnect source_Service_Library
destination_DirectConnect destination_Service_Library
```

Following is the `trscopy` command for UNIX platforms:

```
trscopy.sh source_DirectConnect source_Service_Library
destination_DirectConnect destination_Service_Library
```

The UNIX version of `trscopy` has an option that automatically generates a new TRS Service name in the target TRS Library configuration file and adds that service to the SYBASE interfaces file on the server.

Table 2-2 describes the parameters and options in the `trscopy` command.
CHAPTER 2  Creating a TRS

Table 2-2: Description of trscopy parameters and options

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source_DirectConnect</td>
<td>(Required) Name of the server that contains the source TRS files. It is located in the directory under the following name:</td>
</tr>
<tr>
<td></td>
<td>• %SYBASE%%SYBASE_ECON% for Windows</td>
</tr>
<tr>
<td></td>
<td>• $SYBASE/$SYBASE_ECON for UNIX</td>
</tr>
<tr>
<td></td>
<td>%SYBASE% or $SYBASE is the Sybase environment variable. It points to the Sybase product directory structure that contains DirectConnect for both the source and target TRS service libraries.</td>
</tr>
<tr>
<td>source_TRS_Library</td>
<td>(Required) The name of the source TRS library associated with this server. It must exist under the source DirectConnect server name.</td>
</tr>
<tr>
<td>destination_DirectConnect</td>
<td>(Required) Name of the target DirectConnect server, located in the directory under %SYBASE%%SYBASE_ECON%.</td>
</tr>
<tr>
<td>destination_TRS_Library</td>
<td>(Required) Name of the target TRS library. Follow these guidelines:</td>
</tr>
<tr>
<td></td>
<td>• The executable associated with the server library name must not exist under the target DirectConnect server.</td>
</tr>
<tr>
<td></td>
<td>• The file name must not exist anywhere under the following locales directory structure, based on platform:</td>
</tr>
<tr>
<td></td>
<td>- %SYBASE%\locales, for Windows</td>
</tr>
<tr>
<td></td>
<td>- $SYBASE/locales, for UNIX</td>
</tr>
<tr>
<td>-a</td>
<td>(Optional and for UNIX only) Option that generates new service names for the new service library and adds these new services to the interfaces file.</td>
</tr>
<tr>
<td>-I</td>
<td>(Optional and for UNIX only) A “softlink” option. When possible, use softlinks instead of copying a file.</td>
</tr>
<tr>
<td>-v</td>
<td>(Optional and for UNIX only) Verbose option, which displays a description of each operation while it runs.</td>
</tr>
</tbody>
</table>

Creating additional TRS services

You can create additional services through DirectConnect Manager or through use of your text editor.

To create a new service using DirectConnect Manager, see Creating a new service in the DirectConnect Manager Online Help topic, Managing Access Services.

Using a text editor

Create and change existing services by following these steps:

1. Open the TRS configuration file from one of the following directories:
For Windows platforms:
\%SYBASE\%SYBASE_ECON\%srvrname\trs_service_lib.cfg

For UNIX platforms:
$SYBASE/$SYBASE_ECON/srvrname/trs_service_lib.cfg

2 Add the service name in brackets below the initial service names section.
3 By default, services are not enabled for client connection at start-up. If you want this service to be enabled at start-up, add the EnableAtStartup configuration property, set to yes, below the new service name.
4 Save the file.
5 Stop the server, and then restart it to implement the changes.
6 To be sure that client applications can connect to a new TRS service from a client machine, you must enter the service name in the SYBASE interfaces file on the client machine. If you choose to use the Service Name Redirection utility, make an assigned service name entry in the Service Name Redirection file.

• Start DirectConnect Manager.
• Double-click server name.
• Right-click the Services folder. Select create service.
• Select TRS-LU62 or TRS-TCP/IP from the type of services window and enter the name of your new TRS service.
• Click Finish. The new service name will be displayed.

For instructions about editing the interfaces file, see the Mainframe Connect DirectConnect for z/OS Option Installation Guide.

For information about service name redirection, see the ECDA and Mainframe Connect Server Administration Guide for DirectConnect.

Service library configuration properties

The following table lists all TRS service library configuration properties and identifies the location, in this chapter, for a more detailed description.
TRS is compatible with all mainframe access products that were accessed through Net-Gateway. The following table shows the name of the Net-Gateway start-up parameter that is equivalent to each TRS configuration property.

**Table 2-3: TRS configuration properties**

<table>
<thead>
<tr>
<th>TRS configuration property</th>
<th>Equivalent Net-Gateway start-up parameter</th>
<th>TRS configuration property description</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>AccountFile</td>
<td>(none)</td>
<td>Specifies the directory, path, and file name to which accounting records are written.</td>
<td>“AccountFile” on page 21</td>
</tr>
<tr>
<td>Accounting</td>
<td>-K flag</td>
<td>Turns accounting on and off.</td>
<td>“Accounting” on page 22</td>
</tr>
<tr>
<td>ConnInfoFile (LU 6.2 only)</td>
<td>-L flag</td>
<td>Specifies the directory, path, and file name that contains LU 6.2 connection information for this TRS.</td>
<td>“ConnInfoFile” on page 22</td>
</tr>
<tr>
<td>ConQTimeout (LU 6.2 only)</td>
<td>-Q flag</td>
<td>Specifies the LU 6.2 connection queue timeout period (wait period) in seconds.</td>
<td>“ConQTimeout (LU 6.2 only)” on page 23</td>
</tr>
<tr>
<td>DeactCon (LU 6.2 only)</td>
<td>-d flag</td>
<td>Indicates whether an LU 6.2 connection should be deactivated or left active if a line failure or other error occurs.</td>
<td>“DeactCon (LU 6.2 only)” on page 23</td>
</tr>
<tr>
<td>Description</td>
<td>(none)</td>
<td>An optional customer-supplied description of the service library.</td>
<td>“Description” on page 24</td>
</tr>
<tr>
<td>DirectPrevent</td>
<td>-D flag</td>
<td>Instructs TRS to accept requests from Adaptive Server clients only.</td>
<td>“DirectPrevent” on page 24</td>
</tr>
<tr>
<td>LogInfoFile</td>
<td>-G flag</td>
<td>Specifies the directory, path, and file name that contains client login and security group information.</td>
<td>“LogInfoFile” on page 24</td>
</tr>
<tr>
<td>LogTRS</td>
<td>(none)</td>
<td>Enables or disables logging to the server log file.</td>
<td>“LogTRS” on page 25</td>
</tr>
<tr>
<td>MaxConnections</td>
<td>-M flag</td>
<td>Specifies the maximum number of clients that can be logged into this TRS library concurrently.</td>
<td>“MaxConnections” on page 25</td>
</tr>
<tr>
<td>PEMDest</td>
<td>-P flag</td>
<td>Specifies the destination region handling the IBM Password Expiration Management (PEM) transaction program. Applies to LU 6.2 only.</td>
<td>“PEMDest” on page 26</td>
</tr>
</tbody>
</table>
## Service library configuration properties

<table>
<thead>
<tr>
<th>TRS configuration property</th>
<th>Equivalent Net-Gateway start-up parameter</th>
<th>TRS configuration property description</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>PEMDestType</td>
<td>-m flag</td>
<td>Specifies the type of destination region (<em>PEMDest</em>) managing the PEM server transaction. Applies to LU 6.2 only.</td>
<td>“PEMDestType” on page 26</td>
</tr>
<tr>
<td>ProcessExitEnabled</td>
<td>(none)</td>
<td>Enables the use of process user exits.</td>
<td>“ProcessExitEnabled” on page 27</td>
</tr>
<tr>
<td>ProcessExitFile</td>
<td>(none)</td>
<td>Identifies the path and name of the shared library that you have created.</td>
<td>“ProcessExitFile” on page 27</td>
</tr>
<tr>
<td>RegionInfoFile (TCP/IP only)</td>
<td>(none)</td>
<td>Specifies the directory path and file name for the file that contains TCP/IP connection information for this TRS.</td>
<td>“RegionInfoFile” on page 28</td>
</tr>
<tr>
<td>RPCInfoFile</td>
<td>-R flag</td>
<td>Specifies the directory, path, and file name of the file containing the remote procedure call (RPC) information for this TRS.</td>
<td>“RPCInfoFile” on page 29</td>
</tr>
<tr>
<td>Security</td>
<td>-O flag</td>
<td>Tells TRS whether to validate logins against its own login information in addition to the validation done by the mainframe.</td>
<td>“Security” on page 29</td>
</tr>
<tr>
<td>Send5701</td>
<td>-u flag</td>
<td>Indicates whether the message 5701 should be sent back to the client for use database statements.</td>
<td>“Send5701” on page 30</td>
</tr>
<tr>
<td>TDSTraceFile</td>
<td>(none)</td>
<td>Specifies the directory, path, and name of the file to which TDS information is written.</td>
<td>“TDSTraceFile” on page 30</td>
</tr>
<tr>
<td>TraceProcessUserExits</td>
<td>(none)</td>
<td>Traces entry/exit points of function call to each process user exit you have created.</td>
<td>“TraceProcessUserExits” on page 31</td>
</tr>
<tr>
<td>TraceTRS</td>
<td>-T and -t flags</td>
<td>Specifies the level of TDS tracing that TRS is to record.</td>
<td>“TraceTRS” on page 31</td>
</tr>
<tr>
<td>TruncateLV</td>
<td>-V flag</td>
<td>Truncates any mainframe long varchar fields to 255 bytes before sending them to the client.</td>
<td>“TruncateLV” on page 32</td>
</tr>
</tbody>
</table>
The remainder of this chapter describes each TRS configuration property. Configuration properties are presented in alphabetical order.

### AccountFile

Specifies the directory, path, and name of the file to where TRS writes accounting records. See “Controlling accounting” for more information about the type of accounting information that TRS captures.

**Syntax**

```
AccountFile=newpath
```

where `newpath` is the directory, path, and name of the file to which TRS writes accounting records.

**Default**

- For an LU 6.2-based TRS on Windows, `newpath` is:
  ```
  %SYBASE%\%SYBASE_ECON%\srvrname\log\srvlname.act
  ```
- For a TCP/IP-based TRS on Windows, `newpath` is:
  ```
  %SYBASE%\%SYBASE_ECON%\srvrname\log\srvlname.act
  ```
Service library configuration properties

- For a TRS on UNIX platforms, newpath is:
  
  \$SYBASE/\$SYBASE\_ECON/srvrname/log/ngact.srvlibname

Values

%SYBASE\% (or $SYBASE) is the name of the default Sybase environment variable, which can be reset before server start-up.

srvrname is the name of the DirectConnect server defined during server installation.

srvlibname is the name of the TRS library.

Accounting

Turns accounting on or off. TRS writes accounting records to the file specified in the AccountFile configuration property.

TRS users with administrative privileges can also turn accounting on and off using the sgw\_startact and sgw\_stopact procedures while TRS is running.

**Note** Using DirectConnect Manager, you can turn accounting on and off dynamically while TRS is running or at server start-up.

Syntax

Accounting=[ no | yes ]

Default

no

Values

no turns accounting off.

yes turns accounting on.

ConnInfoFile

Specifies the directory, path, and name of the file that contains LU 6.2 connection information for this TRS. This file is created the first time you define an LU 6.2 connection. A default is defined for TCP/IP on non-UNIX platforms. (To define connections, use the sgw\_addcon procedure.)

Syntax

ConnInfoFile=newpath

where newpath is the directory path and name of the TRS LU 6.2 connection information file.

Default

- For an LU 6.2-based TRS on Windows, newpath is:
For a TCP/IP-based TRS on Windows, \textit{newpath} is:

\%SYBASE\%\%SYBASE\_ECON\%\%srvrname\%cfg\%srvlibname\%cid

For a TRS on UNIX platforms, \textit{newpath} is:

\$SYBASE/$SYBASE\_ECON/srvrname/cfg/ngcid.srvlibname

Values

\%SYBASE\% (or $SYBASE) is the name of the default Sybase environment variable, which can be reset before server start-up.

\textit{srvrname} is the name of the DirectConnect server defined during server installation.

\textit{srvlibname} is the name of the TRS library.

**ConQTimeout (LU 6.2 only)**

Specifies the number of seconds client requests are allowed to wait in a queue for an available LU 6.2 connection to the destination system. A client request times-out (expires) if a connection does not become available in the specified length of time.

**Syntax**

ConQTimeout=\textit{timeout}

where \textit{timeout} is the maximum number of seconds that each client request remains in a queue to wait for an available LU 6.2 connection to the destination system before the client request expires.

**Range**

0 to 50000

**Default**

60

**Comments**

Specify 0 (zero) if you do not want client requests to queue up. When a timeout occurs, TRS returns a message to the client.

**DeactCon (LU 6.2 only)**

Indicates whether TRS deactivates or leaves active an LU 6.2 connection if a line failure or other error occurs on that connection.

If you specify deactivation, TRS marks the failing connections as inactive when an LU 6.2 error occurs. You can restart connections while TRS is running by using the \texttt{sgw\_actcon} procedure.
### Service library configuration properties

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeactCon=[ no</td>
<td>yes ]</td>
</tr>
</tbody>
</table>

#### Description

An optional customer-supplied description of the service library.

**Syntax**

`Description=description`

*description* where is the description of the service library up to 255 alphanumeric characters.

**Default**

A blank string, for example:

```
Description=
```

**Comments**

Specifying the default sets the value to a blank string.

#### DirectPrevent

Instructs TRS to reject all direct requests from a client, forcing clients to route all requests through Adaptive Server. Setting this property to `no` allows clients to send RPCs and language requests directly to TRS.

**Syntax**

`DirectPrevent=[ no | yes ]`

**Default**

`no`

**Values**

- yes rejects all direct requests from the client and forces the client to route all requests through Adaptive Server.
- no allows clients to send RPCs and language requests directly to TRS.

#### LogInfoFile

Specifies the directory, path, and name of the file that contains client login and security group information. TRS creates this file the first time you define a client login, connectivity group, or transaction group. (To define client logins, use the `sgw_addlog` procedure.)
CHAPTER 2    Creating a TRS

User's Guide for Transaction Router Services

Syntax

LogInfoFile=newpath

where newpath is the directory path and file name of the file containing the client login and security group information for this TRS library.

Default

For an LU 6.2-based TRS on Windows, newpath is:

%SYBASE%%SYBASE_ECON%\svrname\cfg\srvlibname.grp

For a TCP/IP-based TRS on Windows, newpath is:

%SYBASE%%SYBASE_ECON%\svrname\cfg\srvlibname.nggrp

For UNIX platforms, newpath is:

$SYBASE/$SYBASE_ECON/svrname/cfg/nggrp.srvlibname

Values

%SYBASE% (or $SYBASE) is the name of the default Sybase environment variable, which can be reset before server start-up.

svrname is the name of the DirectConnect server defined during server installation.

srvlibname is the name of the TRS library.

LogTRS

Enables or disables logging to the server log file, located at:

For UNIX platforms:

$SYBASE/$SYBASE_ECON/svrname/log/srvlibname.log

For Windows platforms:

%SYBASE%%SYBASE_ECON%\svrname\log\srvlibname.log

Syntax

LogTRS=[ no | yes ]

Default

no

Values

yes turns logging on.

no turns logging off.

MaxConnections

Specifies the maximum number of clients that can be logged into this TRS library concurrently.
## Service library configuration properties

### MaxConnections

**Syntax**  
MaxConnections=integer

where integer is a number of clients.

**Range**  
1–n, where n is the maximum number of clients allowed by the server. (For more information, see “MaxConnections” in the DirectConnect Server Administration Guide.)

**Default**  
25

**Comments**  
TRS does not verify the validity of this number.

### PEMDest

 Specifies the destination system for the IBM Password Expiration Management (PEM). PEM is a password management program that IBM provides.

Sybase provides support for PEM as a feature of TRS for LU 6.2. This feature is not available for TRS connections to the mainframe using TCP/IP.

For more information about implementing PEM, see Chapter 6, “Using Password Expiration Management (PEM) with TRS.”

**Syntax**  
PEMDest=destsys

where destsys is system-dependent value identifying the LU 6.2 connection from which the IBM PEM sign-on transaction can be accessed. Use the value supplied for the region parameter when this LU 6.2 connection was defined with the sgw_addcon procedure.

**Default**  
No default. No value is required unless clients are using TRS PEM support.

**Comments**  
Leaving the default blank string in place disables PEM RPCs.

### PEMDestType

 Specifies the type of destination region managing the PEM server transaction as defined by the PEMDest configuration property.

**Syntax**  
PEMDestType=[ CICS | MVS ]

**Default**  
CICS

**Values**  
CICS indicates that the PEMDest value connects to a CICS region.

MVS indicates that the PEMDest value connects to native MVS.
**ProcessExitEnabled**

Enables the use of process *user exits*. Only the exits that you have defined and added to your *user exit* library will be invoked.

**Syntax**

```
ProcessExitEnabled= [ yes | no ]
```

**Default**

*no*

**Values**

- *yes* enables the use of process user exits.
- *no* does not allow process user exits to be invoked.

**ProcessExitFile**

Provides the full path and name of the *user exit* shared library that you have created.

**Syntax**

```
ProcessExitFile= [ path / filename | null ]
```

**Default**

*null*

**Values**

- *path / filename* identifies the full path and file name of the *user exit* shared library that you created.
- *null* indicates that no process *user exit* shared library has been created.

**ProtocolTraceFile**

Specifies the directory path and file name in which DirectConnect uses the back-end transport protocol traces and errors for conversations between the TRS library and the mainframe.

**Syntax**

```
T DST r aceFile= newpath
```

where *newpath* is the directory path and file name to which traces are written for TRS.

**Default**

- On Windows, back-end TCP/IP tracing goes into the *newpath*:
  ```
  %SYBASE%/SYBASE_ECON/srvrname/log/trstcp.ngtcp
  ```
- On UNIX, back-end TCP/IP tracing goes into the *newpath*:
  ```
  $SYBASE/$SYBASE_ECON/srvrname/log/ngtcp.trstcp
  ```
- On Windows, back-end LU 6.2 tracing goes into the *newpath*:
Service library configuration properties

%SYBASE%/SYBASE_ECON/srvrname/log/trslu62.nglu62

- On UNIX, back-end LU6.2 tracing goes into the newpath:
  
  $SYBASE/$SYBASE_ECON/srvrname/log/nglu62.trslu62

Values

%SYBASE% (or $SYBASE) is the name of the default Sybase environment variable, which can be reset before server start-up.

srvrname is the name of the DirectConnect server defined during server installation.

RegionInfoFile

Specifies the directory, path, and name of the file containing the TCP/IP connection information that it creates the first time you define a TCP/IP region. A default has been defined for LU 6.2-based TRS on Windows. (To define regions, use the sgw_addregion procedure.)

Syntax

RegionInfoFile=newpath

where newpath is the directory path and file name of the file containing the TRS TCP/IP connection information.

Default

- For an LU 6.2-based TRS on Windows, newpath is:

  %SYBASE%/SYBASE_ECON/srvrname/cfg/srvlibname.reg

- For a TCP/IP-based TRS on Windows, newpath is:

  %SYBASE%/SYBASE_ECON/srvrname/cfg/srvlibname.ngreg

- For UNIX platforms, newpath is:

  $SYBASE/$SYBASE_ECON/srvrname/cfg/ngreg.srvlibname

Values

%SYBASE% (or $SYBASE) is the name of the default Sybase environment variable, which can be reset before server start-up.

srvrname is the name of the DirectConnect server defined during server installation.

srvlibname is the name of the TRS library.
CHAPTER 2  Creating a TRS

RPCInfoFile

Specifies the directory path and file name of the file containing the RPC information that TRS creates the first time you define an RPC. (To define RPCs, use the sgw_addrpc procedure.)

Syntax

RPCInfoFile=newpath

where newpath is the directory path and file name of the file containing the RPC information for this TRS.

Default

- For an LU 6.2-based TRS on non-UNIX platforms, newpath is:
  %SYBASE%\%SYBASE_ECON%\srvrname\cfg\srvlibname.rpc
- For a TCP/IP-based TRS on non-UNIX platforms, newpath is:
  %SYBASE%\%SYBASE_ECON%\srvrname\cfg\srvlibname.ngrpc
- For UNIX platforms, newpath is:
  $SYBASE/$SYBASE_ECON/srvrname/cfg/ngrpc.srvlibname

Values

%SYBASE% (or $SYBASE) is the name of the default Sybase environment variable, which can be reset before server start-up.

srvrname is the name of the DirectConnect server defined during server installation.

srvlibname is the name of the TRS library.

Security

Tells TRS whether to validate logins against its own login information in addition to the validation done by the mainframe. If you set the Security configuration property to no, TRS forwards transparently the user ID and password to the mainframe (based on the RPC definition security parameter values defined in the sgw_addrpc procedure). You do not need to use the sgw_addlogsecurity procedure to add new users to TRS when you set to no.

Syntax

Security=[ no | yes ]

Default

yes

Values

yes turns security on.

no turns security off.
Service library configuration properties

Send5701

Indicates whether message 5701 should be sent to the client for use database statements.

| Syntax          | Send5701=[ no | yes ] |
|-----------------|------------------|
| Default         | no               |
| Values          | yes sends message 5701 to the client for use database statements. |
|                 | no does not send message 5701 to the client for use database statements. |

TDSTraceFile

Specifies the directory path and file name in which the Tabular Data Stream (TDS) records traces and errors for conversations between the TRS library and the mainframe. TDS is the Sybase application-level protocol that defines the form and content of relational database requests and replies.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>TDSTraceFile=newpath</th>
</tr>
</thead>
<tbody>
<tr>
<td>Where</td>
<td>newpath is the directory path and file name to which TDS traces are written for TRS.</td>
</tr>
<tr>
<td>Default</td>
<td>- For an LU 6.2-based TRS on Windows, newpath is:</td>
</tr>
<tr>
<td></td>
<td>%SYBASE%/%SYBASE_ECON%/srvrname/log/srplibname.tds</td>
</tr>
<tr>
<td></td>
<td>- For a TCP/IP-based TRS on Windows newpath is:</td>
</tr>
<tr>
<td></td>
<td>%SYBASE%/%SYBASE_ECON%/srvrname/log/srplibname.ngtds</td>
</tr>
<tr>
<td></td>
<td>- For UNIX platforms, newpath is:</td>
</tr>
<tr>
<td></td>
<td>$SYBASE/$SYBASE_ECON/srvrname/log/ngtds.srplibname</td>
</tr>
<tr>
<td>Values</td>
<td>%SYBASE% (or $SYBASE) is the name of the default Sybase environment variable, which can be reset before server start-up.</td>
</tr>
<tr>
<td></td>
<td>srvrname is the name of the DirectConnect server defined during server installation.</td>
</tr>
<tr>
<td></td>
<td>srplibname is the name of the TRS library.</td>
</tr>
</tbody>
</table>
**TraceProcessUserExits**

Traces the entry/exit points of the function call to each of the process *user exits* that you have created. Normal setting is *no*, however, a setting of *yes* will assist you in determining execution through your processing user exits.

**Syntax**

```
TraceProcessUserExits=[ yes | no ]
```

**Default**

`no`

**Values**

- *yes*: tracing is turned on for the process user exits you have created.
- *no*: tracing is not turned on for the process user exits that you have created.

**TraceProtocol**

Provides protocol level tracing for DirectConnect using LU 6.2 or TCP/IP, depending on which service library is being traced.

**Syntax**

```
TraceProtocol=[ none | short | long ]
```

**Default**

`none`

**Values**

- *short* or *long*: protocol level tracing is turned on for LU 6.2 or TCP/IP.
- *none*: protocol level tracing is not turned on for LU 6.2 or TCP/IP.

**TraceTRS**

Turns TDS tracing on or off. Using the *long* or *short* option turns on tracing. You can also turn tracing on and off when TRS is running using the `sgw_starttrace` and `sgw_stoptrace` procedures, with the TDS parameters, or by using DirectConnect Manager. Formatted TDS traces are written to a file defined by the `TDSTraceFile` configuration property. See “Starting tracing” on page 131.

**Note** Using DirectConnect Manager, you can turn tracing on and off dynamically while TRS is running or at server start-up.

**Syntax**

```
TraceTRS=[ none | short | long ]
```

**Default**

`none`

**Values**

- *long*: traces both TDS header information and TDS data packets.
short traces TDS header information only.
none turns tracing off.

**TruncateLV**

Truncates any mainframe long varchar fields to 255 bytes before sending them to the client. Setting this property to no causes long varchar data to be sent as text and image datatypes for 4.x TDS clients, or as the appropriate long varchar datatype for 5.0 TDS clients. (Sybase System 10 and later versions use TDS 5.0.)

**Syntax**

TruncateLV=[ no | yes ]

**Default**

no

**Values**

no turns long varchar truncation off, causing long varchar data to be sent as text and image datatypes for 4.x TDS clients, or as the appropriate long varchar datatype for 5.0 TDS clients.

yes turns long varchar truncation on, truncating any mainframe long varchar fields to 255 bytes before sending them to the client.

**UpgradePassword**

Indicates whether pre-Net-Gateway version 3.0.1 passwords (8 bytes maximum) should be upgraded to the new format (30 bytes maximum). If this property is set to yes, all existing old passwords are lost and are initialized to null.

**Syntax**

UpgradePassword=[ no | yes ]

**Default**

yes

**Values**

no prevents the upgrading of pre-TRS passwords.

yes upgrades all pre-TRS passwords to the new format, and deletes all existing passwords and initializes them to null.

**Comments**

- If you are upgrading from Net-Gateway version 2.0, set this configuration property to yes.

Before setting UpgradePassword to yes and triggering the upgrade, copy your log information files to a save directory, as follows:

```
cd %SYBASE%
```
mkdir save_nggrp
    copy log_info_files save_nggrp

For the name of the log information file for your system configuration, see the LogInfoFile configuration property.

UpperCase

Automatically changes lowercase user IDs and passwords to uppercase for clients before forwarding these values to the mainframe.

Syntax UpperCase=[ no | yes ]

Default yes

Values no prevents the automatic changing of lowercase user IDs and passwords to uppercase before forwarding these values to the mainframe. TRS forwards the user ID and password as is.

yes enables the automatic changing of lowercase user IDs and passwords to uppercase.

UseDBRPC

Allows a client to send RPC requests larger than 64KB to the mainframe.

When an RPC is executed, a client sends a TDS_RPC TDS token stream to the server. The server reads the stream, processes the request and returns any results back to the client. The TDS_RPC stream in TDS version 5.0 had a 2-byte integer indicating the total length of the TDS_RPC stream, which limits each TDS_RPC stream to 64KB in length.

DirectConnect provides a new token, named TDS_DBRPC. This token removes the RPC length limit.

Note If you want to send RPC requests larger than 64K to the mainframe, and you are running Open ServerConnect software that predates Open ServerConnect 3.1, check your latest release bulletins to verify that the UseDBRPC property is compatible with your version. If your Open ServerConnect version is incompatible with the UseDBRPC property, your RPC requests will fail when this feature is on.

Syntax UseDBRPC=[ no | yes ]
Service configuration properties

Default yes
Values no turns DBRPC off, preventing a client from sending RPC requests larger than 64K to the mainframe.

yes turns DBRPC on, allowing a client to send RPC requests larger than 64K to the mainframe.
Comments None

Service configuration properties

This section describes the TRS service configuration properties. The following table identifies and describes the TRS service configuration properties:

<table>
<thead>
<tr>
<th>TRS service configuration property</th>
<th>Equivalent Net-Gateway start-up parameter</th>
<th>TRS service configuration property description</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>ClientIdleTimeout</td>
<td>(none)</td>
<td>Specifies the number of minutes a connected TRS client can remain idle before being disconnected.</td>
<td>“ClientIdleTimeout” on page 34</td>
</tr>
<tr>
<td>Description</td>
<td>(none)</td>
<td>An optional customer supplied definition of the TRS service.</td>
<td>“Description” on page 35</td>
</tr>
<tr>
<td>EnableAtStartup</td>
<td>(none)</td>
<td>Specifies whether the TRS service starts and accepts client connections when the DirectConnect server starts.</td>
<td>“EnableAtStartup” on page 35</td>
</tr>
<tr>
<td>XNLChar</td>
<td>(none)</td>
<td>Specifies the maximum size of both char and binary results.</td>
<td>“XNLChar” on page 36</td>
</tr>
<tr>
<td>XNLVarChar</td>
<td>(none)</td>
<td>Specifies the maximum size of both varchar and varbinary results.</td>
<td>“XNLVarChar” on page 36</td>
</tr>
</tbody>
</table>

ClientIdleTimeout

Specifies how many minutes a client connection can remain inactive before an access service terminates the connection.
Syntax:  

\[
\text{ClientIdleTimeout=integer}
\]

Range: 0–1024
Default: 0

Values: \(integer\) is how many minutes a client connection can remain inactive before an access service terminates the connection.

0 indicates that an access service never terminates an idle connection.

Comments: \begin{itemize}
  \item A connection is idle when:
    \begin{itemize}
    \item A client is connected, but did not issue a command.
    \item A command processed, but the client did not issue another command.
    \item A large result set returned from SQL request processing, and the result screen paused for the specified timeout period.
    \end{itemize}
  \item The TRS access service checks client activity once per minute. Therefore, a client can remain inactive for up to one minute beyond the \text{ClientIdleTimeout} value before the TRS access service terminates the connection.
\end{itemize}

Description

An optional customer-supplied description of the TRS service.

Syntax:  

\[
\text{Description=where}
\]

description is the description of the TRS service up to 255 alphanumeric characters.

Default: No default.

Comments: Specifying no value sets the value to a blank string.

EnableAtStartup

Specifies whether this TRS service starts and accepts client connections when the DirectConnect server starts.

Syntax:  

\[
\text{EnableAtStartup=[ no | yes ]}
\]

Default: yes

Values: no means that the TRS service does not start when the server starts.
**Service configuration properties**

**Mainframe Connect DirectConnect for z/OS Option**

- **yes** means that the TRS service starts when the server starts.

  **Comments**
  
  If you are not using DirectConnect Manager to manage your access services, set this property to yes.

---

**XNLChar**

Specifies the maximum size of both char and binary results. If the maximum size is exceeded, the datatype is promoted to text and image, respectively.

- **Syntax**
  
  `XNLChar=integer`

- **Default**
  
  256

- **Values**
  
  `integer` is a valid number between 256 - 2147483647 (two gigabytes).

- **Comments**
  
  Sybase recommends that this value match the maximum size of the char and binary datatypes of the back end database. It is common for this limit to be the same for the char and binary datatypes.

---

**XNLVarChar**

Specifies the maximum size of both varchar and varbinary results. If the maximum size is exceeded, the datatype is promoted to text and image, respectively.

- **Syntax**
  
  `XNLVarChar=integer`

- **Default**
  
  256

- **Values**
  
  `integer` is a valid number between 256 - 2147483647.

- **Comments**
  
  Sybase recommends that the value match the maximum size of the varchar and varbinary datatypes of the back end database. It is common for this limit to be the same for the varchar and varbinary datatypes.
CHAPTER 3 Configuring a TRS

This chapter describes DirectConnect Transaction Router Service (TRS) configuration tasks and the command line administration procedures you use to perform those tasks.

This chapter contains the following topics:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Using TRS administration procedures</td>
<td>37</td>
</tr>
<tr>
<td>Quick reference to TRS administration procedures</td>
<td>39</td>
</tr>
<tr>
<td>Configuration quick-start</td>
<td>43</td>
</tr>
<tr>
<td>Configuring service communications</td>
<td>45</td>
</tr>
<tr>
<td>Configuring RPCs</td>
<td>49</td>
</tr>
<tr>
<td>Configuring a default SQL language handler for TRS</td>
<td>53</td>
</tr>
<tr>
<td>Using Catalog Stored Procedures (CSPs)</td>
<td>56</td>
</tr>
</tbody>
</table>

Using TRS administration procedures

TRS administration procedures begin with `sgw_`, which stands for server gateway.

The TRS administration procedures are `isql` `execute` commands. Start `isql` or your preferred dynamic SQL utility as usual, and then enter the commands at the prompt.

Command conventions

Observe the following conventions when you use TRS administration procedures:

- Run each `exec` command individually; do not batch them.
Using TRS administration procedures

- Enter `go` after each command, or execute the command according to the conventions of your SQL utility. (Generally, `go` is not shown in the syntax illustrations in this guide; it is shown in the examples.)

- Enclose command parameters that contain numerical values in quotation marks.

- Enter all command parameters in the order shown. Separate the parameters with commas. (Spaces are optional.)

  If you omit any parameters, include the commas as placeholders or use the keyword `NULL` (not case sensitive). For example, if you want the TRS user WAYNE to have a password of BLEUCHEZ, you enter the following:

  ```
  exec sgw_chpwd WAYNE, , BLEUCHEZ
  go
  ```

  or

  ```
  exec sgw_chpwd WAYNE, NULL, BLEUCHEZ
  go
  ```

  The first parameter after `sgw_chpwd` is the login, in this case, WAYNE. The second comma (or `null`) holds the place of the TRS password, which you are not changing. The parameter BLEUCHEZ represents the new password that is passed to the mainframe.

- When entering TRS administration procedures, you need to enter only as many characters as required to make each parameter distinct from any other (you must enter at least three characters). For example, to query the status of the clients on TRS, the command is as follows:

  ```
  execute sgw_status clients
  ```

  Or, you can enter the following:

  ```
  exec sgw_status cli
  ```

Viewing command results

The results of the administration procedures display on the screen where you entered the command. If the results take up more lines than one screen can display, the information may scroll by quickly (depending on your SQL utility). In this case, you can use your operating system utilities to direct the results of the procedure to a file.
Quick reference to TRS administration procedures

This section provides a quick reference to the administration procedures available for TRS. Sorted by type of procedure, the tables list the object to be operated on, the procedure to use, and a location you can access for detailed information.

In these procedure tables, the parameter values you should replace with the appropriate values for your site are shown in italics. Parameters shown in uppercase should be entered in UPPERCASE.

Help procedure

To display an online listing of the command syntax for TRS administration procedures, use the exec command, as shown in the following isql example:

```
exec sgw_help
go
```

The results show a list of the commands, with a short description and syntax for each, identifying all optional entries.

Procedure tables

You may find it useful to photocopy the following tables and post them near your workstation for easy reference. The tables provided on the following pages are listed here:

- For add/drop procedures for client, connection, login, region, RPC, and transaction group, refer to Table 3-1 on page 40.
- For change procedures for login (password) and transaction group, refer to Table 3-2 on page 41.
- For display and status procedures for accounting, client, connection, login, region, RPC, parameter, trace and transaction group, refer to Table 3-3 on page 41.
- For start and stop procedures for accounting, connection, TRS, region, RPC, and trace, refer to Table 3-4 on page 42.
- For Password Expiration Management (PEM) procedures for login (host password) and group login (host password) refer to Table 3-5.
Quick reference to TRS administration procedures

Add/drop procedures

For add/drop procedures for client, connection, login, region, RPC, see the following table.

Table 3-1: Add/drop procedures

<table>
<thead>
<tr>
<th>Element</th>
<th>Procedure</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client</td>
<td>sgw_disclient &quot;client_number&quot;</td>
<td>“Disconnecting a client” on page 129</td>
</tr>
<tr>
<td>Connection</td>
<td>sgw_addcon con_name, region, mode, “max_sessions”</td>
<td>“Adding a connection configuration” on page 46</td>
</tr>
<tr>
<td></td>
<td>sgw_dropcon con_name</td>
<td>“Dropping a connection configuration” on page 47</td>
</tr>
<tr>
<td></td>
<td>sgw_dropcon con_name, region</td>
<td>“Dropping individual regions from a connection configuration” on page 48</td>
</tr>
<tr>
<td></td>
<td>sgw_addcongrp group_name</td>
<td>“Adding a connection group” on page 106</td>
</tr>
<tr>
<td></td>
<td>sgw_dropcongrp group_name</td>
<td>“Dropping a connection group” on page 108</td>
</tr>
<tr>
<td></td>
<td>sgw_addcontogrp group_name, con_name</td>
<td>“Adding connections to a connection group” on page 107</td>
</tr>
<tr>
<td></td>
<td>sgw_dropconfromgrp group_name, con_name</td>
<td>“Dropping connections from a connection group” on page 107</td>
</tr>
<tr>
<td>Login</td>
<td>sgw_addlog login, pwd, HOST_LOGIN, HOST_PWD, tran_group, con_group, gwctrl</td>
<td>“Adding a login” on page 101</td>
</tr>
<tr>
<td></td>
<td>sgw_droplog login</td>
<td>“Deleting a user definition” on page 103</td>
</tr>
<tr>
<td>Region</td>
<td>sgw_addregion region, HOSTNAME, “port_number”</td>
<td>“Defining regions to TRS” on page 48</td>
</tr>
<tr>
<td></td>
<td>sgw_addregion region, HOSTNAME, “port_number”, regiontype (GICS, IMS, MVS)</td>
<td>“Defining regions to TRS” on page 48</td>
</tr>
<tr>
<td></td>
<td>sgw_dropregion region</td>
<td>“Dropping a region” on page 49</td>
</tr>
</tbody>
</table>

Note: con_group is for LU 6.2 only. For TCP/IP, include comma or null as a placeholder.
CHAPTER 3  Configuring a TRS

<table>
<thead>
<tr>
<th>Element</th>
<th>Procedure</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPC</td>
<td>sgw_addrpc rpc_name, TRAN_ID, region, security (none</td>
<td>userid</td>
</tr>
<tr>
<td></td>
<td>sgw_droppc rpc_name</td>
<td>“Dropping an RPC” on page 53</td>
</tr>
<tr>
<td></td>
<td>sgw_addrpcctogrp tran_group, rpc_name, rpcpwdlevel (none</td>
<td>user</td>
</tr>
<tr>
<td></td>
<td>sgw_droprpcfromgrp tran_group, rpc_name</td>
<td>“Deleting RPC names from a transaction group” on page 113</td>
</tr>
<tr>
<td>Transaction group</td>
<td>sgw_addtrngroup tran_group, GROUP_LOGIN, GROUP_PWD, langrpc, langpwdlevel (none</td>
<td>user</td>
</tr>
<tr>
<td></td>
<td>sgw_droptngroup tran_group</td>
<td>“Deleting a transaction group” on page 115</td>
</tr>
</tbody>
</table>

Change procedures

For change procedures for login (password) and transaction group, see the following table.

*Table 3-2: Change procedures*

<table>
<thead>
<tr>
<th>Element</th>
<th>Procedure</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Login (password)</td>
<td>sgw_chpwd login, pwd, HOST_PWD</td>
<td>“Changing passwords” on page 103</td>
</tr>
<tr>
<td>Transaction group</td>
<td>sgw_modtrngroup tran_group, GROUP_LOGIN, GROUP_PWD, langrpc, langpwdlevel (none</td>
<td>user</td>
</tr>
</tbody>
</table>

Display and status procedures

For display and status procedures for accounting, client, connection, login, region, RPC, parameter, trace and transaction group, see the following table.

*Table 3-3: Display/status procedures*

<table>
<thead>
<tr>
<th>Element</th>
<th>Procedure</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accounting</td>
<td>sgw_dspact</td>
<td>“Reading the accounting log” on page 133</td>
</tr>
<tr>
<td>Client</td>
<td>sgw_status clients</td>
<td>“Monitoring clients” on page 136</td>
</tr>
<tr>
<td></td>
<td>sgw_status summary</td>
<td>“Summary of clients in each listed state” on page 142</td>
</tr>
</tbody>
</table>
Quick reference to TRS administration procedures

<table>
<thead>
<tr>
<th>Element</th>
<th>Procedure</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connection (LU 6.2 only)</td>
<td>sgw_status connections</td>
<td>“Monitoring connections (LU 6.2 only)” on page 137</td>
</tr>
<tr>
<td></td>
<td>sgw_dspcongrp</td>
<td>“Displaying one connection group” on page 106</td>
</tr>
<tr>
<td></td>
<td>sgw_dspcongrp con_group</td>
<td>“Displaying one connection group” on page 106</td>
</tr>
<tr>
<td>Login</td>
<td>sgw_dsplog</td>
<td>“Displaying current logins” on page 101</td>
</tr>
<tr>
<td>Region (TCP/IP only)</td>
<td>sgw_status region</td>
<td>“Monitoring regions (TCP/IP only)” on page 138</td>
</tr>
<tr>
<td>RPC</td>
<td>sgw_status rpc</td>
<td>“Monitoring RPCs” on page 139</td>
</tr>
<tr>
<td>Parameter</td>
<td>sgw_status parameters</td>
<td>“Displaying TRS configuration properties” on page 140</td>
</tr>
<tr>
<td>Trace</td>
<td>sgw_status trace</td>
<td>“Requesting trace information” on page 141</td>
</tr>
<tr>
<td>Transaction group</td>
<td>sgw_dsptmgrp</td>
<td>“Displaying all transaction groups” on page 110</td>
</tr>
<tr>
<td></td>
<td>sgw_dsptmgrp tran_group</td>
<td>“Assigning transaction groups” on page 108</td>
</tr>
<tr>
<td></td>
<td>sgw_dsptmgrp tran_group, rpc</td>
<td>“Assigning transaction groups” on page 108</td>
</tr>
</tbody>
</table>

Start and stop procedures

For start and stop procedures for accounting, connection, TRS, region, RPC, and trace, see the following table.

**Table 3-4: Start/stop procedures**

<table>
<thead>
<tr>
<th>Element</th>
<th>Procedure</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accounting</td>
<td>sgw_startact</td>
<td>“Activating and deactivating accounting” on page 133</td>
</tr>
<tr>
<td></td>
<td>sgw_stopact</td>
<td>“Activating and deactivating accounting” on page 133</td>
</tr>
<tr>
<td>Connection (LU 6.2 only)</td>
<td>sgw_actcon all</td>
<td>“Restarting all connections” on page 126</td>
</tr>
<tr>
<td></td>
<td>sgw_actcon “con_number”</td>
<td>“Activating a single connection” on page 126</td>
</tr>
<tr>
<td></td>
<td>sgw_deactcon “con_number”</td>
<td>“Deactivating a connection” on page 127</td>
</tr>
<tr>
<td></td>
<td>sgw_deactcon “con_number”, force</td>
<td>“Deactivating a connection” on page 127</td>
</tr>
</tbody>
</table>
Password Expiration Manager (PEM) procedures

For Password Expiration Management (PEM) procedures for login (host password) and group login (host password), see the following table.

<table>
<thead>
<tr>
<th>Element</th>
<th>Procedure</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Login (Host password)</td>
<td>sqw_peminfopwd host_login, host_password</td>
<td>“Obtaining information about passwords” on page 120</td>
</tr>
<tr>
<td>Login (Host password)</td>
<td>sqw_pemchpwd new_password, new_password</td>
<td>“Changing an individual password” on page 121</td>
</tr>
<tr>
<td>Group login (Host password)</td>
<td>sqw_peminogruppwd tran_grp</td>
<td>“Group password” on page 120</td>
</tr>
<tr>
<td>Group login (Host password)</td>
<td>sqw_pemchgrppwd tran_grp, new_pwd, new_pwd</td>
<td>“Changing a group’s password” on page 122</td>
</tr>
</tbody>
</table>

Configuration quick-start

Note This section assumes that you are not enforcing security at TRS.
Configuration quick-start

The following are brief, step-by-step instructions for configuring TRS. These steps help you run the sample programs described in Appendix B, “Testing a TRS Installation with Sample Programs” after you first install TRS.

Refer to the complete description of each procedure for details.

1 Set the TRS Security property to no.
   See “Security” on page 29.

2 Start TRS.
   DirectConnect brings up TRS at start-up as long as the TRS exists in one of the following directories:
   - For Windows: %SYBASE%%SYBASE_ECON%<srvrname>svclib/
   - For UNIX: $SYBASE/$SYBASE_ECON/<srvrname>/svclib/

3 Start an isql session, connecting to TRS.

4 Do one of the following:
   - **LU 6.2 only**: Use sgw_addcon to define the connections your TRS uses.
     See “Adding a connection configuration” on page 46.
     ```
     exec sgw_addcon con_name, region, mode, "max_sessions"
     ```
   - **TCP/IP only**: Use sgw_addregion to specify the regions your TRS uses.
     See “Defining regions to TRS” on page 48.
     ```
     exec sgw_addregion region, hostname, "port_number", region_type
     ```

5 Use the sgw_addrpc procedure to add remote procedure calls (RPCs).
   See “Defining RPCs to TRS” on page 50.
   ```
   exec sgw_addrpc rpc_name, tran_id, region, security
   ```
The TRS client is now able to log in to TRS with a valid host user ID and password and execute the added RPCs.

**Note** Be sure to set up the default SQL language transaction as AMD2, SYRT, or SYIH. For more information, see “Defining a default SQL language handler” on page 53.

### Configuring service communications

Use the instructions in this section if you are installing TRS, a new Open ServerConnect transaction, or if you need to configure TRS to use MainframeConnect for DB2 UDB.

There are some differences in the steps required depending on whether you are using LU 6.2 connections or TCP/IP:

- For LU 6.2, configure TRS by defining mainframe *connections* and client RPCs to TRS.
- For TCP/IP, configure TRS by defining mainframe *regions* and client RPCs to TRS.

### Configuring connections for LU 6.2

This section explains how to define new LU 6.2 connection configurations to TRS and how to remove existing connection configurations.

Every connection between TRS and a transaction processing region must be defined to TRS. When you execute an RPC, TRS chooses a connection with a *region* value that matches the definition in the RPC to execute the transaction.

Consider the following:

- If you are not using parallel sessions, do not use the same LU 6.2 pair or more than one session on it for more than one TRS, or for any other LU 6.2 application. Dedicate a separate set of connections for each TRS.
Configuring service communications

- If you use parallel sessions, you can use an LU 6.2 pair for TRS. However, be sure that you configure a sufficient number of sessions for the total number of Open ServerConnect users and Open ClientConnect users. Also, be sure that the workstation is configured as the “contention winner.” (Check with your mainframe system programmer.)

**Note** When possible, Sybase recommends limiting use of an LU 6.2 pair to only one TRS other LU 6.2 application. This configuration simplifies the analysis if there are any LU 6.2 problems.

Adding a connection configuration

Add a connection configuration to TRS for each LU 6.2 pair defined to your SNA support. To define a new connection configuration to TRS, use this procedure:

```
exec sgw_addcon con_name, region, mode,
"max_sessions"
```

where

- `con_name` is the name assigned to this local connection. It is also the name by which the Local LU is known to your local SNA support. Because there is a secondary name that qualifies this connection, this parameter corresponds to different values for different platforms. See the Mainframe Connect DirectConnect for z/OS Option Installation Guide for specific information about connection name parameter values.

  Length: maximum of eight characters.

- `region` specifies the remote LU name of the target mainframe region in this parameter. This is the Virtual Telecommunications Access Method (VTAM) APPLID name to which your Local LU is bound. An entry in this field is required.

  All RPCs that use this connection configuration to access the mainframe must have this same value specified as the region in their RPC definitions. (See also “Adding an RPC” on page 50.)

  Length: maximum of eight characters.

For different platforms, this parameter corresponds to different values. See the Mainframe Connect DirectConnect for z/OS Option Installation Guide for specific information about the mode name parameter value.
• *mode* is a value that must match the name of the mode defined to the mainframe and to the local SNA support for this *LU* pair.

  Length: maximum of eight characters.

• *“max_sessions”* is the maximum number of sessions that this TRS can have simultaneously allocated from the *LU* pair. Enter one of the following:
  • For parallel sessions, enter a value between 2 and 255.
  • For a single session, this value can only be 1.

  Be sure to enclose numeric parameter values in quotation marks.

  Check with your SNA System Administrator to make sure this number is not larger than the maximum number of sessions (for this mode) defined to the SNA subsystem.

  **Note** If you do not provide a value for *“max_sessions,”* TRS creates a default value = 1 for the connection, which will not support parallel sessions.

---

**Example**

This example adds an LU 6.2 connection configuration named SYBLU01, bound to region (remote LU) TESTREG, with mode name M6S1024V, and not using parallel sessions:

```
exec sgw_addcon SYBLU01, TESTREG, M6S1024V, "1"
go
```

This example adds an LU 6.2 connection configuration named SYBLU01, bound to region (remote LU) PRODEMO, with mode name M6P1024V and eight parallel sessions:

```
exec sgw_addcon SYBLU01, PRODEMO, M6S1024V, "8"
go
```

**Dropping a connection configuration**

To delete all LU 6.2 connection configurations of a particular *con_name* from TRS, use this procedure:

```
exec sgw_dropcon con_name
```

where *con_name* with the name of the connection configuration you want to drop. The connection configuration name appears in the connections status display.
Configuring regions with TCP/IP

Example

To drop the connection configuration named SYBLU01, use the following procedure:

```
exec sgw_dropcon SYBLU01
```

Dropping individual regions from a connection configuration

To delete a connection configuration for a particular LU pair from a connection configuration, use this procedure:

```
exec sgw_dropcon con_name, region
```

Provide a value for the `con_name` parameter and the `region` (optional) parameter to drop a specific connection.

**Warning!** Providing only the `con_name` parameter value deletes all connection configurations for that `con_name`.

Example

The following `isql` example deletes both sets of connection configurations added in the example:

```
exec sgw_dropcon SYBLU01
```

The next example deletes only the eight connections defined to PRODEMO:

```
exec sgw_dropcon SYBLU01, PRODEMO
```

Configuring regions with TCP/IP

This section describes how to define and drop regions to TRS using TCP/IP.

Defining regions to TRS

For TRS to recognize the specified `region` parameter of the `sgw_addrpc` procedure, you must define the region using the following `sgw_addregion` procedure:

```
exec sgw_addregion region, hostname, portnumber,
```
region

where:

- region is a TRS administrator-defined alias for the hostname and portnumber pair, described next. For RPCs to use this region, this value must match the value in their region parameter of the sgw_addrpc procedure. (See “Adding an RPC” on page 50.)
  Length: maximum of eight characters.

- hostname is the value you specify for this parameter that identifies the TCP/IP network host name. This name corresponds to the mainframe in your /etc/hosts file or in your NIS map.
  Length: maximum of 31 characters.

- portnumber is the number you specify for this parameter that must match the port number on which the transaction listens. (This is not the same as the port number used to configure the interfaces file.) TRS does not verify the validity of this number with the CICS TCP/IP Listener.
  This value can be any number between 1024 and 9996.

- region (optional) is the type of the mainframe processing environment specified by the region parameter. Valid values are CICS, MVS, and IMS. If you do not specify a value, the region type defaults to CICS.

Dropping a region

When you want to remove a region from those configured to TRS, use the following procedure:

exec sgw_dropregion region

where region is the name of the region you intend to drop.

Configuring RPCs

A remote procedure call (RPC) is an Open ServerConnect mainframe application. TRS can be configured to invoke any Open ServerConnect mainframe application.
This section explains how to define new RPCs to TRS, and how to remove existing RPC definitions.

**Defining RPCs to TRS**

When TRS receives a request from a client, it needs the following information before it can forward the request to the mainframe:

- The name of the associated mainframe transaction
- The name of the region that identifies connectivity to the mainframe location where the transaction runs (defined in the `sgw_addcon` or `sgw_addregion` procedure)

You define this information to TRS when you add an RPC.

**Adding an RPC**

To define a new RPC to TRS for each new Open ServerConnect transaction and map it to a region, use this procedure:

```
exec sgw_addrpc rpc_name, tran_id, region, security
```

where:

- **rpc_name** is the TRS alias for the remote procedure. This is the name the client uses to call this RPC.
  Length: maximum of 30 characters.
- **tran_id** is the name by which the associated transaction is known on the mainframe. This is the mainframe transaction that TRS calls when a client requests the named procedure. The value of this field must be in uppercase. If the first character is numeric, the `tran_id` must be in quotes.
  Length:
  - For CICS: maximum of 4 characters
  - For IMS: maximum of 8 characters
  - For MVS: maximum of 8 characters
- **region**
  For LU 6.2, specifies the region name used to identify the LU 6.2 connection configuration in the `sgw_addcon` procedure.
At least one defined connection configuration must have this value specified as its region. See “Adding a connection configuration” on page 46. An entry in this field is required.

For TCP/IP, specifies the region name used to identify this TCP/IP connection in the sgw_region procedure. See “Defining regions to TRS” on page 48. This field is required.

Length: maximum of eight characters.

- region
  (TCP/IP only) Specifies the region name used to identify this TCP/IP connection in the sgw_region procedure. See “Defining regions to TRS” on page 48. This field is required.
  Length: maximum of eight characters.

- security
  specifies the type of user login information that TRS passes to the mainframe. The security parameter is not case sensitive.
  
  • The security parameter can have any of the following values to specify the information to send:
    
    none. Do not send login information to the mainframe.
    
    userid. Send only the user ID to the mainframe. To determine which userid is used, see the Security level source section.
    
    both. Send both the user ID and the password to the mainframe. To determine which userid is used, see the Security level source section.

  **Note** This setting is not applicable when using the TRS TCP/IP Library.

  • If you are using LU 6.2, TRS passes the information in the conversation-level security fields of the SNA LU 6.2 Function Management Header 5 (FMH-5).
  
  • With TCP/IP, TRS passes these fields to the Listener Transaction when the called transaction starts.
For example, if you use native CICS security, the none value corresponds to the CICS security option NONE, userid corresponds to IDENTIFY, and both corresponds to the security option VERIFY.

**Note** SNA network products vary in that some do not allow only the user ID to be forwarded; in other words, the ALREADY VERIFIED bit may not be set. Check your platform-specific DirectConnect and vendor SNA documentation for restrictions.

### Example

To add an RPC named SYD2, use this command:

```
exec sgw_addrpc SYD2, SYD2, TESTREG, none
```

This maps SYD2 to the mainframe transaction named SYD2, which executes in the mainframe region named TESTREG. A user ID or password is not passed through to the mainframe when the RPC is invoked.

### Security level source

When you invoke an RPC defined with a security parameter value of userid or both, the values passed to the mainframe for the user ID and password can come from one of three different pairs of values:

- If TRS security is *off*, (see the security configuration parameter), TRS passes to the mainframe the user ID and password that is used to login to TRS.

- If TRS security is *on*, (see the security configuration parameter), and the rpcpwd level for the invoked RPC is defined as *user* (see `sgw_addrpctogrp`), TRS passes to the mainframe the user ID and password defined to TRS using the `sgw_addlog` procedure.

- If the security is *on*, (see the security configuration parameter), and the rpcpwd level for the invoked RPC is defined as *group* (see `sgw_addrpctogrp`), TRS passes to the mainframe the user ID and password defined to TRS using the `sgw_addtrngrp` procedure for the *trn* group of that particular rpc.

For more information about these value sets, see `sgw_addrpc`. 
Dropping an RPC

To drop an RPC, use the following procedure. The RPC must be idle to be dropped.

```
exec sgw_droprpc rpc_name
```

where `rpc_name` is the name of the RPC you intend to drop.

Example

The following example shows how to drop the sample RPC named SYD2:

```
exec sgw_droprpc SYD2
```

Configuring a default SQL language handler for TRS

To pass client SQL language requests through TRS to the DB2 on the mainframe, the TRS System Administrator must configure a default SQL language handler. This language handler is a TRS RPC that is mapped to an Open ServerConnect program on the mainframe which handles the interaction with DB2 UDB.

Sybase provides the following programs:

- For SQL language requests (including cursors, dynamic, and long-running transactions) on CICS, MainframeConnect for DB2 UDB using the CICS transaction name `AMD2`.
- For SQL language requests on IMS or MVS, OmniSQL Access Module for DB2 using the transaction name `SYRT`.

Use the instructions in this section to define the default SQL language handler RPC to TRS by specifying the mainframe transaction ID that handles language requests.

Defining a default SQL language handler

The values you provide for defining the default SQL language handler to TRS depend on the following:

- The environment on the mainframe (CICS, IMS, or MVS)
- The status of TRS security (enforced or not enforced)
Configuring a default SQL language handler for TRS

To define a default SQL language handler for your site configuration, use the `sgw_addrpc` procedure:

```
sgw_addrpc rpc_name, tran_id, region, security
```

Refer to Table 3-6 for the appropriate `rpc_name` and `tran_id` parameters, then set `region` and `security` to the appropriate values for your site.

**Table 3-6: Default SQL language handler settings by host and security settings**

<table>
<thead>
<tr>
<th>TRS Security Enforced</th>
<th>CICS host (MainframeConnect)</th>
<th>MVS host and IMS host (OmniSQL Access Module)</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>Set rpc_name = name you create</code></td>
<td><code>Set tran_id = AMD2</code></td>
<td><code>Set rpc_name = name you create</code></td>
</tr>
<tr>
<td><code>Set tran_id = AMD2</code></td>
<td></td>
<td><code>Set tran_id = SYRT</code></td>
</tr>
</tbody>
</table>

Note: If security is enforced, define the `rpc_name` that you create as the default SQL language handler when defining the `tran_group`. When security is not enforced, the default SQL language `rpc_name` must be SYRT.

**Example**

Following is an `isql` example of the `sgw_addrpc` procedure, which defines a default SQL language handler:

```
exec sgw_addrpc SYRT, AMD2, TESTREG, both
```

where:

- Language requests are routed to MainframeConnect for DB2 UDB, which has a predefined CICS `tran_id` of AMD2.
- TESTREG is the `region` parameter value that corresponds to the LU 6.2 or TCP/IP connection that provides access to the CICS region running MainframeConnect for DB2 UDB.

If TRS security were enforced for this example, the TRS system administrator would need to define a transaction group (using `sgw_addtrngrp`) with SYRT as the default language RPC. SYRT then would become the default SQL language handler for all TRS users assigned to that transaction group.
Defining multiple SQL language handlers

To send SQL language requests to more than one language RPC (that is, if you have copies of MainframeConnect for DB2 UDB or OmniSQL Access Module for DB2 in other regions), define multiple language handlers, each with a different RPC name and a different region name. Clients can explicitly specify the particular language RPC in the request.

Add the alternate language handlers using the `sgw_addrpc` procedure, using the following parameter values:

- An `rpc_name` of your choice
- The `tran_id` as defined in “Default SQL language handler settings by host and security settings” on page 54
- The `region` and `security` values appropriate for your site

The following `isql` example shows three procedures that define different RPCs for the AMD2 transactions at the DALLAS, DETROIT, and MIAMI regions, respectively. This example allows a user to specify the region to which TRS sends the language request: DALLASrpc uses the DALLAS region, DETROITrpc uses the DETROIT region, and MIAMIrpc uses the MIAMI region.

```
exec sgw_addrpc DALLASrpc, AMD2, DALLAS, both
go
exec sgw_addrpc DETROITrpc, AMD2, DETROIT, both
go
exec sgw_addrpc MIAMIrpc, AMD2, MIAMI, both
go
```

If security is enforced, add the preceding RPCs to the appropriate `tran_group` that the `tran_group` users can access.

For example:

```
exec sgw_addrpcctogrp TRANGRP2, DALLASrpc, user
go
exec sgw_addrpcctogrp TRANGRP2, DETROITrpc, user
go
exec sgw_addrpcctogrp TRANGRP2, MIAMIrpc, user
go
```

To obtain results from TESTREG with SYRT as the default (using the “Example” on page 54), a user invokes a SQL query program, such as `isql`, and enters a query similar to the following:

```
select * from payroll
go
```
Using Catalog Stored Procedures (CSPs)

To obtain results from MIAMI, a user enters the following execute command, specifying the RPC named MIAMIrpc:

```sql
exec MIAMIrpc "select * from payroll"
go
```

Adaptive Server stored procedure example

Following is an example of creating an Adaptive Server stored procedure that connects to TRS and uses a parameter to choose the DB2 UDB system to use:

```sql
create proc dblp1
  @salary int
as
  if @salary < 60000
    exec BLUETRS...MIAMIrpc "select *
    from payroll"
  else
    exec BLUETRS...DALLASrpc "select *
    from payroll"
```

In this example, the Adaptive Server is configured to connect to the TRS named BLUETRS and to execute an RPC. The value of the @salary parameter determines the language RPC that the TRS named BLUETRS uses to route the select statement. Based on the value of @salary, one of the following occurs:

- If the value of @salary is less than 60,000, the procedure sends the `select * from payroll` statement to the TRS named BLUETRS. In BLUETRS, the AMD2 transaction in MIAMI executes it. The AMD2 transaction in MIAMI is mapped to the RPC named MIAMIrpc on BLUETRS.
- If the value of @salary is greater than or equal to 60,000, the procedure executes against the AMD2 transaction in DALLAS, which is mapped to the RPC named DALLASrpc on BLUETRS.

Using Catalog Stored Procedures (CSPs)

CSPs serve as a uniform catalog interface for accessing the system tables of different database management systems, including Adaptive Server.
Sybase provides CSPs that can be defined to TRS with MainframeConnect for DB2 UDB. These CSPs correspond to transactions on the mainframe that access the DB2 UDB catalog and return information to a client application in a standard format.

CSPs are implemented as CICS transactions and must be configured as RPCs to TRS. See Chapter 4, “Accessing Catalog Information with CSPs,” for configuration instructions.

The following table outlines the functions provided by CSPs and the four-character CICS transaction name that each procedure name maps to on the host.

**Table 3-7: CSP functions**

<table>
<thead>
<tr>
<th>Procedure name</th>
<th>Related CICS transaction name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>sp_capabilities</td>
<td>SYBP</td>
<td>Returns information about the capabilities of the TRS</td>
</tr>
<tr>
<td>sp_columns</td>
<td>SYB3</td>
<td>Describes the columns of a table</td>
</tr>
<tr>
<td>sp_column_privileges</td>
<td>SYBA</td>
<td>Describes the column permissions of an object</td>
</tr>
<tr>
<td>sp_databases</td>
<td>SYB1</td>
<td>Lists the databases available</td>
</tr>
<tr>
<td>sp_datatype_info</td>
<td>SYBC</td>
<td>Describes the datatypes available</td>
</tr>
<tr>
<td>sp_fkeys</td>
<td>SYB8</td>
<td>Describes the primary key/foreign key relationships</td>
</tr>
<tr>
<td>sp_pkeys</td>
<td>SYB7</td>
<td>Describes the primary key for a table</td>
</tr>
<tr>
<td>sp_server_info</td>
<td>SYBB</td>
<td>Lists the configuration and capabilities</td>
</tr>
<tr>
<td>sp_special_columns</td>
<td>SYBD</td>
<td>Lists the optimal columns to uniquely identify rows and list columns that are automatically updated</td>
</tr>
<tr>
<td>sp_sproc_columns</td>
<td>SYB5</td>
<td>Describes the input/output of executable objects</td>
</tr>
<tr>
<td>sp_statistics</td>
<td>SYB6</td>
<td>Lists the index and statistics information about a table</td>
</tr>
<tr>
<td>sp_stored_procedures</td>
<td>SYB4</td>
<td>Lists the executable procedures</td>
</tr>
<tr>
<td>sp_table_privileges</td>
<td>SYB9</td>
<td>Describes the table permissions</td>
</tr>
<tr>
<td>sp_tables</td>
<td>SYB2</td>
<td>Lists the tables</td>
</tr>
<tr>
<td>sp_thread_props</td>
<td>SYBT</td>
<td>Returns minimal information at this time</td>
</tr>
</tbody>
</table>
For complete information about the mainframe installation, see the Mainframe Connect Installation and Administration Guide for DB2 UDB. For complete information about the syntax and operation of CSPs, see Chapter 4, “Accessing Catalog Information with CSPs,” in this guide.

CSP scripts

Sybase provides three scripts for you to use with CSPs:

- **addcat** - adds the CSPs to TRS.
- **dropcat** - drops the CSPs from TRS.
- **testcat** - tests the CSPs (requires that the AMD2 transaction be installed at the mainframe).

Installing CSPs

The addcat script executes the sgw_addrpc procedure automatically for each CSP (see “Adding an RPC” on page 50). Before you run addcat, modify the script to suit your installation.

Use your text editor to specify the value of these parameters:

- **region** parameter – name of the region you want the CSPs to execute against.
- **security** parameter – value you can change to meet the security requirements at your installation. If you do not change it, the value is *none*.
- **rpc_name** parameter – name or value must be coordinated with any change to the RPC names with the mainframe system programmer. If you are using ODBC applications, do not change the RPC names.
- **tran_id** parameter – value or name of this parameter must be coordinated with any change to the transaction ID with the mainframe system programmer.

After you edit the script to suit your installation, run the addcat script as input to your TRS. The following isql example shows how to run the addcat script with a TRS named new_TRS:

```sql
isql -Snew_TRS -Usa -P < addcat
```

```sql
go
```
This script automatically executes the `sgw_addrpc` procedure for each CSP.

**Testing CSPs**

The `testcat` script uses the AMD2 transaction to create temporary tables and execute each CSP. At least one row is returned for each CSP and the `testcat` script then drops the temporary tables.

**Example**

Run the `testcat` script as input to your TRS. The following `isql` example shows how to run the `testcat` script with a TRS named `new_TRS`:

```
isql -S new_TRS -Usa -P < testcat
```

This script automatically tests each of the CSPs.

**Dropping CSPs**

The `dropcat` script drops the CSPs from TRS. Run the `dropcat` script as input to your TRS.

**Example**

The following `isql` example shows how to run the `dropcat` script with a TRS named `new_TRS`:

```
isql -S new_TRS -Usa -P < dropcat
```

This script automatically drops the CSPs.
CHAPTER 4
Accessing Catalog Information with CSPs

To obtain information about database objects, you need to access the database catalog. Catalog Stored Procedures (CSPs) provide this catalog access. This chapter describes how to use CSPs to access the DB2 UDB catalog.

This chapter contains the following topics:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Using CSPs</td>
<td>61</td>
</tr>
<tr>
<td>Supported CSPs</td>
<td>65</td>
</tr>
<tr>
<td>sp_capabilities</td>
<td>66</td>
</tr>
<tr>
<td>sp_column_privileges</td>
<td>68</td>
</tr>
<tr>
<td>sp_columns</td>
<td>70</td>
</tr>
<tr>
<td>sp_databases</td>
<td>73</td>
</tr>
<tr>
<td>sp_datatype_info</td>
<td>74</td>
</tr>
<tr>
<td>sp_fkeys</td>
<td>76</td>
</tr>
<tr>
<td>sp_pkeys</td>
<td>78</td>
</tr>
<tr>
<td>sp_server_info</td>
<td>80</td>
</tr>
<tr>
<td>sp_special_columns</td>
<td>81</td>
</tr>
<tr>
<td>sp_sproc_columns</td>
<td>83</td>
</tr>
<tr>
<td>sp_statistics</td>
<td>85</td>
</tr>
<tr>
<td>sp_stored_procedures</td>
<td>87</td>
</tr>
<tr>
<td>sp_table_privileges</td>
<td>88</td>
</tr>
<tr>
<td>sp_tables</td>
<td>90</td>
</tr>
<tr>
<td>sp_thread_props</td>
<td>92</td>
</tr>
</tbody>
</table>

Using CSPs

This section describes the use of CSPs, coding instructions that apply to CSPs, and the use of wildcard-character search patterns.
Why use CSPs?

The catalog structures for DB2 UDB and Adaptive Server are different. If you have client applications written to access the SQL Server catalog, you may need to re-code the client application queries to send those queries directly to the DB2 UDB system tables. To avoid modifying your database-specific applications, you can use CSPs to access catalog information. CSPs are compatible with the catalog interface for the Open Database Connectivity (ODBC) Application Program Interface (API).

You need to install and configure CSPs in TRS for the proper functioning of the following clients:

- ODBC
- ASE/CIS

Coding instructions

This section includes general coding information that applies to all CSPs.

Parameters

CSPs have optional and required parameters. Required parameters must have values supplied; optional parameters default to predefined values.

The following rules apply to CSP parameters:

- Both positional and named parameters are supported, but not in the same statement.
- Parameter values can be enclosed in double quotes. Parameter values enclosed in quotes must be in the correct case for the target.
- Object names (table names, column names, and index names) can be created using lowercase letters. The target database automatically converts object names to uppercase unless the object names are enclosed in double quotes. However, when using CSPs, these object names must be referred to using upper-case names.

Syntax

A client application can initiate a CSP by issuing any of the following statements:
exec rpc_name parm1, parm2, . . .
execute rpc_name parm1, parm2, . . .

where:
• *rpc_name* is the name of the stored procedure (for example, *sp_columns*).
• *parm1* and *parm2* are parameter values required or desired for that stored procedure.

## Coding examples

You can execute CSPs with a language command or through an RPC event.

You can specify the parameters for a CSP in one of the following forms:

- Supply all of the parameters:
  
  ```
  sp_columns publishers, "dbo", "pubs2", "pub_id"
  ```

- Use “null” or a comma as a placeholder:
  
  ```
  sp_columns publishers, null, null,"pub_id"
  sp_columns publishers, , , "pub_id"
  ```

- Supply one or more parameters in the following form:

  ```
  @parameter_name = value
  ```

For example, to find information about a particular column, issue the following statement:

```
sp_columns @table_name = publishers, @column_name = "pub_id"
```

The parameter names in the syntax statement must match the parameter names defined by the CSP.

You can use the named parameter form if you process a CSP as an RPC event.

### Table name, owner, and qualifier parameters

This section explains how the parameters *table_name*, *table_owner*, and *tableQualifier* are used in this product:

- *table_name* is the name of the database object about which you want to retrieve catalog information.
- *table_owner* is the owner of the database object about which you want to retrieve catalog information.
The following shows how CSP parameters relate to the DB2 UDB subsystem.

*Figure 4-1: Relationship of CSP parameters and DB2 UDB*

**Wildcard-character search patterns**

The percent (%) wildcard character can be used in parameters that allow wildcard-character search patterns. This wildcard represents any string of zero or more characters. When using CSPs, the wildcard expression must be enclosed in quotation marks.

If the percent (%) character is used in parameters that do not allow wildcard-character search patterns, you will receive a syntax error.

The following table shows some examples of the percent (%) wildcard character and its use:

*Table 4-1: Wild card character examples*

<table>
<thead>
<tr>
<th>Sample string</th>
<th>Matches</th>
</tr>
</thead>
<tbody>
<tr>
<td>“%A%”</td>
<td>All names that contain the letter “A”; for example, A, AT, CAT</td>
</tr>
<tr>
<td>“%”</td>
<td>All names</td>
</tr>
</tbody>
</table>
Escape character

To use a wildcard character as a literal, precede it with an “@” (at) sign. If the parameter normally accepts the wildcard character, you can mix the percent (%) wildcard character with escaped wildcard characters (@ and %) interpreted as literals. If the parameter does not accept the wildcard character, an @ (at) sign must precede the wildcard character to use the character as a literal.

Supported CSPs

The following table shows the supported CSPs and the information that each CSP retrieves.

<table>
<thead>
<tr>
<th>CSP</th>
<th>Information retrieved by the CSP</th>
</tr>
</thead>
<tbody>
<tr>
<td>sp_capabilities</td>
<td>Returns the SQL capabilities of a DB2 access service</td>
</tr>
<tr>
<td>sp_column_privileges</td>
<td>Column privilege information for one table</td>
</tr>
<tr>
<td>sp_columns</td>
<td>Column descriptions for a table</td>
</tr>
<tr>
<td>sp_databases</td>
<td>List of available databases</td>
</tr>
<tr>
<td>sp_datatype_info</td>
<td>Datatype descriptions</td>
</tr>
<tr>
<td>sp_fkeys</td>
<td>Foreign and primary key relationships</td>
</tr>
<tr>
<td>sp_pkeys</td>
<td>Primary key information for a single table</td>
</tr>
<tr>
<td>sp_server_info</td>
<td>Server terms, limits, and capabilities</td>
</tr>
<tr>
<td>sp_special_columns</td>
<td>Additional column information</td>
</tr>
<tr>
<td>sp_sproc_columns</td>
<td>Attributes of procedures input and return parameters</td>
</tr>
<tr>
<td>sp_statistics</td>
<td>Statistics and indexes for one table</td>
</tr>
<tr>
<td>sp_stored_procedures</td>
<td>List of available procedures</td>
</tr>
<tr>
<td>sp_table_privileges</td>
<td>Table privilege information for one table</td>
</tr>
<tr>
<td>sp_tables</td>
<td>List of aliases, synonyms, tables, views, and system tables</td>
</tr>
</tbody>
</table>

The following sections provide descriptions, syntax, parameters, and usage for the supported CSPs.
**sp_capabilities**

**Description**
Returns the SQL capabilities of a DB2 access service.

**Syntax**
sp_capabilities

**Parameters**
None.

This procedure does not allow parameters.

**Usage**
The result set contains information that allows applications to successfully interact with an DB2 access service during normal query processing.

**Results**
The following table shows the result set:

<table>
<thead>
<tr>
<th>Column</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>int</td>
<td>Capability ID</td>
</tr>
<tr>
<td>CAPABILITY_NAME</td>
<td>char(30)</td>
<td>Capability name</td>
</tr>
<tr>
<td>VALUE</td>
<td>int</td>
<td>Capability value</td>
</tr>
<tr>
<td>DESCRIPTION</td>
<td>char(128)</td>
<td>Capability description</td>
</tr>
</tbody>
</table>

The following table shows the ID and values for several DB2 access service functional capabilities:

<table>
<thead>
<tr>
<th>ID</th>
<th>Capability</th>
<th>Value description</th>
</tr>
</thead>
<tbody>
<tr>
<td>101</td>
<td>SQL syntax</td>
<td>1=Sybase T-SQL supported</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2=DB2 SQL supported</td>
</tr>
<tr>
<td>102</td>
<td>Join handling</td>
<td>0=Unsupported</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1=No outer join supported</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2=T-SQL support</td>
</tr>
<tr>
<td>103</td>
<td>Aggregate handling</td>
<td>0=Unsupported</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1=ANSI supported</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2=All functions</td>
</tr>
<tr>
<td>104</td>
<td>AND predicates</td>
<td>0=Unsupported</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1=Supported</td>
</tr>
<tr>
<td>105</td>
<td>OR predicates</td>
<td>0=Unsupported</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1=Supported</td>
</tr>
<tr>
<td>106</td>
<td>LIKE predicates</td>
<td>0=Unsupported</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1=ANSI-style supported</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2=T-SQL supported</td>
</tr>
<tr>
<td>107</td>
<td>Bulk insert handling</td>
<td>0=Unsupported</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1=Supported</td>
</tr>
<tr>
<td>ID</td>
<td>Capability</td>
<td>Value description</td>
</tr>
<tr>
<td>------</td>
<td>-----------------------------------</td>
<td>--------------------------------------------------------</td>
</tr>
<tr>
<td>108</td>
<td>Text and image handling</td>
<td>0=Unsupported 1=Text, no textptr 2=Text and textptr</td>
</tr>
<tr>
<td>109</td>
<td>Transaction handling</td>
<td>0=Unsupported 1=Local supported 2=Two-phase commit supported</td>
</tr>
<tr>
<td>110</td>
<td>Text pattern handling</td>
<td>0=Unsupported 1=Pattern (text) supported</td>
</tr>
<tr>
<td>111</td>
<td>order by</td>
<td>0=Unsupported 1=Supported</td>
</tr>
<tr>
<td>112</td>
<td>group by</td>
<td>0=Unsupported 1=ANSI supported 2=T-SQL supported</td>
</tr>
<tr>
<td>113</td>
<td>Net password encryption</td>
<td>0=Unsupported 1=Supported</td>
</tr>
<tr>
<td>114</td>
<td>Object case sensitivity</td>
<td>0=Case insensitive 1=Case sensitive</td>
</tr>
<tr>
<td>115</td>
<td>distinct</td>
<td>0=Unsupported 1=Supported</td>
</tr>
<tr>
<td>116</td>
<td>Wild card escape</td>
<td>0=Unsupported 1=Supported Non-zero=Escape_char(s)</td>
</tr>
<tr>
<td>117</td>
<td>Union handling</td>
<td>0=Unsupported 1=Supported</td>
</tr>
<tr>
<td>118</td>
<td>String functions</td>
<td>0=Unsupported 1=Substring supported 2=Oracle subset supported 3=T-SQL supported</td>
</tr>
<tr>
<td>119</td>
<td>Expression handling</td>
<td>0=Unsupported 1=ANSI supported 2=T-SQL supported</td>
</tr>
<tr>
<td>120</td>
<td>Character truncation</td>
<td>0=Fixed length character parameters may contain trailing blanks 1=Fixed length character parameters will not contain trailing blanks</td>
</tr>
<tr>
<td>121</td>
<td>Language events</td>
<td>0=Unsupported 1=T-SQL DML without datetime in the where clause supported 2=T-SQL DML supported</td>
</tr>
<tr>
<td>122</td>
<td>Date functions</td>
<td>0=Unsupported 1=T-SQL date functions supported</td>
</tr>
</tbody>
</table>
### `sp_column_privileges`

**Description**
Returns column privilege information for a single database object.

**Syntax**
```
sp_column_privileges table_name [, table_owner] [, table_qualifier] [, column_name]
```

**Parameters**
- `table_name`
  is the name of the table. Wildcard-character search patterns and aliases are not supported. Views are supported but do not include alter or index privileges.

- `table_owner`
  is the name of the table owner. Wildcard-character search patterns are not supported.

- `table_qualifier`
  is ignored. Leave blank or set to NULL.

- `column_name`
  is the name of the column for which you want privilege information. Use wildcard-character search patterns to request information about more than one column. Leave blank or set to NULL to request information about all columns in the table or tables.

**Usage**
- This function corresponds to the ODBC function `SQLColumnPrivileges`.

---

### ID Capability Value description

<table>
<thead>
<tr>
<th>ID</th>
<th>Capability</th>
<th>Value description</th>
</tr>
</thead>
</table>
| 123 | Math functions              | 0=Unsupported  
                                    1=Oracle functions supported 
                                    2=T-SQL math functions supported |
| 124 | T-SQL convert functions     | 0=Unsupported  
                                    1=Supported |
| 125 | T-SQL delete/update         | 0=Sybase extensions not supported 
                                    1=Sybase extensions supported |
| 126 | Insert/select handling      | 0=Unsupported  
                                    1=Supported |
| 127 | Subquery handling           | 0=Unsupported  
                                    1=Supported |
| 128 | IN/NOT IN support           | 0=Unsupported  
                                    1=Supported |
| 129 | CASE support                | 0=Unsupported  
                                    1=Supported |
Information is based on the SYSCOLAUTH, SYSCOLUMNS, and SYSTABAUTH system catalog tables.

Results

sp_column_privileges returns one row for each privilege a user has on a column in a table. Results are ordered by the following columns:

- **TABLE_OWNER**
- **TABLE_NAME**
- **COLUMN_NAME**
- **PRIVILEGE**

The lengths for varchar columns shown in the result set tables are maximums; the actual lengths depend on the target database.

The following table shows the result set for **sp_column_privileges**.

<table>
<thead>
<tr>
<th>Column name</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TABLE_QUALIFIER</td>
<td>varchar (128)</td>
<td>Always NULL.</td>
</tr>
<tr>
<td>TABLE_OWNER</td>
<td>varchar (128)</td>
<td>Authorization ID.</td>
</tr>
<tr>
<td>TABLE_NAME</td>
<td>varchar (128)</td>
<td>Name of the object about which privilege information is returned.</td>
</tr>
<tr>
<td>COLUMN_NAME</td>
<td>varchar (128)</td>
<td>NOT NULL. Column name.</td>
</tr>
<tr>
<td>GRANTOR</td>
<td>varchar (128)</td>
<td>Identifies the user who granted this privilege.</td>
</tr>
<tr>
<td>GRANTEE</td>
<td>varchar (128)</td>
<td>NOT NULL. Identifies the user to whom this privilege was granted.</td>
</tr>
</tbody>
</table>
| PRIVILEGE        | varchar (128)| NOT NULL. Identifies the privilege granted to the grantee on this column as one of the following values:  
|                  |              | - SELECT if the grantee is authorized to select rows in the associated object. |
|                  |              | - UPDATE if the grantee is authorized to insert and update rows in the associated object. |
| IS_GRANTABLE     | varchar (3)  | Indicates whether the grantee is authorized to grant privilege on this column to other users; always NULL. |
sp_columns

Description
Returns information about the type of data that can be stored in one or more columns.

Syntax
sp_columns table_name [, table_owner]
[ , table_qualifier] [, column_name]

Parameters

table_name
is the table name. Use the wildcard character to request information about more than one table. Aliases are not supported.

table_owner
is the owner of the database object about which column information is requested. Use the wildcard character to request information about tables owned by more than one user. If you do not specify a table owner, sp_columns looks first for tables owned by the current user and then for tables owned by the database owner.

table_qualifier
is ignored. Leave blank or set to NULL.

column_name
is the name of the column for which you want information. Use the wildcard character to request information about more than one column. Leave empty or set to NULL to request information about all columns in the table or tables.

Usage
• If column_name is provided, sp_columns returns information only for the column or columns that match.
• This function corresponds to the ODBC function SQLColumns.
• Information is based on the SYSCOLUMNS and SYSSYNONYMS system catalog tables.

Results
sp_columns returns one row containing a description of each column in a table. Results are ordered by the following columns:

• TABLE_OWNER
• TABLE_NAME

The lengths for varchar columns shown in the result set tables are maximums; the actual lengths depend on the target database.

The following table shows the result set for sp_columns.
Table 4-6: Result set for sp_columns

<table>
<thead>
<tr>
<th>Column</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TABLE_QUALIFIER</td>
<td>varchar(128)</td>
<td>Always NULL</td>
</tr>
<tr>
<td>TABLE_OWNER</td>
<td>varchar(128)</td>
<td>Table owner identifier</td>
</tr>
<tr>
<td>TABLE_NAME</td>
<td>varchar(128)</td>
<td>Table name</td>
</tr>
<tr>
<td>COLUMN_NAME</td>
<td>varchar(128)</td>
<td>Column name</td>
</tr>
<tr>
<td>DATA_TYPE</td>
<td>smallint</td>
<td>Integer code for the ODBC datatype</td>
</tr>
<tr>
<td>TYPE_NAME</td>
<td>varchar(128)</td>
<td>String representing the datatype name in the target database</td>
</tr>
<tr>
<td>PRECISION</td>
<td>int</td>
<td>Number of significant digits of the column on the target database</td>
</tr>
<tr>
<td>LENGTH</td>
<td>int</td>
<td>Length of the column in bytes</td>
</tr>
<tr>
<td>SCALE</td>
<td>smallint</td>
<td>Number of digits to the right of the decimal point</td>
</tr>
<tr>
<td>RADIX</td>
<td>smallint</td>
<td>Base for numeric types</td>
</tr>
<tr>
<td>NULLABLE</td>
<td>smallint</td>
<td>Indicates whether the column accepts NULL values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 SQL_NO_NULLS if the column does not accept NULL values</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 SQL_NULLABLE if the column accepts NULL values</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 SQL_NULLABLE_UNKNOWN if it is not known if the column accepts NULL values</td>
</tr>
<tr>
<td>REMARKS</td>
<td>varchar(254)</td>
<td>A description of the column</td>
</tr>
<tr>
<td>SS_DATA_TYPE</td>
<td>smallint</td>
<td>The SQL Server datatype name</td>
</tr>
<tr>
<td>COLID</td>
<td>smallint</td>
<td>The column ID number</td>
</tr>
<tr>
<td>REMOTE_DATA_TYPE</td>
<td>int</td>
<td>An integer representing the underlying target database datatype (composite value)</td>
</tr>
</tbody>
</table>

**ODBC Datatypes**

The following table describes the DB2 UDB datatypes and matching ODBC integer identifiers that are returned in the TYPE_NAME and DATA_TYPE columns of the sp_columns, sp_datatype_info, sp_special_columns, and sp_sproc_columns result sets.
### Table 4-7: ODBC datatypes

<table>
<thead>
<tr>
<th>DB2 UDB datatype</th>
<th>Target datatype maximum physical length</th>
<th>ODBC type</th>
<th>ODBC integer ID</th>
<th>DB2 UDB datatype description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHARACTER( ) FOR BIT DATA</td>
<td>254</td>
<td>SQL_BINARY</td>
<td>-2</td>
<td>Fixed length character for bit data</td>
</tr>
<tr>
<td>VARCHAR( ) FOR BIT DATA</td>
<td>254</td>
<td>SQL_VARBINARY</td>
<td>-3</td>
<td>Variable length character for bit data</td>
</tr>
<tr>
<td>LONG VARCHAR FOR BIT DATA</td>
<td>32714</td>
<td>SQL_LONGVARBINARY</td>
<td>-4</td>
<td>Variable length character for bit data</td>
</tr>
<tr>
<td>CHARACTER( )</td>
<td>254</td>
<td>SQL_CHAR</td>
<td>1</td>
<td>Fixed length character</td>
</tr>
<tr>
<td>VARCHAR()</td>
<td>254</td>
<td>SQL_VARCHAR</td>
<td>12</td>
<td>Variable length character</td>
</tr>
<tr>
<td>LONG VARCHAR()</td>
<td>32714</td>
<td>SQL_LONGVARCHAR</td>
<td>-1</td>
<td>Variable length character</td>
</tr>
<tr>
<td>CHARACTER( ) FOR MIXED DATA</td>
<td>254</td>
<td>SQL_BINARY</td>
<td>-2</td>
<td>Fixed length character (DBCS or SBCS)</td>
</tr>
<tr>
<td>VARCHAR( ) FOR MIXED DATA</td>
<td>254</td>
<td>SQL_VARBINARY</td>
<td>-3</td>
<td>Variable length character (DBCS or SBCS)</td>
</tr>
<tr>
<td>LONG VARCHAR( ) FOR MIXED DATA</td>
<td>32714</td>
<td>SQL_LONGVARBINARY</td>
<td>-4</td>
<td>Variable length character (DBCS or SBCS)</td>
</tr>
<tr>
<td>GRAPHIC( )</td>
<td>127</td>
<td>SQL_BINARY</td>
<td>-2</td>
<td>Fixed length graphic (DBCS)</td>
</tr>
<tr>
<td>VARGRAPHIC( )</td>
<td>127</td>
<td>SQL_VARBINARY</td>
<td>-3</td>
<td>Variable length graphic (DBCS)</td>
</tr>
<tr>
<td>LONG VARGRAPHIC</td>
<td>16357</td>
<td>SQL_LONGVARBINARY</td>
<td>-4</td>
<td>Variable length graphic (DBCS)</td>
</tr>
<tr>
<td>SMALLINT</td>
<td>2</td>
<td>SQL_SMALLINT</td>
<td>5</td>
<td>2-byte binary integer</td>
</tr>
<tr>
<td>INTEGER</td>
<td>4</td>
<td>SQL_INTEGER</td>
<td>4</td>
<td>4-byte binary integer</td>
</tr>
<tr>
<td>REAL</td>
<td>4</td>
<td>SQL_REAL</td>
<td>7</td>
<td>4-byte floating point</td>
</tr>
<tr>
<td>FLOAT()</td>
<td>4</td>
<td>SQL_REAL</td>
<td>7</td>
<td>4-byte floating point with a precision less than 22</td>
</tr>
<tr>
<td>FLOAT()</td>
<td>8</td>
<td>SQL_DOUBLE</td>
<td>8</td>
<td>8-byte floating point with a precision equal to or greater than 22</td>
</tr>
<tr>
<td>DOUBLE PRECISION</td>
<td>8</td>
<td>SQL_DOUBLE</td>
<td>8</td>
<td>8-byte floating point</td>
</tr>
</tbody>
</table>
The `REMOTE_DATATYPE` column contains a 32-bit composite datatype value that represents the target database datatype. The following table describes the datatype value.

<table>
<thead>
<tr>
<th>DB2 UDB datatype</th>
<th>Target datatype maximum physical length</th>
<th>ODBC type</th>
<th>ODBC integer ID</th>
<th>DB2 UDB datatype description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DECIMAL()</td>
<td>31</td>
<td>SQL_DECIMAL</td>
<td>3</td>
<td>Packed decimal number</td>
</tr>
<tr>
<td>NUMERIC</td>
<td>31</td>
<td>SQL_NUMERIC</td>
<td>2</td>
<td>Zoned decimal number</td>
</tr>
<tr>
<td>DATE</td>
<td>10</td>
<td>SQL_DATE</td>
<td>9</td>
<td>Date</td>
</tr>
<tr>
<td>TIME</td>
<td>8</td>
<td>SQL_TIME</td>
<td>10</td>
<td>Time</td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td>26</td>
<td>SQL_DATETIME</td>
<td>11</td>
<td>Timestamp</td>
</tr>
</tbody>
</table>

**REMOTE_DATATYPE**

The `REMOTE_DATATYPE` column contains a 32-bit composite datatype value that represents the target database datatype. The following table describes the datatype value.

<table>
<thead>
<tr>
<th>Bit(s)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bits 0-7</td>
<td>ODBC (target) datatype (can be extended for types not defined in ODBC)</td>
</tr>
<tr>
<td>Bit 8</td>
<td>Returns 1 if nullable, 0 if not nullable</td>
</tr>
<tr>
<td>Bit 9</td>
<td>Returns 1 if case sensitive, 0 if not case sensitive</td>
</tr>
<tr>
<td>Bits 10, 11</td>
<td>Always returns 10 (binary) meaning updatability unknown</td>
</tr>
<tr>
<td>Bits 12, 13</td>
<td>Reserved, always returns 00 (binary)</td>
</tr>
<tr>
<td>Bits 14, 15</td>
<td>Returns the following: 10 (binary) meaning NEWUSERSTYPE (used for REAL)</td>
</tr>
</tbody>
</table>

For numeric types:
- Bits 16-23: Precision
- Bits 24-31: Scale

For non-numeric types:
- Bits 16-31: Length

**sp_databases**

**Description**

Returns a list of databases on a target DBMS.
**sp_databases**

**Syntax**

```
sp_databases
```

**Parameters**

None.

This procedure does not allow parameters.

**Usage**

Information is based on the SYSDATABASE system catalog table.

**Results**

`sp_databases` returns a list of databases available to the client. Results are ordered by DATABASE_NAME.

The lengths for varchar columns shown in the result set tables are maximums; the actual lengths depend on the target database.

The following table shows the result set for `sp_databases`.

<table>
<thead>
<tr>
<th>Column</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATABASE_NAME</td>
<td>varchar(32)</td>
<td>Name of an available database</td>
</tr>
<tr>
<td></td>
<td>NOT NULL</td>
<td></td>
</tr>
<tr>
<td>DATABASE_SIZE</td>
<td>int</td>
<td>Size of the named database in kilobytes, otherwise NULL</td>
</tr>
<tr>
<td>REMARKS</td>
<td>varchar(254)</td>
<td>Always NULL</td>
</tr>
</tbody>
</table>

**sp_datatype_info**

**Description**

Returns information about a particular datatype or about all supported datatypes.

**Syntax**

```
sp_datatype_info [data_type]
```

**Parameters**

`data_type`

is the ODBC code number for the specified datatype about which `sp_datatype_info` returns information. See Table 4-7 on page 72 for a description of these codes.

**Usage**

- The `data_type` parameter specifies the ODBC datatype for which information is requested. If this parameter is not provided, `sp_datatype_info` returns information about all supported datatypes.

- This function corresponds to the ODBC function SQLGetTypeInfo.

**Results**

`sp_datatype_info` returns a list of datatypes with information about each. Results are ordered by the following columns:
The lengths for varchar columns shown in the result set tables are maximums; the actual lengths depend on the target database. The following table shows the result set for `sp_datatype_info`.

**Table 4-10: Result set for sp_datatype_info**

<table>
<thead>
<tr>
<th>Column</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TYPE_NAME</td>
<td>varchar(128)</td>
<td>Name of the T-SQL datatype or the target database datatype that corresponds to the ODBC datatype in the DATA_TYPE column.</td>
</tr>
<tr>
<td></td>
<td>NOT NULL</td>
<td></td>
</tr>
<tr>
<td>DATA_TYPE</td>
<td>smallint</td>
<td>ODBC datatype to which all columns of this type are mapped.</td>
</tr>
<tr>
<td></td>
<td>NOT NULL</td>
<td></td>
</tr>
<tr>
<td>PRECISION</td>
<td>int</td>
<td>Maximum precision allowed for this datatype. (NULL is returned for datatypes where precision is not applicable.)</td>
</tr>
<tr>
<td>LITERAL_PREFIX</td>
<td>varchar(128)</td>
<td>Character(s) used to prefix a literal; NULL is returned for datatypes where a literal prefix is not applicable.</td>
</tr>
<tr>
<td>LITERAL_SUFFIX</td>
<td>varchar(128)</td>
<td>Character(s) used to mark the end of a literal; NULL is returned for datatypes where a literal suffix is not applicable.</td>
</tr>
<tr>
<td>CREATE_PARAMS</td>
<td>varchar(128)</td>
<td>Description of the creation parameters required for this datatype, for example: precision and scale; NULL is returned if the datatype does not have creation parameters.</td>
</tr>
<tr>
<td>NULLABLE</td>
<td>smallint</td>
<td>Indicates whether the datatype accepts NULL values:</td>
</tr>
<tr>
<td></td>
<td>NOT NULL</td>
<td>• 0 – the column does not accept NULL values.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 1 – the column accepts NULL values.</td>
</tr>
<tr>
<td>CASE_SENSITIVE</td>
<td>smallint</td>
<td>Indicates whether the datatype distinguishes between uppercase and lowercase characters:</td>
</tr>
<tr>
<td></td>
<td>NOT NULL</td>
<td>• 0 – the datatype is not a character type or is not case sensitive.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 1 – the datatype is a character type and is case sensitive.</td>
</tr>
<tr>
<td>SEARCHABLE</td>
<td>smallint</td>
<td>Indicates how this datatype is used in <em>where</em> clauses:</td>
</tr>
<tr>
<td></td>
<td>NOT NULL</td>
<td>• 0 – the datatype cannot be used in a <em>where</em> clause.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 1 – the datatype can be used in a <em>where</em> clause.</td>
</tr>
</tbody>
</table>
sp_fkeys

Description
Returns primary and foreign key information for the specified table or tables. Foreign keys must be declared using the ANSI integrity constraint mechanism.

Syntax
```
sp_fkeys pktable_name [, pktable_owner] [, pktable_qualifier] [, fktable_name] [, fktable_owner] [, fktable_qualifier]
```

Parameters
- `pktable_name` is the name of the table containing the primary key. Views, aliases, and wildcard-character search patterns are not supported. You must specify either this parameter or the `fktable_name` parameter, or both.

<table>
<thead>
<tr>
<th>Column</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
</table>
| UNSIGNED_ATTRIBUTE    | smallint | Indicates whether this attribute is unsigned:  
  • 0 – the datatype is signed.  
  • 1 – the datatype is unsigned.  
  • NULL – the datatype is not numeric. |
| MONEY                 | smallint | Indicates whether this is a money datatype:  
  • 0 – it is not a money datatype.  
  • 1 – it is a money datatype. |
| AUTO_INCREMENT        | smallint | Indicates whether this datatype automatically increments:  
  • 0 – columns of this datatype do not automatically increment.  
  • 1 – columns of this datatype automatically increment.  
  • NULL – the column is not numeric and does not have a sign. |
| LOCAL_TYPE_NAME       | varchar 128 | The database name or the T-SQL name for the datatype. |
| MINIMUM_SCALE         | smallint | Minimum scale for the datatype; NULL if scale is not applicable. |
| MAXIMUM_SCALE         | smallint | Maximum scale for the datatype; NULL if scale is not applicable. |
**pktable_owner**

is the owner of the table containing the primary key. Wildcard-character search patterns are not supported. If you do not specify this parameter, `sp_fkeys` looks first for a table owned by the current user and then for a table owned by the database owner.

**pktable_qualifier**

is ignored. Leave blank or set to NULL.

**fktable_name**

is the name of the table containing the foreign key. Views, aliases, and wildcard-character search patterns are not supported.

**fktable_owner**

is the owner of the table containing the foreign key. Views, aliases, and wildcard-character search patterns are not supported. If you do not specify this parameter, `sp_fkeys` looks first for a table owned by the current user and then for a table owned by the database owner.

**fktable_qualifier**

is ignored. Leave blank or set to NULL.

**Usage**

- This function corresponds to the ODBC function `SQLForeignKeys`.
- Information is based on the `SYSCOLUMNS`, `SYSTYPEFOREIGNKEYS`, `SYSINDEXES`, `SYSRELs`, and `SYSSYNONYMS` system catalog tables.
- For information about creating a foreign key, see the appropriate `IBM DATABASE 2 SQL Reference` manual.

**Results**

`sp_fkeys` returns a row for each column that is part of the foreign key or primary key in a primary key/foreign key relationship.

Results are ordered by the following columns:

- **PKTABLE_OWNER**
- **PKTABLE_NAME**
- **KEY_SEQ**

The lengths for varchar columns shown in the result set tables are maximums; the actual lengths depend on the target database.

The following table shows the result set for `sp_fkeys`. 
Table 4-11: Result set for sp_pkeys

<table>
<thead>
<tr>
<th>Column</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PKTABLE_QUALIFIER</td>
<td>varchar(128)</td>
<td>NULL</td>
</tr>
<tr>
<td>PKTABLE_OWNER</td>
<td>varchar(128)</td>
<td>Primary key table owner</td>
</tr>
<tr>
<td>PKTABLE_NAME</td>
<td>varchar(128)</td>
<td>NOT NULL Primary key table name</td>
</tr>
<tr>
<td>PKCOLUMN_NAME</td>
<td>varchar(128)</td>
<td>NOT NULL Primary key column name</td>
</tr>
<tr>
<td>FKTABLE_QUALIFIER</td>
<td>varchar(128)</td>
<td>NULL</td>
</tr>
<tr>
<td>FKTABLE_OWNER</td>
<td>varchar(128)</td>
<td>Foreign key table owner</td>
</tr>
<tr>
<td>FKTABLE_NAME</td>
<td>varchar(128)</td>
<td>NOT NULL Foreign key table name</td>
</tr>
<tr>
<td>FKCOLUMN_NAME</td>
<td>varchar(128)</td>
<td>NOT NULL Foreign key column name</td>
</tr>
<tr>
<td>KEY_SEQ</td>
<td>smallint</td>
<td>NOT NULL Column sequence number in key (starting with 1)</td>
</tr>
<tr>
<td>UPDATE_RULE</td>
<td>smallint</td>
<td>Action to be applied to the foreign key when the SQL operation is update:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 0 means cascade</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 1 means restrict</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 2 means set null</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• NULL means not applicable to the target database</td>
</tr>
<tr>
<td>DELETE_RULE</td>
<td>smallint</td>
<td>Action to be applied to the foreign key when the SQL operation is delete:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 0 means cascade</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 1 means restrict</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 2 means set null</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• NULL means not applicable to the target database</td>
</tr>
<tr>
<td>FK_NAME</td>
<td>varchar(128)</td>
<td>Foreign key identifier; NULL if not applicable to the target database</td>
</tr>
<tr>
<td>PK_NAME</td>
<td>varchar(128)</td>
<td>Primary key identifier; NULL if not applicable to the target database</td>
</tr>
</tbody>
</table>

**sp_pkeys**

Description  
Returns primary key information for the specified table or tables.

Syntax  
```
sp_pkeys table_name [ , table_owner]
[ , table_qualifier]
```
Parameters

`table_name`

is the name of the table. Wildcard-character search patterns are not supported. Views and aliases are not supported.

`table_owner`

is the owner of the table. Wildcard-character search patterns are not supported. If you do not specify this parameter, `sp_fkeys` looks first for a table owned by the current user and then for a table owned by the database owner.

`table_qualifier`

is ignored. Leave blank or set to NULL.

Usage

- This function corresponds to the ODBC function `SQLPrimaryKeys`.
- Information is based on the `SYSINDEXES`, `SYSKEYS`, and `SYSSYNONYMS` system catalog tables.
- For information about creating a foreign key, see the appropriate `IBM DATABASE 2 SQL Reference`.

Results

`sp_pkeys` returns a row for each column in the primary key. Results are ordered by:

- `TABLE_OWNER`
- `TABLE_NAME`
- `KEY_SEQ`

The lengths for `varchar` columns shown in the result set tables are maximums; the actual lengths depend on the target database.

The following table shows the result set for `sp_pkeys`. 
**Table 4-12: Result set for sp_pkeys**

<table>
<thead>
<tr>
<th>Column</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TABLE_QUALIFIER</td>
<td>varchar(128)</td>
<td>NULL</td>
</tr>
<tr>
<td>TABLE_OWNER</td>
<td>varchar(128)</td>
<td>Primary key table owner (authorization ID)</td>
</tr>
<tr>
<td>TABLE_NAME</td>
<td>varchar(128)</td>
<td>Primary key table name</td>
</tr>
<tr>
<td></td>
<td>NOT NULL</td>
<td></td>
</tr>
<tr>
<td>COLUMN_NAME</td>
<td>varchar(128)</td>
<td>Primary key column name</td>
</tr>
<tr>
<td></td>
<td>NOT NULL</td>
<td></td>
</tr>
<tr>
<td>KEY_SEQ</td>
<td>smallint</td>
<td>Sequence number of the column in a multi-column primary key</td>
</tr>
<tr>
<td></td>
<td>NOT NULL</td>
<td></td>
</tr>
<tr>
<td>PK_NAME</td>
<td>varchar(128)</td>
<td>Primary key identifier; NULL if not applicable to the target database</td>
</tr>
</tbody>
</table>

**sp_server_info**

**Description**
Returns a list of attribute names and matching values for the target DBMS.

**Syntax**
sp_server_info [attribute_id]

**Parameters**

attribute_id
is the integer ID of the attribute. Wildcard-character search patterns are not supported.

**Usage**

- If the attribute_id parameter is not provided, sp_server_info returns information about all attributes.
- This function does not correspond to any ODBC function, but returns some of the information returned by SQLGetInfo.

**Results**

sp_server_info returns a list of the requested attributes and their values.

The lengths for varchar columns shown in the result set tables are maximums; the actual lengths depend on the target database.

The following table shows the result set for sp_server_info.
Table 4-13: Result set for sp_server_info

<table>
<thead>
<tr>
<th>Column</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATTRIBUTE_ID</td>
<td>int</td>
<td>Numeric identifier of the attribute</td>
</tr>
<tr>
<td></td>
<td>NOT NULL</td>
<td></td>
</tr>
<tr>
<td>ATTRIBUTE_NAME</td>
<td>varchar(60)</td>
<td>Attribute name</td>
</tr>
<tr>
<td>ATTRIBUTE_VALUE</td>
<td>varchar(254)</td>
<td>Attribute value</td>
</tr>
</tbody>
</table>

**sp_special_columns**

**Description**
Retrieves the following information about columns within a specified table or view:

- The optimal set of columns that uniquely identify a row in the table or view
- A list of the columns that are automatically updated when any value in the row is updated

**Syntax**
```sql
sp_special_columns table_name [, table_owner] [, table_qualifier] [, col_type]
```

**Parameters**
- `table_name`
  is the name of the table. Views, aliases, and wildcard-character search patterns are not supported.

- `table_owner`
  is the owner of the table. Wildcard-character search patterns are not supported. If you do not specify this parameter, `sp_special_columns` looks first for a table owned by the current user and then for a table owned by the database owner.

- `table_qualifier`
  is ignored. Leave blank or set to NULL.

- `col_type`
  is a value that requests information about columns of a specific type as follows:
  - `R` returns information about columns with values that uniquely identify any row in the table.
  - `V` returns information about columns with values that are automatically generated by a target each time a row is inserted or updated.

**Usage**
- This function corresponds to the ODBC function SQLSpecialColumns.
sp_special_columns

- Information is based on the SYSINDEXES, SYSKEYS, and SYSCOLUMNS system catalog tables.

Results

sp_special_columns returns information about the columns that uniquely identify a row in a table.

The result set consists of a row for each column of an index that uniquely identifies each row of the table. If there are multiple unique indexes on a table, the one that is described by the result set is the first that exists in the following list:

- A primary key with clustered index
- A primary key without clustered index
- A unique, clustered index
- A unique, non-clustered index

The result set is ordered by the column name in the index.

The lengths for varchar columns shown in the result set tables are maximums; the actual lengths depend on the target database.

The following table shows the result set for sp_special_columns.

<table>
<thead>
<tr>
<th>Column</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCOPE</td>
<td>smallint</td>
<td>Actual scope of the row ID:</td>
</tr>
<tr>
<td></td>
<td>NOT NULL</td>
<td>• 0 SQL_SCOPE_CURROW</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 1 SQL_SCOPE_TRANSACTION</td>
</tr>
<tr>
<td>COLUMN_NAME</td>
<td>varchar(128)</td>
<td>Column name</td>
</tr>
<tr>
<td></td>
<td>NOT NULL</td>
<td></td>
</tr>
<tr>
<td>DATA_TYPE</td>
<td>smallint</td>
<td>ODBC datatype to which all columns of this type are mapped</td>
</tr>
<tr>
<td></td>
<td>NOT NULL</td>
<td></td>
</tr>
<tr>
<td>TYPE_NAME</td>
<td>varchar(128)</td>
<td>Name of the target database datatype that corresponds to the ODBC datatype in</td>
</tr>
<tr>
<td></td>
<td>NOT NULL</td>
<td>the DATA_TYPE column</td>
</tr>
<tr>
<td>PRECISION</td>
<td>int</td>
<td>Maximum precision for the datatype in the target database; NULL if precision</td>
</tr>
<tr>
<td></td>
<td></td>
<td>is not applicable</td>
</tr>
<tr>
<td>LENGTH</td>
<td>int</td>
<td>Length of the column in bytes</td>
</tr>
<tr>
<td>SCALE</td>
<td>smallint</td>
<td>Number of digits to the right of the decimal point; NULL if scale is not</td>
</tr>
<tr>
<td></td>
<td></td>
<td>applicable</td>
</tr>
<tr>
<td>PSEUDO_COLUMN</td>
<td>smallint</td>
<td>Indicates whether the column is a pseudo-column; the access service always</td>
</tr>
<tr>
<td></td>
<td></td>
<td>returns 0 SQL_PC_UNKNOWN</td>
</tr>
</tbody>
</table>
sp_sproc_columns

Description
Returns descriptive information for the input and return parameters for stored procedures in the current environment.

Syntax
sp_sproc_columns sp_name [, sp_owner] [, sp_qualifier] [, column_name]

Parameters
sp_name
is the name of the stored procedure. Use the wildcard character to request information about more than one stored procedure.

sp_owner
is the owner of the stored procedure. Use the wildcard character to request information about stored procedures owned by more than one user. If you do not specify this parameter, sp_sproc_columns looks first for a procedure owned by the current user and then for a procedure owned by the database owner.

sp_qualifier
is ignored. Leave blank or set to NULL.

column_name
is the set of columns to be included in the result set. Use the wildcard character to request information about more than one column. If you do not supply a column_name parameter, sp_sproc_columns returns information about all columns for the stored procedure.

Usage
• The access service selects information from the SYSPROCCOLUMNS table. The cspdb2.sql script creates this table during installation of DirectConnect. However, you need to update the SYSPROCCOLUMNS table manually.

• This function corresponds to the ODBC function SQLProcedureColumns.

Results
sp_sproc_columns returns a list of available procedures. Results are ordered by the following columns:

• PROCEDURE_OWNER
• PROCEDURE_NAME
• COLUMN_TYPE

The lengths for varchar columns shown in the result set tables are maximums; the actual lengths depend on the target database.

The following table shows the result set for sp_sproc_columns.
### Table 4-15: Result set for sp_sproc_columns

<table>
<thead>
<tr>
<th>Column</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROCEDURE_QUALIFIER</td>
<td>varchar 128</td>
<td>Always NULL</td>
</tr>
<tr>
<td>PROCEDURE_OWNER</td>
<td>varchar 128</td>
<td>Value from the corresponding column of SYSPROCCOLUMNS table</td>
</tr>
<tr>
<td>PROCEDURE_NAME</td>
<td>varchar 128</td>
<td>Name of the stored procedure</td>
</tr>
<tr>
<td>COLUMN_NAME</td>
<td>varchar 128</td>
<td>Name of the input parameter or result set column</td>
</tr>
<tr>
<td>COLUMN_TYPE</td>
<td>smallint</td>
<td>Type of data in this procedure column:</td>
</tr>
<tr>
<td></td>
<td>NOT NULL</td>
<td>• 1 SQL_PARAM_INPUT – the procedure column is an input parameter</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 3 SQL_RESULT_COL – the procedure column is a result set column</td>
</tr>
<tr>
<td>DATA_TYPE</td>
<td>smallint</td>
<td>Integer code for the ODBC SQL datatype equivalent of the target database</td>
</tr>
<tr>
<td></td>
<td>NOT NULL</td>
<td>Equivalent for this procedure column</td>
</tr>
<tr>
<td>TYPE_NAME</td>
<td>varchar 128</td>
<td>String representing the datatype name in the target database</td>
</tr>
<tr>
<td>PRECISION</td>
<td>int</td>
<td>Precision of the procedure column on the target database; NULL if precision is not applicable</td>
</tr>
<tr>
<td>LENGTH</td>
<td>int</td>
<td>Length of the column in bytes</td>
</tr>
<tr>
<td>SCALE</td>
<td>smallint</td>
<td>Number of digits to the right of the decimal point; NULL if scale is not applicable</td>
</tr>
<tr>
<td>RADIX</td>
<td>smallint</td>
<td>Base for numeric types; NULL if radix is not applicable</td>
</tr>
<tr>
<td>NULLABLE</td>
<td>smallint</td>
<td>Indicates whether the procedure column accepts NULL values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 0 – the column does not accept NULL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 1 – the column accepts NULL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 2 – it is not known if the column accepts NULL values</td>
</tr>
<tr>
<td>REMARKS</td>
<td>varchar 254</td>
<td>Description of the procedure column</td>
</tr>
</tbody>
</table>
sp_statistics

Description
Returns statistics information for a single table and the indexes associated with
that table.

Syntax
sp_statistics table_name [, table_owner]
[, table_qualifier] [, index_name] [, is_unique]

Parameters

- table_name
  is name of the table. Views, aliases, and wildcard-character search patterns
  are not supported.

- table_owner
  is the owner of the database object about which column privilege
  information is requested. Wildcard-character search patterns are not
  supported. If you do not specify this parameter, sp_statistics looks first for a
  table owned by the current user and then for a table owned by the database
  owner.

- table_qualifier
  is ignored. Leave blank or set to NULL.

- index_name
  is the name of the index. Wildcard-character search patterns are not
  supported.

- is_unique
  is one of the following values:
  “Y” if unique indexes are to be returned
  “N” if unique indexes are not to be returned

Usage

- If index_name is specified, sp_statistics returns only information about
  that index.

- This function corresponds to the ODBC function SQLStatistics.

Results

sp_statistics returns information about the named table. Results are ordered by
the following columns:

- NON_UNIQUE
- TYPE
- INDEX_QUALIFIER
- INDEX_NAME
- SEQ_IN_INDEX
The lengths for `varchar` columns shown in the result set tables are maximums; the actual lengths depend on the target database.

The following table shows the result set for `sp_statistics`.

### Table 4-16: Result set for `sp_statistics`  

<table>
<thead>
<tr>
<th>Column</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>TABLE_QUALIFIER</code></td>
<td><code>varchar(128)</code></td>
<td>Always NULL</td>
</tr>
<tr>
<td><code>TABLE_OWNER</code></td>
<td><code>varchar(128)</code></td>
<td>Table owner authorization ID</td>
</tr>
<tr>
<td><code>TABLE_NAME</code></td>
<td><code>varchar(128)</code></td>
<td>Name of the table or view</td>
</tr>
<tr>
<td><code>NON_UNIQUE</code></td>
<td><code>smallint</code></td>
<td>Indicates whether the index permits duplicate values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 0 (FALSE) means the index prohibits duplicate values</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 1 (TRUE) means the index allows duplicate values</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• NULL is returned if TYPE is SQL_TABLE_STAT</td>
</tr>
<tr>
<td><code>INDEX_QUALIFIER</code></td>
<td><code>varchar(128)</code></td>
<td>Always NULL</td>
</tr>
<tr>
<td><code>INDEX_NAME</code></td>
<td><code>varchar(128)</code></td>
<td>Index name; NULL is returned if TYPE is SQL_TABLE_STAT</td>
</tr>
<tr>
<td><code>TYPE</code></td>
<td><code>smallint</code></td>
<td>Type of information returned:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 0 SQL_TABLE_STAT – statistics for a table</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 1 SQL_INDEX_CLUSTERED – a clustered index</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 2 SQL_INDEX_HASHED – a hashed index</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 3 SQL_INDEX_OTHER – another type of index</td>
</tr>
<tr>
<td><code>SEQ_IN_INDEX</code></td>
<td><code>smallint</code></td>
<td>Sequence of the column in the index (the first column is 1); NULL is</td>
</tr>
<tr>
<td></td>
<td></td>
<td>returned if TYPE is SQL_TABLE_STAT</td>
</tr>
<tr>
<td><code>COLUMN_NAME</code></td>
<td><code>varchar(128)</code></td>
<td>Column name; NULL is returned if TYPE is SQL_TABLE_STAT.</td>
</tr>
<tr>
<td><code>COLLATION</code></td>
<td><code>char(1)</code></td>
<td>Sort sequence for the column:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• A – ascending</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• D – descending</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• NULL – returned if TYPE is SQL_TABLE_STAT</td>
</tr>
<tr>
<td><code>CARDINALITY</code></td>
<td><code>int</code></td>
<td>Cardinality of the table or index:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Number of rows in the table if TYPE is SQL_TABLE_STAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Number of unique values in the index if TYPE is not SQL_TABLE_STAT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• NULL if the value is not available from the target database</td>
</tr>
</tbody>
</table>
sp_stored_procedures

Description
Returns a list of available procedures.

Syntax
sp_stored_procedures [sp_name] [, sp_owner]
[ , sp_qualifier]

Parameters
sp_name
is the stored procedure name. Use the wildcard character to request
information about more than one stored procedure. If left blank,
sp_stored_procedures returns information for all procedures.

sp_owner
is the owner of the stored procedure. Use the wildcard character to request
information about procedures owned by more than one user.

sp_qualifier
is ignored. Leave blank or set to NULL.

Usage
• This function corresponds to the ODBC function SQLProcedures.

Results
sp_stored_procedures lists and describes stored procedures. Results are
ordered by the following columns:

• PROCEDURE_QUALIFIER
• PROCEDURE_OWNER
• PROCEDURE_NAME
sp_table_privileges

The lengths for varchar columns shown in the result set tables are maximums; the actual lengths depend on the target database.

The following table describes the result set for sp_stored_procedures.

<table>
<thead>
<tr>
<th>Column name</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROCEDURE_QUALIFIER</td>
<td>varchar(128)</td>
<td>Always NULL</td>
</tr>
<tr>
<td>PROCEDURE_OWNER</td>
<td>varchar(128)</td>
<td>Procedure owner</td>
</tr>
<tr>
<td>PROCEDURE_NAME</td>
<td>varchar(128)</td>
<td>Procedure name</td>
</tr>
<tr>
<td>NUM_INPUT_PARAMS</td>
<td>int</td>
<td>Number of input parameters in the stored procedure</td>
</tr>
<tr>
<td>NUM_OUTPUT_PARAMS</td>
<td>int</td>
<td>Number of return parameters in the stored procedure</td>
</tr>
<tr>
<td>NUM_RESULT_SETS</td>
<td>int</td>
<td>Number of result sets returned by the stored procedure</td>
</tr>
<tr>
<td>REMARKS</td>
<td>varchar(254)</td>
<td>Describes the procedure</td>
</tr>
<tr>
<td>PROCEDURE_TYPE</td>
<td>smallint</td>
<td>Defines the procedure type:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 SQL_PT_UNKNOWN – it cannot be determined whether the procedure returns a value</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 SQL_PT_PROCEDURE – the returned object is a procedure; it does not have a return value</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 SQL_PT_FUNCTION – the returned object is a function; it has a return value</td>
</tr>
</tbody>
</table>

sp_table_privileges

Description
Returns privilege information for one or more database objects.

Syntax
sp_table_privileges table_name [ , table_owner] [ , table_qualifier]

Parameters
- table_name
  is the name of the table. Use the wildcard character to request information about more than one table. Aliases are not supported.
table_owner

is the owner of the database object about which column privilege information is requested. Use the wildcard character to request information about tables owned by more than one user. If you do not specify this parameter, sp_table_privileges looks first for a table owned by the current user and then for a table owned by the database owner.

table_qualifier

is ignored. Leave blank or set to NULL.

Usage

- The access service selects information from the SYSTABAUTH system catalog table.
- This function corresponds to the ODBC function SQLTablePrivileges.

Results

sp_table_privileges returns a list of one or more database objects with privilege information about each. Results are ordered by the following columns:

- TABLE_OWNER
- TABLE_NAME
- PRIVILEGE

The lengths for varchar columns shown in the result set tables are maximums; the actual lengths depend on the target database.

The following table shows the result set for sp_table_privileges.

<table>
<thead>
<tr>
<th>Table 4-18: Result set for sp_table_privileges</th>
</tr>
</thead>
<tbody>
<tr>
<td>Column name</td>
</tr>
<tr>
<td>------------</td>
</tr>
<tr>
<td>TABLE_QUALIFIER</td>
</tr>
<tr>
<td>TABLE_OWNER</td>
</tr>
<tr>
<td>TABLE_NAME</td>
</tr>
<tr>
<td>GRANTOR</td>
</tr>
<tr>
<td>GRANTEE</td>
</tr>
</tbody>
</table>
sp_tables

<table>
<thead>
<tr>
<th>Column name</th>
<th>Datatype</th>
<th>Notes</th>
</tr>
</thead>
</table>
| PRIVILEGE   | varchar (128) NOT NULL | Identifies the privilege granted to the grantee on this object as one of the following values:  
• SELECT – the grantee is authorized to select rows in the associated object.  
• INSERT – the grantee is authorized to insert rows into the associated object.  
• UPDATE – the grantee is authorized to update rows in the associated object.  
• REFERENCES – the grantee is authorized to refer to one or more columns of the table within a constraint (for example: unique, referential, or table check constraint). |
| IS_GRANTABLE | varchar (3) | Indicates whether the grantee is authorized to grant privilege on this object to others users with one of the following values:  
• YES – the grantee can grant this privilege to others.  
• NO – the grantee cannot grant this privilege to others.  
• NULL – it is unknown or not applicable to the target database. |

sp_tables

Description
Returns a list of objects stored in the database.

Syntax
sp_tables [table_name] [ table_owner] [ table_qualifier] [ table_type]

Parameters

- **table_name**
  is the name of the table. Use the wildcard character to request information about more than one table.

- **table_owner**
  is the owner of the table. Use the wildcard character to request information about tables owned by more than one user.

- **table_qualifier**
  is ignored. Leave empty or set to NULL.
Chapter 4  Accessing Catalog Information with CSPs

**table_type**

is a list of values, separated by commas, requesting information about all objects of a specific type(s) as follows:

```
"'TABLE', 'SYSTEM TABLE', 'VIEW', 'ALIAS', 'SYNONYM'"
```

**Note** You must enclose each table type with single quotation marks, and enclose the entire parameter with double quotation marks. Enter table types in uppercase.

**Usage**

- This function corresponds to the ODBC function SQLTables.

**Results**

`sp_tables` returns a list of database objects. Results are ordered by the following columns:

- **TABLE_TYPE**
- **TABLE_OWNER**
- **TABLE_NAME**

The lengths for varchar columns shown in the result set tables are maximums; the actual lengths depend on the target database.

The following table shows the result set for `sp_tables`.

<table>
<thead>
<tr>
<th>Column</th>
<th>Datatype</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TABLE_QUALIFIER</td>
<td>varchar(128)</td>
<td>Always NULL</td>
</tr>
<tr>
<td>TABLE_OWNER</td>
<td>varchar(128)</td>
<td>Table owner</td>
</tr>
<tr>
<td>TABLE_NAME</td>
<td>varchar(128)</td>
<td>Name of the object about which information is returned</td>
</tr>
<tr>
<td>TABLE_TYPE</td>
<td>varchar(128)</td>
<td>One of the following:</td>
</tr>
<tr>
<td></td>
<td>NOT NULL</td>
<td>- ‘ALIAS’</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- ‘SYNONYM’</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- ‘SYSTEM TABLE’</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- ‘TABLE’</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- ‘VIEW’</td>
</tr>
<tr>
<td>REMARKS</td>
<td>varchar(254)</td>
<td>A description of the table or NULL</td>
</tr>
</tbody>
</table>
### sp_thread_props

**Description**
Enables the client to retrieve and set various thread properties.

**Syntax**
```plaintext
sp_thread_props [ property_name [ , property_value ]]
```

**Parameters**
- `property_name` is the name of the property to be set or shown.
- `property_value` is the value to which the property is to be set.

**Usage**
If you do not provide any parameters, or if you provide only `property_name`, the access service returns a single result set consisting of every instance of `property_name` and the value for each.
CHAPTER 5

Configuring a TRS Library for Security

This chapter explains how to configure TRS to control client access to the following:

- TRS
- Specific host connections
- Mainframe transactions

**Note** If you do not enforce security at TRS (that is, if you set the TRS Security configuration property to no), there are still topics in this chapter that may be helpful. For example, if you defined RPCs to send a user ID and password to your mainframe security, the mainframe must recognize the user ID and password, even if you set the Security property to no.

This chapter contains the following topics:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Security overview</td>
<td>93</td>
</tr>
<tr>
<td>Security quick-start</td>
<td>96</td>
</tr>
<tr>
<td>TRS Administrator’s security tasks</td>
<td>98</td>
</tr>
<tr>
<td>User-level security</td>
<td>100</td>
</tr>
<tr>
<td>Changing user passwords and logins</td>
<td>102</td>
</tr>
<tr>
<td>Conversation-level security</td>
<td>104</td>
</tr>
<tr>
<td>Connection-level security (LU 6.2 only)</td>
<td>105</td>
</tr>
<tr>
<td>Transaction-level security</td>
<td>108</td>
</tr>
</tbody>
</table>

**Security overview**

TRS provides client access security, identifies security considerations and responsibilities, and uses existing security procedures to enforce security.
Security overview

Security features
You can restrict client access to a mainframe processing environment in the following ways:

- Require client identification for access to TRS. Only clients specifically defined to a TRS are allowed to send requests through TRS.
- *(For LU 6.2 only)* Restrict access to mainframe connections. Each client login is assigned a group of connections it can use.
- Restrict access to mainframe transactions. Each client login is assigned a group of permitted mainframe transactions.

By default, TRS security is automatically enabled when you start TRS. You can specifically override it by setting the TRS Security configuration property to *no*.

Security considerations
When you plan security, you must consider security requirements at each of the following network nodes: client, TRS, and mainframe. Your security plan for TRS must address the following issues:

- **Client permissions.** Does the client have permission to log into the network? Can the client’s login information be stored and passed along to TRS for permission checking at that level? Can it be passed to the mainframe to support security systems in use there?

- **Adaptive Server permissions.** If client requests are routed to TRS through Adaptive Server, which commands, data objects, stored procedures, and views does the client have permission to use? Will the client use long running transactions? (Long-running transactions can be sent through ASE/CIS.)

- *(LU 6.2 only)* **Mainframe connection permission.** Does the client have permission to use a given LU 6.2 connection to the mainframe?

- **Mainframe transaction permission.** Does the client have permission to execute a given mainframe transaction?

- **Mainframe data resource permission.** Does the client have permission to access or modify the data in a particular file or database?
For information on client login and Adaptive Server security, see the Adaptive Server and Open Client and Open Server documentation. Sybase security at the mainframe is described in the Open ServerConnect Installation and Administration Guide for IBM CICS/MVS and MainframeConnect Installation and Administration Guide for DB2 UDB. You can find additional information about mainframe security in your vendor documentation.

Security responsibilities

Each instance of a TRS LU62 or TRS TCP/IP Library has its own responsibilities for security. The following section discusses security responsibilities peripheral to TRS.

Client workstation

Most sites on the network have a secure login procedure that verifies the user’s identity and authorization by requiring a unique user ID and password. The user ID, password, and profile information can be passed to Adaptive Server and to TRS.

Adaptive Server

Adaptive Server can grant or deny a user permission to call a particular remote procedure. Requests routed to TRS through Adaptive Server undergo security checks. The TRS administrator can apply this security mechanism to all TRS requests by setting the TRS DirectPrevent configuration property to yes, which requires all client requests to pass through Adaptive Server before they are routed to TRS.

There are two ways to get to TRS from ASE:

- ASE/CIS
- ASE site handler

Network level

The vendor’s SNA support software allows login information to be sent to the mainframe in FMH-5 fields along with client requests. This facility allows you to use external security products that require client login information.
Security quick-start

TCP/IP sends login information to the CICS Listener Transaction when the CICS transaction starts.

Security quick-start

Here are brief, step-by-step instructions for setting up security for TRS. This section assumes that mainframe security is already configured to match the values you will specify as you go through these steps. See the complete description of each procedure that follows in this chapter for details.

1. Set the TRS Security configuration property to yes.
2. Start TRS.
3. Assign a password to the "sa" account.
   (See “Changing user passwords and logins” on page 102.)
   ```
   exec sgw_chpwd sa, password
   ```
   **Note** Remember this password. If you forget passwords for all TRS logins with administration privileges, you will have to reconfigure all of TRS security.

4. *(LU 6.2 only)* Use the following `sgw_addcon` procedure to define the connections your TRS uses. Specify LUs that use a mode entry that supports conversation level security. Talk to your VTAM system programmer and verify the `PSERVIC` property has a value of “x’12’” or “x’10’” in the tenth byte.
   ```
   exec sgw_addcon con_name, region, mode, "max_sessions"
   ```
   See “Adding a connection configuration” on page 46.

5. *(LU 6.2 only)* Use the following `sgw_addcongrp` procedure to add a connection group.
   ```
   exec sgw_addcongrp group_name
   ```
   See “Adding a connection group” on page 106.

6. For LU 6.2 or TCP/IP do the following:
   - *(LU 6.2 only)* Use the `sgw_addcongrp` procedure to add connections to the connection group.
exec sgw_addcontogrp group_name, con_name
See “Adding connections to a connection group” on page 107.

- (TCP/IP only) Use the sgw_addregion procedure to specify the regions.
  
  exec sgw_addregion region, hostname, "port_number"

  See “Defining regions to TRS” on page 48.

7 Use the sgw_addrpc procedure to add RPCs. Use one of the following security parameters to specify the login information to send for each RPC:
  
  exec sgw_addrpc rpc_name, tran_id, region, security

In the sgw_addrpc procedure, the security parameter can have any of the following values to specify the information to send:

- none – do not send login information to the mainframe.
- userid – send only the user ID to the mainframe.
- both – send both the user ID and the password to the mainframe. (Use values that your mainframe security recognizes.)

See “Adding an RPC” on page 50.

8 Use the sgw_addtrngrp procedure to add a transaction group:
  
  exec sgw_addtrngrp tran_group, GROUP_LOGIN, GROUP_PWD, langrpc, langpwdlevel

See “Adding a transaction group” on page 111.

**Note** Be sure that the values of GROUP_LOGIN and GROUP_PWD are in uppercase.

9 Use the sgw_addrpctogrp procedure to add RPCs to the transaction group. For each RPC you add to the group, specify the source of the mainframe login using one of the following rpcpwdlevel parameters:

- none – do not send login information to the mainframe.
- user – send the host login and password specified in the sgw_addlog procedure (see the next step) to the mainframe.
TRS Administrator’s security tasks

Under TRS security, every client login must be defined to TRS. This login definition specifies the client login ID and password, as well as an optional mainframe login ID and password for each. A login definition also includes an assignment to a connection group (LU 6.2 only) and mainframe transaction group. Clients using that login can only access connections and transactions in their assigned groups.

A transaction group lists RPCs that are defined to TRS. Each RPC in the group corresponds to a specific mainframe transaction. When a client calls a remote procedure, the corresponding mainframe transaction executes.

The TRS Administrator’s basic responsibilities are outlined in “Security quick-start” on page 96, which is an overview of steps to set up TRS security.
CHAPTER 5  Configuring a TRS Library for Security

Overriding security

If you do not want to enforce security at TRS, you can disable TRS security by setting the TRS Security configuration property to no. This option tells TRS not to verify logins (except for “sa”) or access to verify to transactions and connections.

When you set the Security property to no, user IDs and passwords used to log in to TRS are forwarded transparently to the mainframe on each RPC. This method uses mainframe security only. See “Adding an RPC” on page 50 for information about RPC security definitions.

User IDs

When you enforce security at TRS, you can choose to assign a single mainframe ID to all clients that use a certain transaction or group of transactions rather than have all individual user IDs and passwords defined. This group ID is specified as part of the transaction group definition with the sgw_addtrngrp procedure. See “Adding a transaction group” on page 111 for more information.

System Administrator’s account

When first installed, TRS has a single client login defined as “sa” (system administrator). This login has permission to use all control and security features of TRS. Initially, a password is not required to log in as “sa.” You should define your own password for the “sa” login as soon as you begin setting up TRS.

Use this procedure to change the password:

```
exec sgw_chpwd login, gateway_pwd, HOST_PWD
```

- Replace login with “sa,” and gateway_pwd with the password for TRS.
- You can omit the HOST_PWD parameter unless you defined the “sa” account at the mainframe as well.
- You do not need to include the comma as a placeholder, because it is the last parameter in the procedure.
- If you include a password for the transaction processing region at the mainframe (host), enter it in uppercase.
User-level security

For more information, see “Changing user passwords and logins” on page 102.

**Note** Remember the password of the TRS “sa.” If you forget the passwords for all TRS logins with administrator privileges, you will have to reconfigure security.

Defining logins to TRS

When TRS security is enabled, a login definition must be defined for every client that wants to access TRS. This definition includes the login ID and password and groups of transactions and connections (LU 6.2 only) that are available to clients using this login.

When you define a login to TRS, you can specify a mainframe ID and password for that login. This feature enables a TRS client attempting access to mainframe resources to use IDs and passwords that the mainframe recognizes.

If security is enforced at TRS, when TRS receives a client request, it checks the client’s login ID and password against its list of login definitions. If the client’s login information matches a login definition entry, TRS accepts the login request. If it does not recognize the login information, it rejects the request.

Only clients with IDs defined to TRS are allowed to login to TRS.

See “Adding a login” on page 101 for more information about defining a login.

User-level security

When security is enforced at TRS (the Security configuration property is set to yes), every user who sends requests to a transaction processing region through TRS must be defined to that TRS.

A user definition includes the following information:

- The user’s login ID and password
- The transaction processing region (host) login ID and password in uppercase
- *(LU 6.2 only)* The assigned connection group that the user is permitted to use to access a mainframe
The assigned transaction group defining the collection of RPCs the user is permitted to use

The permission to perform TRS control operations

Displaying current logins

To display a summary of all existing logins, use this procedure:

```
exec sgw_dsplog
```

The sgw_dsplog procedure displays the login and host login name, the transaction group name, the connection group name (LU 6.2 only), and indicates whether the login can access the control procedures. All users can execute the status procedures.

Adding a login

To add a login definition to TRS, use this procedure:

```
exec sgw_addlog login, pwd, HOST_LOGIN, HOST_PWD, tran_group, con_group, gwctrl
```

where:

- `login` is the login ID of the user, sent from the client application. For example, this would be the value provided in the -U flag specified in `isql`. Length: maximum of 30 characters.
- `pwd` is the login password.
- `HOST_LOGIN` is the login ID by which this user is known to the mainframe. Leave this field blank only if you are also not specifying a `HOST_PWD`. The value for this field must be in uppercase. Length: maximum of eight characters.
- `HOST_PWD` is the password for the `HOST_LOGIN`. The value for this field must be in uppercase. Leave this field blank only if you are also not specifying a `HOST_LOGIN`. Length: maximum of eight characters.
- `tran_group` is the name of the collection of RPCs this user can access. This collection must be defined to TRS, and a user can be assigned to only one transaction group (see “Adding a connection group” on page 106). Length: maximum of eight characters.
Changing user passwords and logins

- **con_group** (LU 6.2 only) is the name of the collection of connections this user can access. This connection group must be defined to TRS, and a user can be assigned only one connection group (see “Adding a connection group” on page 106).
  
  For TCP/IP only, include a comma or null as a placeholder, but do not provide a value for the con_group parameter.
  
  Length: maximum of eight characters.

- **gwctrl** is the TRS administration procedures permission indicator. Choose one of the following values:
  
  - **yes** grants the user permission to access and make changes using control, configuration, and security procedures.
  
  - **no** means the user has status-querying permission only.

  **Note** If you type something other than yes or no, the gwctrl parameter defaults to no.

Example

To add the user named BERTHA to an LU 6.2 TRS, use the sgw_addlog procedure:

```
exec sgw_addlog bertha, BIGBLUE, BIG, BLEUBRT, TGROUP1, FINANCE, yes

```

This isql example adds TRS user named BERTHA with a password of BIGBLUE, and a host login and password of BIG and BLEUBRT, respectively. BERTHA can use RPCs defined to the transaction group named TGROUP1, and connections included in the connection group named FINANCE. BERTHA has permission to administer TRS.

Changing user passwords and logins

Users can change their own passwords. Users with control authority can change other users’ passwords. (Control authority is defined by a yes value for the gwctrl parameter of the sgw_addlog procedure.)
Changing passwords

To change the TRS password or the TRS record of this user's password for a login, use this procedure:

```
exec sgw_chpwd login, pwd, HOST_PWD
```

where:

- `login` is the name of the TRS login for which you intend to change the password.
- `pwd` is the password for TRS.
- `HOST_PWD` is the password for the mainframe. The value for this parameter must be in uppercase.

**Note** If you do not have a value for a parameter, (that is, if you only want to change one password) include the comma or null as a placeholder.

**Example**

To change the mainframe password for a user named BERTHA and keep the same TRS password, use this procedure:

```
exec sgw_chpwd BERTHA, null, BLUEBRT
```

The TRS password BERTHA is unchanged, and her new mainframe password is BLUEBRT.

Changing logins

To change a user's login ID for TRS or for the mainframe (the `HOST_LOGIN` parameter of the `sgw_addlog` procedure), drop the login and add it again with the new ID.

For information about dropping a login, see “Deleting a user definition” on page 103.

Deleting a user definition

You can remove user definitions from the TRS list of logins. To delete a user from the list, use this procedure:

```
exec sgw_droplog login
```

where: `login` is the TRS login name of the user you intend to drop.
Conversation-level security

Example

Use this procedure to remove the user named BERTHA from the TRS list of logins:

```
exec sgw_droplog BERTHA
go
```

Conversation-level security

For LU 6.2, conversation-level security occurs when TRS passes client login information to the mainframe in the conversation-level security fields of the Function Management Header (FMH)-5 along with the client’s request. The mainframe uses this login information to determine whether the client has permission to use the requested resources. For TCP/IP, TRS sends the user ID and password to the Sybase Listener Transaction when the transaction starts.

When configuring RPCs and TRS security, you need to make decisions about:

- When to pass login information to the mainframe
- What login information to pass to the mainframe

When to forward login information

The mainframe may or may not require a full user ID and password complement for every requested transaction. When defining an RPC to TRS even when security is not enabled, you can specify the level of security information that best matches its mainframe component. Your choices are:

- none – TRS passes the request to the mainframe without any user ID or password.
- userid – TRS passes the user ID to the mainframe along with the request.
- both (user ID and password) – TRS passes the user ID and the password to the mainframe along with the request.
What login information to forward

Because user ID and password requirements at the mainframe can be different from those at the client workstation, you can specify a separate mainframe ID and password in the login definition (these values must be in uppercase). When mainframe values are specified, TRS forwards these mainframe values with the client request. If mainframe values are not specified, TRS does not forward the login information.

You can specify an alternate mainframe ID and password for a transaction group. When you add a transaction to the group, specify whether the login definition ID and password or the transaction group ID and password are passed to the mainframe with requests for that transaction.

Connection-level security (LU 6.2 only)

When connection-level security is enforced at TRS (the Security configuration property is set to yes), a user must have explicit permission to use a particular host connection. You assign a connection group to each user defined to TRS. A connection group is a list of connections that are defined to your SNA support and TRS.

Connection groups

Assigning a connection group to a user gives that user permission to use any connection belonging to that group. A user can belong to only one connection group and can use only the connections in that group. If a user login definition does not have a connection group assigned to it, and that user sends a request when security is enforced at TRS, then TRS rejects that request.

Use the connection group procedures to:

- Define the connections that make up a connection group
- Modify that list by adding or deleting connections
- Query connection groups to determine the connections that belong to them
- Add or delete entire connection groups
Connection-level security (LU 6.2 only)

All connections listed in a connection group must be defined to TRS and to your SNA support. When a connection is defined, you can assign it to any number of connection groups. Likewise, you can assign a connection group to any number of users.

Connection-level security enables you to:

- Dedicate a single specific connection to a particular user. To do this, define a connection group to include a single connection, then assign that connection group to a single user.
- Dedicate a group of connections to a particular user. To do this, define a connection group to include the desired connections, then assign that connection group to a single user.
- Dedicate a group of connections to a specific group of users. To do this, define a connection group to include the desired connections, then assign that connection group to all users in the group.

To add new connection groups to TRS and to modify and delete existing connection groups, use the procedures described in the following sections.

Displaying current connection groups

To display all connection groups currently defined to TRS, use this procedure:

```
exec sgw_dspcongrp
```

Displaying one connection group

To display detail about a particular connection group, use this procedure:

```
exec sgw_dspcongrp group_name
```

where: `group_name` is the name of a connection group you want to display.

Example

```
exec sgw_dspcongrp FINANCE
go
```

This procedure returns a list of the connections in the connection group named FINANCE.

Adding a connection group

To define a new connection group, use this procedure:

```
exec sgw_addcongrp group_name
```
where: \textit{group\_name} is the name of the connection group you intend to add. The connection group name can be a maximum of eight characters.

Example

To add the FINANCE connection group, use this procedure:

\begin{verbatim}
exec sgw_addcongrp FINANCE

go
\end{verbatim}

Add connections to the new group as shown in the next section.

\textbf{Adding connections to a connection group}

After you add the new connection group, specify the connections that belong to it. For each connection you add, use this procedure:

\begin{verbatim}
exec sgw_addcontogrp group\_name, con\_name
\end{verbatim}

where:

\begin{itemize}
  \item \textit{group\_name} is the name of the connection group to which you intend to add a connection.
  \item \textit{con\_name} is the name of the connection you intend to add.
\end{itemize}

Re-execute the \texttt{sgw_addcontogrp} procedure for each connection you want to add to the group.

Example

To add the connection named SYBLU01 to the FINANCE connection group, use this procedure:

\begin{verbatim}
exec sgw_addcontogrp FINANCE, SYBLU01

go
\end{verbatim}

\textbf{Dropping connections from a connection group}

To remove connections from a connection group, use this procedure:

\begin{verbatim}
exec sgw_dropconfromgrp group\_name, con\_name
\end{verbatim}

where:

\begin{itemize}
  \item \textit{group\_name} is the name of the connection group from which you intend to drop a connection.
  \item \textit{con\_name} is the name of the connection you intend to drop.
\end{itemize}

Example

To delete the connection named SYBLU01 from the connection group named FINANCE, use this procedure:

\begin{verbatim}
exec sgw_dropconfromgrp FINANCE, SYBLU01

go
\end{verbatim}
Dropping a connection group

To delete an existing connection group, use this procedure:

exec sgw_dropcongrp group_name

where:

group_name is the name of the connection group you intend to drop.

Transaction-level security

When security is enforced at TRS (the Security configuration property is set to yes), a user must have explicit permission to use a particular RPC. To grant a user access to an RPC, assign a transaction group to the user’s login in the sgw_addlog procedure.

Assigning transaction groups

A transaction group is a collection of RPCs defined to TRS. Assigning a transaction group to a user gives that user permission to invoke a remote procedure, causing the corresponding mainframe transaction to execute. A user can belong to only one transaction group and can execute only the transactions in that group.

If a user request specifies an RPC that is not included in the user’s transaction group, TRS rejects the request and returns an error message to the user.

A transaction group can include any number of RPC names. It can also include one RPC name for which the associated mainframe transaction processes SQL language requests dynamically, called the language RPC. An RPC can exist in many transaction groups.

Defining a default SQL language handler

If you do not enforce security at TRS, the default RPC name for a SQL language handler is SYRT. To define the SYRT RPC to TRS, use the sgw_addrpc procedure. If security is enforced at TRS, a default language RPC name does not exist.
Defining group logins

Each user login has an associated mainframe login user ID and password, which are passed to the transaction processing region along with the client request. You can override this login for certain client requests with a group login that applies to all users who are assigned to the same transaction group. A group login and its password is defined when the transaction group is defined. (See “Adding a transaction group” on page 111.)

Specifying login ID levels

When you add a transaction to a transaction group, you must specify the login ID level passed to the transaction processing region whenever that transaction is requested:

- **user** – the user’s transaction processing region login information.
- **group** – the transaction group login information.
- **none** – no login information.

The transaction group login allows you to use a single transaction processing region login for multiple users (for example, everyone in the Accounts Receivable Department).

Transaction group procedures

The transaction group administration procedures allow you to add, modify, and delete transaction groups. Use these procedures to:

- Define the list of RPCs that belong to a group
- Modify that list by adding and deleting RPCs
- Add or delete entire transaction groups
- Specify a group login for the transaction group
- Specify the login, if any, to pass to the transaction processing region with a request


Transaction-level security

- List the following information about a transaction group:
  - The RPCs that belong to the group
  - The language transaction used by its users
  - The transaction processing region login information this group uses

All RPC names listed in a transaction group must be defined to TRS. They must map to transactions the names of which are defined to the mainframe transaction processing region.

After you define an RPC, you can assign it to any number of transaction groups. Also, you can define a transaction group to any number of users. Each user, however, can be associated with only one transaction group.

Displaying all transaction groups

You can add new transaction groups to TRS and modify and delete them. To display information about existing groups, use one this procedure:

```
exec sgw_dsptrngrp
```

The `sgw_dsptrngrp` procedure, when entered without parameters, displays all transaction groups.

Displaying one transaction group

To display details about a particular transaction group, use this procedure:

```
exec sgw_dsptrngrp tran_group, rpc
```

where:

- `tran_group` is the name of the transaction group you want to display (see the following example).
- `rpc` is a keyword that you enter as a fixed-string, optional parameter to only display the RPCs that are members of that transaction group and the RPC password levels. If you omit `rpc`, the member RPCs are not included in the results.

Example

```
exec sgw_dsptrngrp TGROUP1
```

The results of this procedure list the following information:

- Group name
CHAPTER 5 Configuring a TRS Library for Security

- Group login
- Group password
- Language handler
- Language password source

Or, you can include the optional rpc fixed-string parameter, as shown:

```plaintext
exec sgw_dsptrngrp TGROUP1, rpc go
```

The results of this procedure list only the RPC name and the RPC password source.

Adding a transaction group

To define a new transaction group, use this procedure (replace the italicized parameters as shown):

```plaintext
exec sgw_addtrngrp tran_group, GROUP_LOGIN GROUP_PWD, langrpc, langpwdlevel
```

where:

- `tran_group` is the name of the transaction group.
  Length: maximum of eight characters.

- `GROUP_LOGIN` is the alternate transaction processing region login that member transactions can use. When `langpwdlevel` is set to group, the `GROUP_LOGIN` overrides the `HOST_LOGIN` of the user calling this procedure. This value must be in uppercase. Null is valid.
  Length: maximum of eight characters.

- `GROUP_PWD` is the alternate transaction processing region password that member transactions can use. When `langpwdlevel` is set to group, this password overrides the `HOST_LOGIN` of the user calling this procedure. This value must be in uppercase. Null is valid for TRS LU62 only.
  Length: maximum of eight characters.

- `langrpc` is the RPC name used to process SQL language requests. This is the name assigned to all language requests by users of this transaction group. Null is valid.
  Length: maximum of thirty characters.
*langpwdlevel* is the source of the transaction processing region login information for language RPCs. It indicates whether transaction processing region login ID and password should be passed to the transaction processing region with this transaction request, and if so, whether the user’s HOST_LOGIN or the transaction group’s GROUP_LOGIN information should be used. This parameter can have one of the following values:

- **none** – do not send login information to the transaction processing region.
- **user** – send the user’s HOST_LOGIN and HOST_PWD.
- **group** – send the GROUP_LOGIN and GROUP_PWD defined here.

**Example**

This example creates the transaction group named TGROUP1:

```ini
exec sgw_addtrngrp TGROUP1,,AMD2,user
go
```

This example gives the TGROUP1 transaction group the following characteristics:

- It does not use group logins or passwords.
- It uses the AMD2 language RPC.
- It forwards the HOST_LOGIN and HOST_PWD information of the users assigned to this group (in the *sgw_addlog* procedure) to the transaction processing region.

**Adding RPCs to a transaction group**

After you define a transaction group, you must specify the transactions that belong to it. A transaction group contains one language RPC and any number of standard RPCs.

To add an RPC to the transaction group, use this procedure:

```ini
exec sgw_addrpctogrp tran_group, rpc_name, rpcpwdlevel
```

where:

- **tran_group** is the name of the transaction group to which you want to add an RPC.
  Length: maximum of eight characters.
• **rpc_name** is the name of the RPC you want to add. This is the remote procedure called by the client. Length: maximum of 30 characters.

• **rpcpwdlevel** indicates whether user identification is passed to the transaction processing region with this transaction request and, if user identification is to be passed, indicates the origin of the identification. This parameter can have one of the following values:
  • none – do not send login information to the transaction processing region.
  • user – use the user ID and password from the HOST_LOGIN and HOST_PWD values of the user login definition.
  • group – use the user ID and password from the GROUP_LOGIN and GROUP_PWD values of the transaction group definition.

### Specifying an RPC password level

Specify one of the following IDs to send to the mainframe with the request:

- The group ID for the transaction group, defined using the **sgw_addtrngrp** procedure.
- The client’s mainframe login and password from the client’s login definition (**userid**), defined using the **sgw_addlog** procedure.
- none, which indicates that login information should not be sent to the mainframe with that transaction. In combination with setting the TRS Security configuration property to **yes**, this means authorization checking does not occur.

**Example**

```
exec sgw_addrpctogrp TGROUP1, SYV2, user
```

This isql example adds a standard RPC named SYV2 to the transaction group named TGROUP1. The user’s alternate transaction processing region ID (HOST_LOGIN and HOST_PWD) is sent to the transaction processing region.

### Deleting RPC names from a transaction group

To remove an RPC name from a transaction group, use this procedure:

```
exec sgw_droprpcfromgrp tran_group, rpc_name
```

where:
Transaction-level security

- *tran_group* is the name of the transaction group from which you want to delete the RPC.
- *rpc_name* is the name of the RPC you want to delete.

**Example**

To make sure the RPC named SYV2 is no longer part of the TGROUP1 transaction group, use this procedure:

```
exec sgw_droprpcfromgrp TGROUP1, SYV2
```

**Modifying a transaction group**

To change values in an existing transaction group, use this procedure:

```
exec sgw_modtrngrp tran_group, GROUP_LOGIN, GROUP_PWD, langrpc, langpwdlevel
```

where:

- *tran_group* is the name of the transaction group.
- *GROUP_LOGIN* is the alternate transaction processing region login that member transactions can use. When *langpwdlevel* is set to *group*, the GROUP_LOGIN overrides the HOST_LOGIN of the client calling this procedure. This value must be in uppercase.
- *GROUP_PWD* is the alternate transaction processing region password that member transactions can use. When *langpwdlevel* is set to *group*, this password overrides the HOST_PWD of the client calling this procedure. This value must be in uppercase.
- *langrpc* is the RPC name used to process SQL language requests. This is the name assigned to all language requests by users who use this transaction group.
- *langpwdlevel* is the source of the transaction processing region login information for language RPCs. It indicates whether transaction processing region login ID and password should be passed to the transaction processing region with this transaction request, and if so, whether the user’s HOST_LOGIN or the transaction group’s GROUP_LOGIN information should be used. This parameter can be one of the following values:
  - *none*, which means, do not send login information to the transaction processing region.
  - *user*, which means, send the HOST_LOGIN and HOST_PWD.
  - *group*, which means, send the GROUP_LOGIN and GROUP_PWD defined here.
CHAPTER 5 Configuring a TRS Library for Security

Example

If the TGROUP1 transaction group langpwdlevel is currently set to user, this isql example sets it to group:

```
exec sgw_modtrngrp TGROUP1, JOE, MOE, AMD2, group
go
```

The GROUP_LOGIN and GROUP_PWD are now set to JOE and MOE, respectively. The language RPC remains AMD2, and the langpwdlevel is now group. If langpwdlevel is the only parameter you are changing the value of, you can enter this procedure as follows:

```
exec sgw_modtrngrp TGROUP1,,,group
go
```

The commas serve as placeholders for the unchanged parameters.

Deleting a transaction group

To delete a transaction group from the TRS security system, use this procedure:

```
exec sgw_droptrgnrp tran_group
```

where:

*tran_group* is the name of transaction group you want to delete.

Example

This procedure deletes the transaction group named TGROUP1:

```
exec sgw_droptrgnrp TGROUP1
go
```
CHAPTER 6

Using Password Expiration Management (PEM) with TRS

This chapter describes how to implement and use the IBM Advanced Program-to-Program Communications (APPC) Password Expiration Management (PEM) with DirectConnect (TRS).

Note This chapter applies only to LU 6.2.

This chapter contains the following topics:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>What is PEM?</td>
<td>117</td>
</tr>
<tr>
<td>Implementing PEM functionality for LU 6.2 TRS</td>
<td>119</td>
</tr>
<tr>
<td>Obtaining information about passwords</td>
<td>120</td>
</tr>
<tr>
<td>Changing passwords</td>
<td>121</td>
</tr>
<tr>
<td>Setting up new users</td>
<td>123</td>
</tr>
</tbody>
</table>

What is PEM?

PEM is a password management program that IBM provides with:

- CICS 3.3, through an optional PTF UN90057
- CICS versions 4.1 and later
- z/OS

Sybase provides support for PEM as a feature of TRS for LU 6.2. This feature is not available for TRS connections to the mainframe using TCP/IP.
**What is PEM?**

**PEM server capabilities**

The PEM server provides capabilities for an APPC application to:

- Retrieve information regarding the success or failure of the host logon process
- Validate any supplied user ID and password
- Determine when the host password expires
- Update the host password for a specified user ID

**Starting a host transaction**

When you attempt to start a host transaction from TRS, the request may fail due to a host security violation. An expired password, incorrect password setup, or some other reason can cause the failure.

With PEM disabled, an LU 6.2 user cannot determine the exact cause of this security violation. SNA allows only a single error message to be returned to the error log, regardless of the cause.

With PEM enabled for TRS, if a host security violation occurs, TRS sends an error message to the client informing the user to execute a PEM RPC to obtain more information. The exact message depends on whether the request was made by an individual user ID or a transaction group’s user ID. For example, TRS returns the following error message if a security violation occurs as a result of a request made by an individual user ID:

```
34331, "The requested host transaction could not be started because of a host security violation. Please execute sgw_peminfpwd for more information."
```

**Changing the host password**

LU 6.2 TRS support for PEM also allows you to execute procedure calls to change the host password for either an individual user or for a transaction group at both the mainframe and TRS security levels.

PEM returns the following information in response to any of these procedure calls:

- The current successful host login date and time
CHAPTER 6 Using Password Expiration Management (PEM) with TRS

- The last successful host login date and time
- The date and time the current host password expires (can be null if the password never expires)
- The revoke count (number of unsuccessful host logins since last successful logon)

Note PEM does not display the actual password itself.

The following sections explain how to implement and use PEM functionality as an additional feature of TRS for LU 6.2.

Implementing PEM functionality for LU 6.2 TRS

This section assumes that PEM is already installed and all related host work is complete on the mainframe, as described in your IBM documentation.

CICS SIT table property

You may need to ask your CICS system programmer and the external security manager to change the setting of the CICS SIT table property, ISRDELAY=n. This property defines the intersystem refresh delay, which determines how long users remain signed on to the host when running transactions with the Inter System Communication (ISC) setting. Its setting may affect the ability of users to log in more than once or to run multiple host transactions from TRS within the defined time period. By default, the delay is set to 30 minutes. Sybase recommends setting ISRDELAY=0; for CICS version 4.1, this parameter is USRDLY=0.

To implement TRS support for PEM after you install the TRS software, set the TRS PEMDest configuration property, which specifies the remote LU name (the name of the transaction processing region) in which the PEM server sign-on transaction resides on the host. See “PEMDest” on page 26 for more information.
Obtaining information about passwords

Use one of the following RPCs to obtain information about recent attempts to log onto the host and to determine the expiration date of a host password:

- `sgw_peminfopwd` – retrieves information about an individual user’s host password expiration date and logon attempts.
- `sgw_peminfogrppwd` – retrieves information about a transaction group’s host password expiration date and logon attempts.

The following sections describe syntax and usage notes for each procedure call.

User password information

To obtain information about an individual user’s host password expiration date and recent logon attempts, execute the following RPC:

```bash
exec sgw_peminfopwd [hostuserid, hostpwd]
```

If you include the `hostuserid` and `hostpwd` parameters, TRS passes the specified user ID and password to the PEM server.

If you do not specify any parameters, TRS assumes you are requesting information about the client from which you are making the request. It passes one of the following to the PEM server, depending on whether security is enabled:

- If TRS security is enabled (the `Security` configuration property is set to `yes`), the client’s `HOST_LOGIN` and `HOST_PWD`, as defined by `sgw_addlog` or by a previous `sgw_pemchpwd` procedure call
- If security is not enabled (the `Security` configuration property is set to `no`), the user ID and password that the client used to log onto TRS

Maximum length for the user ID and password is eight characters each.

Group password

To obtain information about a transaction group’s host password expiration date and recent logon attempts, execute the following RPC:

```bash
exec sgw_peminfogrppwd tran_group
```
The `tran_group` parameter is required. It specifies the name of the transaction group for which you want logon and password information. TRS passes to the PEM server the transaction group’s `GROUP_LOGIN` and `GROUP_PWD`, as defined by `sgw_addtrngrp` or by a previous `sgw_pemchgrppwd` procedure call.

**Note** You must have Gateway Control Access permission to execute this procedure call.

### Changing passwords

With PEM enabled, you can change a user’s or group’s host password, using one of the following RPCs:

- `sgw_pemchpwd` to change an individual user’s host password
- `sgw_pemchgrppwd` to change a transaction group’s host password

**Note** You must have Gateway Control Access permission to execute the procedure call for group password changes.

If you successfully change the password, the following message appears on the client:

```
The password for host userid 'username' has been successfully changed.
```

Syntax and usage notes for each procedure call are described in the following sections.

### Changing an individual password

TRS clients can change their own host password by executing the following RPC, where `newpwd` is the new host password for the client:

```
exec sgw_pemchpwd newpwd, newpwd
```

You must be logged on as the user whose password you want to change. Depending on whether security is enabled, TRS passes one of the following to the PEM server:
Changing passwords

- The client’s HOST_LOGIN and HOST_PWD, as defined by sgw_addlog or by a previous sgw_pemchpwd procedure call, if security is enabled (the TRS Security configuration property is set to yes)
- The user ID and password that the client used to log on to TRS, if security is off (the TRS Security configuration property is set to no)

The user ID, current password, and new password must be up to eight characters. You must enter the new password twice, as shown in the preceding syntax example.

This operation updates a user’s host password at the mainframe security system, and, when TRS security is enabled, it also updates the user’s HOST_PWD at the TRS security level.

Only the individual user can change his or her password on the host; the TRS administrator cannot perform this task.

Note When security is not enabled, changing the host password does not change the password under which are currently logged in. When you change your host password, you cannot execute any RPCs until you log out of TRS and log in with the correct password.

Changing a group’s password

To change a transaction group’s host password, you must have Gateway Control Access permission. Execute the following RPC:

```
exec sgw_pemchgrppwd tran_group newpwd, newpwd
```

The tran_group parameter is required. It specifies the name of the transaction group you want to change the password for.

You must enter the new password (newpwd) twice, as shown.

This operation updates the host password of the group user ID at the mainframe security system, as well as the transaction group’s GROUP_PWD at the TRS security level, which was last defined by sgw_addtrngrp or by a previous sgw_pemchgrppwd procedure call.
Setting up new users

You can use PEM procedure calls to access logon information or change a user’s host password only if the user already has a valid host password that is known to the mainframe security system.

**Note** You cannot use the `sgw_pemchpwd` or `sgw_pemchgrppwd` procedure calls to set up the initial host password for a new user.

The TRS administrator coordinates host security setup for new users with the mainframe external security administrator:

- The TRS administrator uses the `sgw_addlog` or `sgw_addtrngrp` procedure to set up an individual or group user ID and initial host password at the TRS security level.
- The mainframe external security administrator implements the assigned user ID and host password at the mainframe security level.

After initial setup is complete, the new user should logon to the system and change the administrator-assigned password to a private one using the `sgw_pemchpwd` procedure call.

For more information about setting up new users, see “Adding a login” about using `sgw_addlog`.

For more information about setting up new transaction groups, see “Adding a transaction group” in Chapter 4 about using `sgw_addtrngrp`. 
CHAPTER 7

Controlling a TRS

This chapter explains the controlling administration tasks a TRS may require while it is running.

This chapter contains the following topics:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Controlling connections (LU 6.2 only)</td>
<td>125</td>
</tr>
<tr>
<td>Controlling regions (TCP/IP Only)</td>
<td>128</td>
</tr>
<tr>
<td>Disconnecting a client</td>
<td>129</td>
</tr>
<tr>
<td>Controlling RPCs</td>
<td>129</td>
</tr>
<tr>
<td>Controlling tracing</td>
<td>130</td>
</tr>
<tr>
<td>Controlling accounting</td>
<td>132</td>
</tr>
<tr>
<td>Stopping TRS</td>
<td>133</td>
</tr>
</tbody>
</table>

If you are enforcing security at TRS, the procedures described in this chapter require administration permissions. See “Adding a login” on page 101 for information about TRS administration permissions.

Controlling connections (LU 6.2 only)

If you are using LU 6.2, this section describes how to start a single connection and all connections, how to prevent inactive connections, and how to stop a connection gradually and abruptly.

Activating connections

To reactivate a connection, use one of the following procedures. You can activate a single connection or restart all inactive connections.
By default, connections are active as a result of defining them. Connections may require reactivation if they have been made inactive, either as a result of use of the `sgw_deactcon` RPC or a problem on the SNA network while the `sgw_deactcon` property is set to `yes`.

### Activating a single connection

To activate a single connection, use this procedure:

```bash
exec sgw_actcon "con_number"
```

where `"con_number"` is the number of the connection you intend to start. This is the 'connect number' with the value displayed in the `sgw_status` connections procedure. Enclose numeric parameter values in quotation marks.

**Example**

To activate connection number 1, use this procedure:

```bash
exec sgw_actcon "1"
go
```

### Restarting all connections

To restart all connections, use this procedure:

```bash
exec sgw_actcon all
```

where the `all` option activates all connections, allowing you to recover when your SNA support stops or connections become inactive for any reason.

### Marking connections as inactive

To have TRS mark connections as “inactive” if it receives an unrecoverable error when trying to use the connection, set the `DeactCon` configuration parameter to `yes`. When the error that caused the connection to be marked “inactive” is corrected, reactivate the connection.

### Preventing inactive connections

To prevent TRS from marking connections “inactive,” you can set the TRS `DeactCon` configuration property to `no`. We recommend this option for remote sites that run unattended.
### Deactivating a connection

To deactivate a connection, use either of these procedures:

```bash
exec sgw_deactcon "con_number"
```

or

```bash
exec sgw_deactcon "con_number", force
```

where:

- "con_number" is the number of the connection you intend to deactivate. This is the connection number value displayed in the `sgw_status` connection procedure. Enclose numeric parameter values in quotation marks.

- `force` is optional. If you use the `force` option, the connection you specify ends, even if it is currently executing. However, on some TRS platforms, even a forced deactivate allows the current request to complete before deactivating the connection.

If you do not use the `force` option, TRS allows any transactions in progress to complete before it deactivates the connection. While these transactions finish processing, the connection is considered to be "draining."

**Example**

To deactivate connection number “1,” use this procedure:

```bash
exec sgw_deactcon "1", force
```

The `force` option causes connection number “1” to deactivate even if it is currently executing.

### Deactivating LU 6.2 connections

This section describes how to deactivate connections in an LU 6.2 environment before you disconnect clients.

If you need to disconnect a client that is waiting for transaction results, you or your system programmer can use one of the following methods to deactivate the connection before you disconnect the client:

- **Using VTAM:**
  ```bash
  VARY NET, INACT, ID=lu_name, FORCE
  ```

- **Using isql:**
  ```bash
  exec sgw_deactcon "con_number", force
  ```
Controlling regions (TCP/IP Only)

This section describes how to activate a single region or all regions in a TCP/IP environment and how to deactivate a region.

Activating regions

To activate a region, use one of the following procedures. You can restart all inactive regions or just activate a single region.

Restarting all regions

To restart all regions, use this procedure:

```
exec sgw_actregion all
```

where the `all` option activates all regions, allowing you to recover when your TCP/IP support stops or regions become inactive for any reason.

Activating a single region

To activate a single region, use this procedure:

```
exec sgw_actregion region
```

where `region` is the name of the region you intend to activate. This is the name you assigned to the region in the `sgw_addregion` procedure.

Example

To activate the region named `TESTREG`, use this procedure:

```
exec sgw_actregion TESTREG
```

Deactivating a region

Deactivating a region prevents users from using that region.

To deactivate a region, use this procedure:

```
exec sgw_deactregion region
```

where `region` is the name of the region you intend to deactivate. This is the name you assigned to the region in the `sgw_addregion` procedure.
Example

To deactivate the region named TESTREG, use this procedure:

```
exec sgw_deactregion TESTREG
go
```

## Disconnecting a client

You can force a particular client to disconnect. Generally, you use this command when you want to disconnect idle clients or clients having network problems. To disconnect a client, use this procedure:

```
exec sgw_disclient "client_number"
```

where "client_number" is the number of the client you intend to disconnect. Obtain the client number from the sgw_status clients procedure. Enclose numeric parameter values in quotation marks.

If you disconnect a client that invoked a long running transaction before the transaction ends, TRS deallocates the conversation and disconnects the client.

Example

To disconnect client number 7, use this procedure:

```
exec sgw_disclient "7"
go
```

If a transaction is in process, this command disconnects clients that are actively reading and processing results.

## Controlling RPCs

You can take an RPC out of service by declaring it inactive. TRS rejects any client call to an inactive RPC name. A typical reason to deactivate an RPC is that the associated mainframe transaction is temporarily off line.

## Activating an RPC

To make a defined RPC available (activate), use this procedure:

```
exec sgw_actrpc rpc_name
```

where rpc_name is the name of the RPC you intend to activate.
Controlling tracing

Example

To activate the SYV2 RPC, use this procedure:

```
exec sgw_actrpc SYV2
go
```

Deactivating an RPC

To make a defined RPC unavailable (deactivate), use this procedure:

```
exec sgw_deactrpc rpc_name
```

where `rpc_name` is the name of the RPC.

Example

To deactivate the SYV2 RPC, use this procedure:

```
exec sgw_deactrpc SYV2
go
```

Controlling tracing

The TRS tracing facility provides entry/exit tracing, tracing of the TRS interface with the back-end transport protocol, and TDS header and data tracing. When you enable tracing, tracing information is written to a set of error logs.

Ordinarily, you do not need to trace TRS activity. The tracing facility is provided to help Sybase Technical Support assist you if you call about certain errors. Tracing can also be useful for diagnosing local area network (LAN) and client application problems. For more information about tracing, see the sections describing the TRS TraceTRS, TraceProtocol, ProtocolTraceFile, and TDSTraceFile configuration properties in Chapter 2, “Creating a TRS”.

Mainframe-based tracing is described in the Open ServerConnect Programmer’s Reference guides. COBOL and PL/1 versions of this guide are available.

To enable or disable entry/exit tracing, set the servers TraceEntryExit property. TRS entry/exit tracing accesses the following file:

For UNIX:

```
$SYBASE/$SYBASE_ECON/srvname/log/srvname.trc
```

For Windows:
%SYBASE%/%SYBASE_ECON%/srvname/log/srvname.trc

To enable or disable TDS tracing before TRS starts running, set the appropriate properties in the TRS configuration file. See the sections describing the TRS TraceTRS and TDSTraceFile configuration properties in Chapter 2, “Creating a TRS”.

To enable or disable protocol tracing before TRS starts running, set the appropriate properties in the TRS configuration file. See the sections describing the TRS TraceProtocol and ProtocolTraceFile configuration properties in Chapter 2, “Creating a TRS”.

You can also use DirectConnect Manager to enable and disable tracing while TRS runs.

Starting tracing

To start TDS tracing, use the following procedure:

exec sgw_starttrace TDS

where TDS activates TDS tracing. TDS tracing accesses the following directory:

$SYBASE/$SYBASE_ECON/srvname/log

To start protocol tracing, use the following procedure:

exec sgw_starttrace PROT

where PROT activates tracing of the DirectConnect interface with the backend transport protocol layer, for either TCP/IP or LU 6.2.

- On Windows, back-end TCP/IP tracing goes into
  $SYBASE/$SYBASE_ECON/srvname/log/trstcp.ngtcp
- On UNIX, back-end TCP/IP tracing goes into
  $SYBASE/$SYBASE_ECON/srvname/log/ngtcp.trstcp
- On Windows, back-end LU 6.2 tracing goes into
  $SYBASE/$SYBASE_ECON/srvname/log/trslu62.nglu62
- On UNIX, back-end LU 6.2 tracing goes into
  $SYBASE/$SYBASE_ECON/srvname/log/nglu62.trslu62

If no parameter is entered, the default is TDS.
Stopping tracing

To stop TDS tracing, use the following procedure:

```
exec sgw_stoptrace TDS
```

where TDS tracing is disabled.

To stop protocol tracing, use the following procedure:

```
exec sgw_stoptrace PROT
```

where PROT tracing is disabled for either LU 6.2 or TCP/IP.

If no parameter is entered the default is TDS.

Controlling accounting

TRS allows you to record accounting information. This section describes how to record accounting at TRS. Mainframe-based accounting is explained in the Open ServerConnect documentation.

The TRS accounting facility records the following information:

- The name by which TRS is known.
- The RPC the named client calls.
- The connection name or mainframe name relevant to this RPC.
- The date and time that TRS sent the request to the transaction processing region.
- The time elapsed since the request was sent.

The time elapsed count starts when TRS receives the request and continues until the final result row is sent to the client. Some applications, such as Data Workbench, read a few rows at a time, and then request more rows as the user requests them to be displayed, allowing the user to read the results. This time is included in the total duration.

- The total number of bytes sent and received with this RPC.
- The total number of Sybase TDS packets sent and received with this RPC. A packet is 512 bytes or less.
To turn on accounting before TRS starts running, set the appropriate properties in the TRS configuration file. See the sections describing the Accounting and AccountFile configuration properties in Chapter 3, “Configuring a TRS”.

**Note** You can also use DirectConnect Manager to enable and disable accounting while TRS is running.

### Activating and deactivating accounting

To start and stop the TRS accounting facility while TRS is running, use these procedures:

```bash
exec sgw_startact
exec sgw_stopact
```

Executing these procedures is equivalent to setting the Accounting configuration property to `no`.

### Reading the accounting log

When you activate accounting, TRS writes the accounting records to the accounting log. See the AccountFile configuration property for the name of the accounting log file.

To display the accounting log, use this procedure:

```bash
exec sgw_dspact
```

Each accounting log record is returned in a row.

### Stopping TRS

Generally, TRS runs continuously. If you need to deactivate TRS, use the following procedure to disconnect each client and allow conversations in progress to finish first:

```bash
exec sgw_shutdown
```
Use the following procedure to disconnect each client immediately without waiting for conversations in progress to finish:

```bash
exec sgw_shutdown now
```

With the preceding procedure, TRS does not accept any new client requests.

When you are ready to start TRS again, set the configuration properties described in Chapter 3, “Configuring a TRS”.

Stopping TRS
CHAPTER 8

Monitoring a TRS

This chapter explains how to obtain information about TRS users, connections, regions, and RPCs; how to obtain trace status; and how to determine the options specified when TRS starts.

This chapter contains the following topics:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monitoring the status of TRS</td>
<td>135</td>
</tr>
<tr>
<td>Monitoring clients</td>
<td>136</td>
</tr>
<tr>
<td>Monitoring connections (LU 6.2 only)</td>
<td>137</td>
</tr>
<tr>
<td>Monitoring regions (TCP/IP only)</td>
<td>138</td>
</tr>
<tr>
<td>Monitoring RPCs</td>
<td>139</td>
</tr>
<tr>
<td>Displaying TRS configuration properties</td>
<td>140</td>
</tr>
<tr>
<td>Requesting trace information</td>
<td>141</td>
</tr>
</tbody>
</table>

Monitoring the status of TRS

You can use the sgw_status procedure to query the status of TRS. The status procedures tell you the following:

- The clients logged in to TRS
- The system on which they are running
- Remote procedure calls (RPCs) they call
- Connections they use
- Accounting information

The following procedure queries the status of TRS:

```
exec sgw_status options
```

The following are values for `options` in the `sgw_status` procedure, which are described in this chapter:

- clients
Monitoring clients

To query the status of clients, use this procedure:

```
exec sgw_status clients
```

This procedure displays the information in the following table for all active clients.

### Table 8-1: Description of sgw_status clients results

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Login</td>
<td>The login name of the user.</td>
</tr>
<tr>
<td>Client_Number</td>
<td>TRS issues a unique client number each time a user logs in. A user logged in more than once has the same Login and a different Client_Number for each connection to TRS.</td>
</tr>
<tr>
<td>RPC_Name</td>
<td>The RPC called by the client.</td>
</tr>
<tr>
<td></td>
<td>• If the request is a direct call from an Open Client DB-Library application, this field contains the RPC name specified in the dbrpcinit statement.</td>
</tr>
<tr>
<td></td>
<td>• If the request is an indirect call from a Adaptive Server stored procedure, this is the RPC name that the stored procedure used when it called TRS.</td>
</tr>
<tr>
<td>Host_Tran</td>
<td>The name of the mainframe (host) transaction being invoked. This is the mainframe transaction associated with the RPC name. If a transaction is not in progress, this field is blank.</td>
</tr>
<tr>
<td>Client_Machine</td>
<td>The name of the machine on which the client program is running.</td>
</tr>
<tr>
<td>Con_Number</td>
<td>The connection number. TRS uses this number to represent the client’s current SNA connection. If the client is not using any connections, this field is blank. (LU 6.2 only)</td>
</tr>
</tbody>
</table>
You may find an entry in this screen with SH listed under the State field but without an associated Login field. This entry represents the site handler for a remote Adaptive Server.

To display the name of the remote server in the Login field, you add the following configuration command at Adaptive Server, and then restart Adaptive Server:

```
sp_addserver servername, local
```

Replace servername with the name of the remote Adaptive Server.

### Monitoring connections (LU 6.2 only)

To query the status of the connections that TRS uses, use this procedure:

```
exec sgw_status connections
```
This procedure displays the defined connections currently known to TRS and the status of each.

The following table shows the connection information that returns.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Con_Number</td>
<td>This number represents the connection being described.</td>
</tr>
<tr>
<td>Status</td>
<td>The connection availability, which indicates whether the connection is currently available for use. Valid values are:</td>
</tr>
<tr>
<td></td>
<td>• A (active) – the connection is available.</td>
</tr>
<tr>
<td></td>
<td>• I (inactive) – the connection is not available.</td>
</tr>
<tr>
<td></td>
<td>• D (draining) – the connection is not available.</td>
</tr>
<tr>
<td></td>
<td>A connection is considered to be “draining” if you deactivated it while it was in use. It remains in draining status until the request completes, then becomes inactive.</td>
</tr>
<tr>
<td>Connection</td>
<td>The name of the SNA connection as defined in the sgw_addcon procedure.</td>
</tr>
<tr>
<td>Mode</td>
<td>The name of the mode used with this connection as defined in the sgw_addcon procedure.</td>
</tr>
<tr>
<td>DestSys</td>
<td>The name of the transaction processing region accessed by the connection as defined in the sgw_addcon procedure.</td>
</tr>
<tr>
<td>Host_Tran</td>
<td>The name of the mainframe transaction being invoked as defined in the sgw_addrpc procedure. If a transaction is not being invoked, this field value is Null.</td>
</tr>
<tr>
<td>Client_Number</td>
<td>The client currently using this connection. This is the same number used to identify the login of each client in the result of the sgw_status clients command. If the client is not using a connection, this field is blank.</td>
</tr>
</tbody>
</table>

**Monitoring regions (TCP/IP only)**

To query the status of the regions that TRS uses, use this procedure:

```
exec sgw_status region
```

The information displayed is shown in the following table.
Table 8-3: Description of sgw_status region results

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Region</td>
<td>The name of the transaction processing region as specified in the sgw_addregion procedure.</td>
</tr>
<tr>
<td>Host Name</td>
<td>The TCP/IP network host name as specified in the sgw_addregion procedure.</td>
</tr>
<tr>
<td>Port</td>
<td>The number of the port as specified in the sgw_addregion procedure.</td>
</tr>
</tbody>
</table>
| Status | The region availability, which indicates whether the named region is currently available for use. Valid values are:  
| | • A (active) – the region is available for use.  
| | • I (inactive) – the region is unavailable for use. |

Table 8-4: Description of sgw_status rpc results

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPC</td>
<td>The name of the RPC being called.</td>
</tr>
</tbody>
</table>
| Status | The availability of the RPC, which indicates whether the named RPC is currently available for use. Valid values are:  
| | • A (active) – the RPC is available for use.  
| | • I (inactive) – the RPC is unavailable for use. |
| Host_Tran | The name of the mainframe transaction to be invoked. |
| Security_Fields | The mainframe access permission requirements. This field specifies the administration procedure parameter values that TRS passes to the mainframe. Valid values are:  
| | • U (login ID) – the login ID is passed to the mainframe.  
| | • B (both) – both the ID and password are passed to the mainframe.  
| | • N (none) – login information is not passed to the mainframe. |
| Destsys | The name of the transaction processing region with which this RPC is associated as defined in the sgw_addrpc procedure. |
Displaying TRS configuration properties

To display the current property settings in the TRS configuration file, use this procedure:

```
exec sgw_status parameters
```

**Note** See Chapter 3, “Configuring a TRS” for complete information about setting up the TRS configuration file.

This procedure displays the properties shown in the following table.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>The version/release level of the current TRS and the platform and operating system on which it is running.</td>
</tr>
<tr>
<td>Server name</td>
<td>The name of the DirectConnect server.</td>
</tr>
<tr>
<td>Protocol type</td>
<td>The network protocol used by this TRS, either LU 6.2 or TCP/IP.</td>
</tr>
<tr>
<td>National language</td>
<td>The default language for TRS. (This is also set at the mainframe in SYGWMCST, which is the global customization module.)</td>
</tr>
<tr>
<td>Char set</td>
<td>The default character set for TRS. (This is also set at the mainframe in SYGWMCST, which is the global customization module.)</td>
</tr>
<tr>
<td>Direct RPCs disabled</td>
<td>Indicates whether TRS will accept an RPC directly from a client or whether all RPCs must be indirect, that is, routed through Adaptive Server. Valid values are:</td>
</tr>
<tr>
<td></td>
<td>• Yes – indirect routing is required.</td>
</tr>
<tr>
<td></td>
<td>• No – indirect routing is not required; direct routing is permitted.</td>
</tr>
<tr>
<td>Max users</td>
<td>The maximum number of users allowed to use TRS at one time.</td>
</tr>
<tr>
<td>Max site handlers</td>
<td>The maximum number of site handlers allowed. A site handler controls the network connection to a remote server.</td>
</tr>
<tr>
<td>Truncate longvarchar</td>
<td>The truncation flag for MainframeConnect for DB2 UDB. (This is also set at the mainframe in SYGWMCST, which is the global customization module.) This indicates whether the data in fields of the datatype long varchar are to be truncated to 255 bytes and passed to the client. If the truncation flag is not used, varchar data is sent as text and image datatypes for 4.x TDS clients, or as long varchar datatype for 5.0 TDS clients. Valid values are:</td>
</tr>
<tr>
<td></td>
<td>• Yes. The data is truncated.</td>
</tr>
<tr>
<td></td>
<td>• No. Other datatypes are returned.</td>
</tr>
</tbody>
</table>
CHAPTER 8  Monitoring a TRS

Requesting trace information

The TRS tracing facility provides TDS header and data tracing. When you enable tracing, TRS writes tracing information to the trace file name specified in the TDSTraceFile configuration property.

Ordinarily, you do not trace TRS activity. The tracing facility is provided to help Sybase Technical Support understand what occurred if you have to call about specific errors.

To request information about the status of the trace facility, use this procedure:

```
exec sgw_status trace
```

The following table shows the information that the procedure displays.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Security enforced</td>
<td>This indicates whether TRS security is enabled or overridden. Valid values are:</td>
</tr>
<tr>
<td></td>
<td>• Yes. Security is enforced at TRS.</td>
</tr>
<tr>
<td></td>
<td>• No. Security is not enforced at TRS, except for the “sa” account.</td>
</tr>
<tr>
<td></td>
<td>(The RPC security definition is sent to the mainframe, if specified in the RPC definition.)</td>
</tr>
<tr>
<td>Interfaces file</td>
<td>The complete path and filename of the interfaces file for this TRS.</td>
</tr>
<tr>
<td>RPC file</td>
<td>The complete path and filename of the file that contains RPC information for this TRS.</td>
</tr>
<tr>
<td>Security grp file</td>
<td>The complete path and filename of the file that contains security information for this TRS.</td>
</tr>
<tr>
<td>Accounting</td>
<td>Indicates if accounting is activated for this TRS. Valid values are:</td>
</tr>
<tr>
<td></td>
<td>• Yes – accounting is on.</td>
</tr>
<tr>
<td></td>
<td>• No – accounting is off</td>
</tr>
<tr>
<td>Connection file</td>
<td>The complete path name of the file that contains connection information for this TRS.</td>
</tr>
<tr>
<td>Con wait q (secs)</td>
<td>The number of seconds that the connection request waits in the queue for an available LU 6.2 connection.</td>
</tr>
<tr>
<td>Region file (TCP/IP only)</td>
<td>The complete path and file name of the file that contains region information for this TRS.</td>
</tr>
</tbody>
</table>
Summary of clients in each listed state

Table 8-6: Description of sgw_status trace results

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>The version and release level of the current TRS and the platform and operating system on which it is running.</td>
</tr>
<tr>
<td>Trace</td>
<td>The trace indicator, which indicates whether tracing is enabled for this TRS. Valid values are:</td>
</tr>
<tr>
<td></td>
<td>• Active – the trace facility is enabled.</td>
</tr>
<tr>
<td></td>
<td>• Inactive – the trace facility is disabled.</td>
</tr>
<tr>
<td>Logfile</td>
<td>The name of the file for this DirectConnect server where TRS log records are written.</td>
</tr>
<tr>
<td>TDSlog</td>
<td>The name of the file for this TRS that contains trace data between TRS and the mainframe.</td>
</tr>
</tbody>
</table>

Summary of clients in each listed state

To request the summary of all clients and their current state use the following procedure:

    exec sgw_status summary

This command tabulates the number of clients in each state as listed below:

Table 8-7: Summary of clients and their current state

<table>
<thead>
<tr>
<th>State</th>
<th>Count</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>0</td>
<td>Connected; No transactions</td>
</tr>
<tr>
<td>AL</td>
<td>0</td>
<td>Allocating conversation</td>
</tr>
<tr>
<td>WA</td>
<td>0</td>
<td>Waiting for first results</td>
</tr>
<tr>
<td>RS</td>
<td>0</td>
<td>Reading from host</td>
</tr>
<tr>
<td>WC</td>
<td>0</td>
<td>Writing to client</td>
</tr>
<tr>
<td>SH</td>
<td>0</td>
<td>Site handler</td>
</tr>
<tr>
<td>GC</td>
<td>1</td>
<td>Gateway control</td>
</tr>
<tr>
<td>CQ</td>
<td>0</td>
<td>Queued; Awaiting connection</td>
</tr>
<tr>
<td>IT</td>
<td>0</td>
<td>Idle; Transaction active</td>
</tr>
</tbody>
</table>
Sending Requests to TRS

Clients using Sybase mainframe access products (see “Related products” on page 6) can send requests to TRS to access mainframe data. TRS forwards the requests to the mainframe and returns results in the same format as the results that Adaptive Server returns. Communication between TRS and the mainframe is transparent to the client. This appendix describes the types of requests a client can send to TRS.

This appendix contains the following topics:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description of request types</td>
<td>143</td>
</tr>
<tr>
<td>Unsupported calls</td>
<td>146</td>
</tr>
</tbody>
</table>

Description of request types

Clients can send two types of requests to TRS:

- SQL language requests
- Remote procedure calls (RPCs)

Requests can be sent to TRS two ways, directly or indirectly:

- Direct requests are RPCs or SQL language statements that access TRS without an intermediary server.
- Indirect requests invoke an RPC on Adaptive Server, ASE/CIS or Replication Server, which then sends a request to TRS.

If you are using MainframeConnect for DB2 UDB, TRS directs requests to the AMD2 transaction at the mainframe. If you are using ASE/CIS Access Module for DB2 UDB for IMS and MVS, TRS directs requests to the SYRT transaction at the mainframe. (See “Configuring a default SQL language handler for TRS” on page 53.)
Description of request types

Clients can send SQL language requests and RPCs to the AMD2 transaction. TRS handles any request sent indirectly (that is, through Adaptive Server) as an RPC. Long-running transactions cannot be sent through Adaptive Server because Adaptive Server logs out of TRS after each request. This is not true for ASE/CIS.

Size of requests to AMD2

The AMD2 transaction can process language requests up to 32K. RPC parameters submitted to AMD2 must be CHAR parameters. The transaction concatenates multiple RPC parameters into one SQL statement for DB2 UDB. (Include sufficient blanks in each parameter to make a valid statement for DB2 UDB.)

Sending SQL statements to DB2 UDB

When sending SQL language requests to DB2 UDB, the client can send only SQL statements that are understood by DB2 UDB. See the MainframeConnect Installation and Administration Guide for DB2 UDB for information about SQL compatibility.

Accessing DB2 UDB data

Clients can send SQL statements that access DB2 UDB data to TRS directly or indirectly. TRS sends SQL language requests to the AMD2 mainframe transaction, which submits the SQL statements to DB2 using DB2’s dynamic SQL facility. AMD2 performs the requested actions and returns results to TRS, which forwards them to the client.

Sending RPCs to TRS

As described in Chapter 3, “Configuring a TRS”, TRS maps RPCs to mainframe transactions. If your site uses MainframeConnect for DB2 UDB, RPCs are mapped either to AMD2 or to the Catalog RPCs, which retrieve specific catalog information about DB2 UDB system tables.

Clients can use any of the following methods to send an RPC to TRS:
• Using isql or another SQL utility, send the RPC using the execute command.
• Include the RPC in an Open Client application, and send the RPC directly to TRS.
• Send an RPC indirectly. A client can call a stored procedure in Adaptive Server that in turn sends an RPC to TRS. (Do not use this method for long-running transactions unless you are using the functionality of ASE/CIS.)

Sending RPCs directly to TRS

To send RPCs directly to TRS, use the execute command, which is described in this section. See the Sybase Adaptive Server Reference Manual for detailed information about the execute command.

The syntax for the exec command is:

\[\text{exec procedure_name \{[@parameter_name=value] \[[@parameter_name=value...]]\}}\]

where

• procedure_name the name by which the RPC is defined to TRS.
• @parameter_name=value is the value assigned to one of the RPC parameters. Repeat this argument for each of the RPC parameters. The name is optional.

@parameter_name=value allows you to enter the parameters in any order, as long as the mainframe program can recognize parameters by name. If you use this form for any parameter, you must use it for all parameters in the same exec statement.

Sending RPCs indirectly to the mainframe

To send an indirect request to the mainframe, a client application issues an RPC that resides on Adaptive Server.

After parsing and pre-processing the request, Adaptive Server sends the request and parameters to TRS for forwarding to a mainframe transaction. When the results return, they follow the same route in reverse.
Unsupported calls

If you set the TRS DirectPrevent configuration property to yes, TRS rejects all direct calls from client applications, requiring all requests to be sent indirectly. Routing all requests through Adaptive Server allows you to use additional front-end tools and provide additional security checks. You should not use this method if the client submits long-running transactions.

Stored procedures that call TRS

When Adaptive Server calls TRS, it follows the same procedure it does for any other call to a remote server. The stored procedure uses the exec statement to call the remote procedure and forward the parameters.

An example of a stored procedure is shown in Chapter 3, “Configuring a TRS”.

Unsupported calls

This section lists the Sybase Open Client DB-Library and CT-Library calls that are not supported in this release.

DB-Library calls

Routines designed to process results of COMPUTE calls:

<table>
<thead>
<tr>
<th>dbadata</th>
<th>dbaltlen</th>
<th>dbaltbind_ps</th>
</tr>
</thead>
<tbody>
<tr>
<td>dbadlen</td>
<td>dbaltop</td>
<td>dbanullbind</td>
</tr>
<tr>
<td>dbaltbind</td>
<td>dbalttype</td>
<td>dbbylist</td>
</tr>
<tr>
<td>dbaltcolid</td>
<td>dbaltutypex</td>
<td></td>
</tr>
</tbody>
</table>

Browse mode routines:

<table>
<thead>
<tr>
<th>dbequal</th>
<th>dbtput</th>
<th>dbtabcount</th>
</tr>
</thead>
<tbody>
<tr>
<td>dbfreequal</td>
<td>dbcolbrowse</td>
<td>dbtabname</td>
</tr>
<tr>
<td>dbtsnewval</td>
<td>dbcolsourse</td>
<td>dbtabsourse</td>
</tr>
<tr>
<td>dbtsnewlen</td>
<td>dbtabbrowse</td>
<td></td>
</tr>
</tbody>
</table>

Registered procedure routines:
**Network routines:**

- `dbncreate`  
- `dbnpdefine`  
- `dbregdrop`  
- `dbreglist`  
- `dbreghandle`  
- `dbreginit`  
- `dbregnowatch`  
- `dbregparam`  
- `dbregwatch`  
- `dbregwatchlist`  
- `dbregparm`  
- `dbsetnotifs`  

**Bulk copy routines:**

- `bcp_batch`  
- `bcp_bind`  
- `bcp_colfmt`  
- `bcp_colptr`  
- `bcp_collen`  
- `bcp_columns`  
- `bcp_control`  
- `bcp_done`  
- `bcp_exec`  
- `bcp_init`  
- `bcp_moretext`  
- `bcp_sendrow`  
- `BCP_SETL`  

**Two-phase commit routines:**

- `abort_xact`  
- `build_xact_string`  
- `close_commit`  
- `commit_xact`  
- `open_commit`  
- `remove_xact`  
- `scan_xact`  
- `start_xact`  
- `stat_xact`  

**Routines that process options:**

- `dbclopt`  
- `dbisopt`  
- `dbsetopt`  

**Other disallowed routines:**

- `dbchange`  
- `DBCMDROW`  
- `DBCURCMD`  
- `dbchange`  
- `DBMORECMDS`  
- `DBNUMORDERS`  
- `DBOFFSET`  
- `dbgetoff`  
- `dbisopt`  
- `dbsetopt`  
- `dbsetnotifs`  

**Client-Library calls**

Routines designed to process results of `COMPUTE` calls:

- `ct_compute_info`

Browse mode routines:
**Unsupported calls**

<table>
<thead>
<tr>
<th>ct_br_column</th>
<th>ct_br_table</th>
</tr>
</thead>
</table>

**Network routines:**

<table>
<thead>
<tr>
<th>ct_sendpassthru</th>
<th>ct_recvpassthru</th>
</tr>
</thead>
</table>

**Bulk copy routines:**

<table>
<thead>
<tr>
<th>blk_alloc</th>
<th>blk_drop</th>
<th>blk_rowxfer</th>
</tr>
</thead>
<tbody>
<tr>
<td>blk_bound</td>
<td>blk_getrow</td>
<td>blk_sendrow</td>
</tr>
<tr>
<td>blk_colval</td>
<td>blk_gettxt</td>
<td>blk_sendtext</td>
</tr>
<tr>
<td>blk_default</td>
<td>blk_init</td>
<td>blk_srvinit</td>
</tr>
<tr>
<td>blk_describe</td>
<td>blk_rowalloc</td>
<td>blk_textxfer</td>
</tr>
<tr>
<td>blk_done</td>
<td>blk_rowdrop</td>
<td></td>
</tr>
</tbody>
</table>
Testing a TRS Installation with Sample Programs

This appendix describes the TRS sample programs. It provides instructions for testing TRS using LU 6.2 and TCP/IP network protocols.

This appendix contains the following topics:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>When to test your installation</td>
<td>149</td>
</tr>
<tr>
<td>Where to find the sample programs</td>
<td>149</td>
</tr>
<tr>
<td>How to test your TRS installation</td>
<td>150</td>
</tr>
</tbody>
</table>

This appendix describes the steps required to define a sample connection or region and RPC for testing only. The administration procedures for defining regions and RPCs to TRS are described in detail in Chapter 3, “Configuring a TRS” Also see Configuration quick-start in “Configuration quick-start” on page 43.

When to test your installation

Use the instructions in this appendix after all of the mainframe access product components are installed at the workstation and at the mainframe.

Where to find the sample programs

The sample programs are located in the following directories:

- For Windows: %SYBASE%\sample\trs
How to test your TRS installation

Follow the steps in this section to ensure that TRS is installed correctly. This section describes how to define a single region and RPC, and how to test them before you define other regions.

Starting TRS

Start DirectConnect with TRS enabled. Run the samples with security disabled by setting the TRS Security configuration property to no.

Defining the connection for Windows (LU 6.2 only)

Follow these instructions to define the test connection.

1. Log in to TRS as “sa” using isql or your preferred dynamic SQL utility. For example, enter:
   
isql -S service_name -Usa -P
   
   where service_name is the unique name of this TRS.

2. At the prompt, enter a command similar to the following, replacing the parameter values shown here with values that are appropriate for your installation.
   
   exec sgw_addcon con_name, region, mode, "max_sessions"
   
   where

   • con_name is the name assigned to this connection. This is the name by which the connection is known to your SNA support. For different platforms, this parameter corresponds to different values. See the Mainframe Connect DirectConnect for z/OS Option Installation Guide for specific information about connection name parameter values.

   For UNIX: $SYBASE/sample/trs

Note: The samples require Open Client DB-Library on the workstation.
• **region** specifies the remote LU name of the mainframe transaction processing region in this parameter. This is the Virtual Telecommunications Access Method (VTAM) APPLID name to which this connection is bound. An entry in this field is required.

All RPCs that use this connection to access the mainframe must have this same value specified as the **region** in their RPC definitions. (See also “Adding an RPC” on page 50.)

• **mode** needs to match this value to the name of the mode defined to the mainframe and to the local SNA support for this connection (up to eight characters). For different platforms, this parameter corresponds to different values. See the Mainframe Connect DirectConnect for z/OS Option Installation Guide for specific information about the mode name parameter value.

• **"max_sessions"** is the maximum number of sessions that can run concurrently over this connection. If you use parallel sessions, enter a value between 2 and 254. If you do not use parallel sessions, this value can only be 1. Enclose numeric parameter values in quotation marks.

**Note** Check with your SNA System Administrator to make sure this number is not larger than the maximum number of sessions (for this mode) defined to the SNA subsystem.

**Example**

In the following `isql` example, SYBLU01 is the connection name, CICSQA is the region name, SYBMODE is the mode name, and “1” is the number of maximum sessions.

```
exec sgw_addcon SYBLU01, CICSQA, SYBMODE, "1"
go
```

**Defining the test region (TCP/IP only)**

Follow these instructions to define the test region.

1. **Log in to TRS as “sa” using `isql` or your preferred dynamic SQL utility. For example, enter:**

   `isql -Sservice_name -Usa -P`

   where `service_name` is the unique name of this TRS.
2 At the isql prompt, enter a command similar to the following, replacing the parameter values shown here with values that are appropriate for your installation.

```bash
exec sgw_addregion region, hostname, portnumber, regiontype
```

where

- `region` is the value used within TRS only. The value you specify here must match the value you specify in the `region` parameter of the `sgw_addrpc` procedure. This name can be up to eight characters.

- `hostname` is the value you specify for the TCP/IP network host name. This is the name corresponding to the mainframe in your `/etc/hosts` file or in your NIS map. This name can be up to 30 characters.

- `portnumber` is the number you specify that must match the port number on which the CSKL transaction listens. It can be any number between 1024 and 9996. (This is not the same as the port number used to configure the `interfaces` file.)

- `regiontype` is the type of the mainframe processing environment specified by the `region` parameter. Valid values are CICS, MVS, and IMS. If you do not specify a value, the region type defaults to CICS.

**Example**

In the following example, `CICSQA` is the region, `BLUES` is the host name, “3003” is the port number that the CICS Listener transaction is running on, and `CICS` is the region type.

```bash
exec sgw_addregion CICSQA, BLUES, "3003", CICS
```

**Defining the test RPC**

Define an RPC to execute in the specified region. The SYM2 transaction is a simple CICS transaction that fabricates data. It does not require external resources such as DB2 UDB.

- At the prompt, enter a command similar to the one shown below.

```bash
exec sgw_addrpc rpc_name, tran_id, region, security
```

where

- `rpc_name` is the name of the remote procedure. This is the name the client uses to call this RPC. The name can be up to 30 characters.
APPENDIX B Testing a TRS Installation with Sample Programs

- **tran_id** is the name of the associated mainframe transaction. This is the mainframe transaction that is called when a client requests the named procedure. The value of this field must be in uppercase. For CICS, use four characters. For IMS, use up to eight characters.

- **region** (LU 6.2 only) specifies the remote LU name of the region in this parameter. Set this value to match the VTAM APPLID of the CICS or IMS region (the destination subsystem) in which the transaction (specified in tran_id) executes.

  At least one defined connection must have this value specified as its region. See also “Adding a connection configuration” on page 46. An entry in this field is required.

- **region** (TCP/IP only) is used within TRS only to represent the CICS region name. It must match the value you specify for the region parameter in the sgw_addregion procedure. See “Defining regions to TRS” on page 48. An entry in this field is required.

- **security** specifies the type of user login information to be passed to the transaction processing region.

  - Using LU 6.2, the information is passed in the conversation-level security fields of the SNA LU 6.2 Function Management Header 5 (FMH-5).

  - Using TCP/IP, these fields are sent to the CICS Listener Transaction when the CICS transaction is started.

  The security parameter can have any of the following values to specify which information is sent:

  - **none** – do not send login information to the mainframe.
  - **userid** – send only the user ID to the mainframe.
  - **both** – send both the user ID and the password to the mainframe.

  For example, if you use native CICS security, the none value corresponds to the CICS security option **NONE**, userid corresponds to **IDENTIFY**, and both corresponds to the security option **VERIFY**.

**Example**

```
exec sgw_addrpc SYM2, SYM2, CICSQA, none
```

where

- **SYM2** (first entry) is the RPC name.
- **SYM2** (second entry) is the transaction ID at the mainframe.
- **CICSQA** is the CICS region name.

User’s Guide for Transaction Router Services 153
How to test your TRS installation

- none indicates that user IDs are not passed to the mainframe.

The CICS region name (CICSQA in the preceding example) must match the following:

- For TCP/IP, the region name given in the sgw_addregion procedure.
- For LU 6.2, the region parameter in the sgw_addcon procedure.

Running the sample

Enter the following at the dynamic SQL utility prompt to run the SYM2 sample:

eexec SYM2 a, 4

The output should be similar to the following:

```
TESTDATA

-------------------------------
U6T42P01
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
U6T42P01
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
U6T42P01
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
U6T42P01
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
```

(4 rows affected, return status = 0)

Checking for error messages

The TRS request can return any of several types of error messages. Some messages are written to the error log at TRS, while others are returned to the client.

For more information, see the DirectConnect Error Message Guide.
Looking at additional sample programs

After you successfully run the SYM2 sample, continue with some of the other samples provided in the following directories and in the README file in that directory:

- For non-UNIX: %SYBASE%/sample/TRS/sym2
- For UNIX: $SYBASE/sample/TRS/sym2

Define the samples to TRS using the administration procedures described previously.

SYVn transactions

The SYVn transactions read a VSAM file and return the records. The SYVn RPC passes two parameters: a starting and ending byte offset.

Add the RPC using the sgw_addrpc procedure described under “Defining the test RPC” on page 152. These transactions call the following programs:

- SYV1 calls the PL/I program, SYCASA1.
- SYV2 calls a COBOL program, SYCASA2.
- SYV3 calls an assembler program, SYCASA3, which is supplied on the mainframe.

To execute the SYVn RPC you defined, do one of the following:

- Enter the following command at the dynamic SQL utility prompt: (Replace SYVn with SYV1, SYV2, or SYV3.)
  
  \texttt{exec SYVn 0,9999}

- If the directory containing the samples is on the search path, enter this command at the server console prompt:

  \texttt{SYVn 0,9999}

The output should be similar to the following:

\begin{verbatim}
SYGWLSA1
---------------------------------------------------
sample vsam rpc data rec 0 sample vsam rpc data rec 0
sample vsam rpc data rec 1 sample vsam rpc data rec 1
sample vsam rpc data rec 2 sample vsam rpc data rec 2
`
`
(10 rows affected, return status = 0)
\end{verbatim}
How to test your TRS installation

SYDn Transactions

SYDn transactions execute a static DB2 UDB query to one of the DB2 UDB sample tables. The parameter is the department number. Add the RPC using the sgw_addrpc procedure described in “Defining the test RPC” on page 152.

To execute the SYDn RPC, do one of the following:

1. Enter this command at the isql prompt (replace SYDn with SYD1 or SYD2):
   ```
   exec SYDn D11
   go
   ```

2. If the directory containing the samples is on the search path, enter this command at the server console prompt:
   ```
   SYDn D11
   ```

The output should be similar to the following:

<table>
<thead>
<tr>
<th>LAST_NAME</th>
<th>EMP_DEPT</th>
<th>EMP_PHONE</th>
<th>SALARY</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADAMSON</td>
<td>D11</td>
<td>4510</td>
<td>25,280.00</td>
</tr>
<tr>
<td>BROWN</td>
<td>D11</td>
<td>4501</td>
<td>27,740.00</td>
</tr>
<tr>
<td>PIANKA</td>
<td>D11</td>
<td>3782</td>
<td>22,250.00</td>
</tr>
<tr>
<td>STERN</td>
<td>D11</td>
<td>6423</td>
<td>32,250.00</td>
</tr>
<tr>
<td>WALKER</td>
<td>D11</td>
<td>2986</td>
<td>20,450.00</td>
</tr>
<tr>
<td>LUTZ</td>
<td>D11</td>
<td>0672</td>
<td>29,840.00</td>
</tr>
<tr>
<td>SCOUTTEN</td>
<td>D11</td>
<td>1682</td>
<td>21,340.00</td>
</tr>
<tr>
<td>YOSHIMURA</td>
<td>D11</td>
<td>2890</td>
<td>24,680.00</td>
</tr>
<tr>
<td>JONES</td>
<td>D11</td>
<td>0942</td>
<td>18,270.00</td>
</tr>
</tbody>
</table>

(9 rows affected, return status = 0)

The rest of the sample transactions demonstrate Open ServerConnect programming techniques.

Looking at catalog RPC scripts

If you plan to use the Catalog RPCs, see the appropriate section in this guide for instructions on running the addcat installation script.

The scripts that install, test, and delete the Catalog RPCs are provided in the following directories:

1. For Windows: %SYBASE%\scripts
2. For UNIX: $SYBASE/scripts
APPENDIX C

Localization

This appendix describes localization information for TRS. If you need more complete information about localization, refer to the Open Client/Server Supplement for your platform.

This appendix contains the following topics:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>What is localization?</td>
<td>157</td>
</tr>
<tr>
<td>Environment variables for localization</td>
<td>159</td>
</tr>
<tr>
<td>Localization files</td>
<td>160</td>
</tr>
<tr>
<td>How Client-Library and Server-Library set up default localization</td>
<td>163</td>
</tr>
</tbody>
</table>

What is localization?

Localization is the process of setting up an application to run in a particular national language environment. A localized application:

- Generates messages in a local language and character set
- Uses local datetime formats

A locale name is a character string that represents a language, character set, and sort order combination. For example, the locale name “fr” might represent the following language, character set, and sort order combination:

    french/iso_1/binary

Sybase predefines locale names, which are listed in the locales file. For information on the locales file.
What is localization?

How servers handle conversions

When a localized client application connects to TRS, Adaptive Server, or Open Server, the server checks to see if it supports the client’s language and character set. If it does, then the server:

- Automatically handles all character set translation
- Issues server messages in the client’s language and character set

If TRS does not support the language or character set sort of the client, it issues a warning message to this effect, and Client–Library fails the connection. However, DB-Library accepts the connection.

The following table describes these client and server behaviors:

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>yes</td>
<td>yes</td>
<td>Performs all necessary message translation and character set conversion</td>
<td>Performs all necessary message translation and character set conversion</td>
<td>Operates normally</td>
<td>Operates normally</td>
</tr>
<tr>
<td>no</td>
<td>yes</td>
<td>N/A for Adaptive Server, because when Adaptive Server supports a language, it supports all character sets for that language</td>
<td>Uses the language and character set of the Open Server application</td>
<td>N/A for Adaptive Server; fails the connection for Open Server</td>
<td>N/A for Adaptive Server; accepts the connection for Open Server</td>
</tr>
<tr>
<td>yes</td>
<td>no</td>
<td>Uses the language us_english and the client’s character set</td>
<td>Uses the language and character set of the Open Server application</td>
<td>Fails the connection</td>
<td>Accepts the connection</td>
</tr>
</tbody>
</table>
Environment variables for localization

TRS examines environment variables when determining which language, character set, sort order, and datetime formats to use for an application.

TRS uses standard POSIX localization environment variables.

Some systems automatically set environment variables when a user logs in. If your system does this, either reset the variables after logging in or make sure that their automatic values correspond to an entry in the Sybase locales file.

The following table lists the environment variables that are related to TRS localization:

**Table C-2: TRS localization environment variables**

<table>
<thead>
<tr>
<th>Environment variable</th>
<th>Definition</th>
<th>When</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>LC_ALL</strong></td>
<td>Indicates which language and character set to use for messages, datatype conversions, and datetime formats.</td>
<td>- The application calls cs_ctx_alloc.&lt;br&gt; - The application calls cs_locale with type as CS_LC_ALL and buffer as NULL.</td>
</tr>
<tr>
<td><strong>LC_CTYPE</strong></td>
<td>Indicates which character set to use for datatype conversions.</td>
<td>The application calls cs_locale with type as CS_LC_CTYPE and buffer as NULL.</td>
</tr>
<tr>
<td><strong>LC_COLLATE</strong></td>
<td>Indicates which collating sequence (sort order) to use when sorting and comparing character data.</td>
<td>The application calls cs_locale with type as CS_LC_COLLATE and buffer as NULL.</td>
</tr>
<tr>
<td><strong>LC_MESSAGE</strong></td>
<td>Indicates which language and character set to use for messages.</td>
<td>The application calls cs_locale with type as CS_LC_MESSAGE and buffer as NULL.</td>
</tr>
</tbody>
</table>
Localization files

This section contains information on Sybase files that are related to localization.

**Note** The directories shown in this appendix are for a Windows platform. For UNIX platforms, the directory path is `$SYBASE/locales` or `$SYBASE/charsets`.

Where localization files come from

Open Client/Server products, including TRS, come with the files to support one language and one or more character sets and sort orders.

At installation time, these files are automatically loaded into the `%SYBASE%` directory tree, in the locations illustrated in Table C-3.
The files to support additional languages are packaged as “Language Modules for Connectivity.”

When you install a language module, the language, character set, and sort order files to support the new language are automatically loaded into the %SYBASE% directory tree in the correct locations.

**Location of localization files**

Two directories in the %SYBASE% directory tree contain files related to localization:

- %SYBASE%\locales, which contains the locales file (locales.dat) and a subdirectory for each available language.
- %SYBASE%\charsets, which contains a subdirectory for each available character set.

The following table shows where localization files are located in the %SYBASE% directory tree:

<table>
<thead>
<tr>
<th>Subdirectory</th>
<th>charset_name</th>
<th>binary.srt</th>
<th>charset.loc</th>
<th>dictionary.srt</th>
<th>noaccents.srt</th>
<th>nocase.srt</th>
<th>nocasepref.srt</th>
</tr>
</thead>
<tbody>
<tr>
<td>charsets</td>
<td>charset_name</td>
<td>binary.srt</td>
<td>charset.loc</td>
<td>dictionary.srt</td>
<td>noaccents.srt</td>
<td>nocase.srt</td>
<td>nocasepref.srt</td>
</tr>
<tr>
<td>locales</td>
<td>language_name</td>
<td>charset_name</td>
<td>blklib.loc</td>
<td>ctlib.loc</td>
<td>common.loc</td>
<td>cslib.loc</td>
<td>oslib.loc</td>
</tr>
</tbody>
</table>

| locales.dat |

The following table shows information about some localization files.
### Localization files

**Table C-4: Files related to localization**

<table>
<thead>
<tr>
<th>File name</th>
<th>File location</th>
<th>What it contains</th>
</tr>
</thead>
<tbody>
<tr>
<td>locales.dat</td>
<td>%SYBASE%\locales\</td>
<td>Entries that map a locale name to a language and character set. This is the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>locales file. For more information, see “Locales file” on page 162.</td>
</tr>
<tr>
<td>common.loc</td>
<td>%SYBASE%\locales\language_name\charset_name*</td>
<td>Common information for the language_name language and charset_name character set, including date names and orders and money formats and symbols.</td>
</tr>
<tr>
<td>charset.loc</td>
<td>%SYBASE%\locales\language_name\charset_name*</td>
<td>Character set information for the language_name language and charset_name character set.</td>
</tr>
<tr>
<td>binary.srt</td>
<td>%SYBASE%\charset\charset_name*</td>
<td>The binary sort order for the charset_name character set.</td>
</tr>
</tbody>
</table>

***.loc files**

The %SYBASE%\locales\language_name\charset_name\*.loc files contain product error messages in the language and character set specified by their parent directories.

These files enable TRS to report errors in a specific language and character set.

**Character set files**

The %SYBASE%\charsets\charset_name\* files contain information related to a particular character set, including sort order, case, and accent information.

**Locales file**

The locales file associates locale names with languages, character sets, and sort orders. TRS uses the locales file when loading localization information.

The locales file is called locales.dat and is located in the %SYBASE%\locales directory.

The locales file directs TRS to language, character set, and sort order names, but does not contain actual localized messages or character set information.
Format of locales file entries

The locales file has platform-specific sections. An entry defines a locale as the combination of a language, character set, and sort order.

\[
\text{locale} = \text{locale\_name}, \text{language\_name}, \text{charset\_name} \ [, \text{sort\_order\_name}]
\]

If the sort order is not specified, it is “binary.”

When the locale being defined is the default for the site, the locale\_name is “default.” For example, the following entry defines the default locale as us\_english with the iso\_1 character set and binary sort order:

\[
\text{locale} = \text{default}, \text{us\_english}, \text{iso\_1}
\]

How Client-Library and Server-Library set up default localization values

When a Client-Library or Server-Library application calls the CS-Library routine cs\_ctx\_alloc to allocate a context structure, CS-Library loads default localization information into the new context structure.

To load default localization information, CS-Library follows these steps:

1. CS-Library looks for a locale name, searching for the following environment variables, in order: LC\_ALL and LANG
   - If LC\_ALL is defined, CS-Library uses its value as the locale name. If LC\_ALL is not defined but LANG is defined, CS-Library uses its value as the locale name.
   - If neither LC\_ALL nor LANG is defined, CS-Library uses a locale name of “default.”

2. CS-Library looks up the locale name in the locales file to determine which language and character set are associated with it.

3. CS-Library loads localized messages and character set information appropriate to the language and character set determined in step 2.

This process provides the new context structure with all of the localization information that it needs.
How Client-Library and Server-Library set up default localization values
This chapter describes the steps to implement and use the TRS process user exits for both LU 6.2 and TCP/IP. TRS allows you to create user exits that are invoked from the TRS application prior to executing the actual event.

This appendix contains the following topics:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supported user exits</td>
<td>165</td>
</tr>
<tr>
<td>Implementing user exits</td>
<td>166</td>
</tr>
<tr>
<td>Configuring TRS to implement user exits</td>
<td>168</td>
</tr>
<tr>
<td>Testing user exits</td>
<td>168</td>
</tr>
<tr>
<td>Interface specifications</td>
<td>169</td>
</tr>
</tbody>
</table>

**Supported user exits**

TRS supports user exits corresponding to Open Server defined Connect and Disconnect events. Within user exits you are able to manipulate User ID and User Password information prior to the event being executed by TRS. It is not necessary to modify any of this information to implement user exits. However, you may need to manipulate the password for security reasons for an application. For example, a user’s password may be modified to become a time-restricted password for interpretation by an authentication server.

The following are the Open Server defined events for which user exits are supported:

- Connect
- Disconnect
Implementing user exits

The directories shown in this appendix are for UNIX platforms. For Windows platforms, the variable %SYBASE%/%SYBASE_ECON% is used.

Connect

The Connect user exits provides the ability to override both the User ID and User Password. The user exits is called at the end of all DirectConnect connection processing, but prior to the TRS connection processing. This allows the User ID and User Password to remain intact for Open Client and Open Server connections and change for connections from TRS to DB2 UDB.

The TRS connection handler will query the length of the user exits returned User ID and Password string buffers to determine data content. When data is present, it will be transferred to the TRS User ID and Password buffers.

Refer to the following directory for an implementation example:

$SYBASE/$SYBASE_ECON/servername/sample/trs/ue/ue_connect.cpp

Disconnect

The Disconnect user exit is called only when the client disconnects from TRS. All parameters are passed as constants and cannot be modified.

Refer to the following directory for an implementation example:

$SYBASE/$SYBASE_ECON/servername/sample/trs/ue/ue_disconnect.cpp

Implementing user exits

TRS user exits for Open Server Connect and Disconnect events should be written and tested using the sample exits and test harness provided at $SYBASE/$SYBASE_ECON/serverName/sample/trs/ue. Following is a description of the source, header, makefiles, libraries and executable files, many of which are dependent on your platform.

The source files are defined as:
APPENDIX D  TRS Process User Exits

- **ue_connect.cpp** – connection event sample user exit code.
- **ue_disconnect.cpp** – disconnect event sample user exit code.
- **ue_test.cpp** – test harness to invoke the Connect and Disconnect user exits.

The following header files, depending on your platform:
- **ue_platform.h** – platform required header for implementing user exits where `<platform>` is one of: aix, hpux, sol, nt.
- **ue_classes.h** – required by **ue_test.cpp**.
- **ue_global.h** – required user exit definitions.

The following makefiles, depending on your platform:
- **makeexe.platform**
  where `platform` is: aix, hpux, sol, or nt.
  Generates **ue_test <platform>**, the **user exit** test harness.
- **makefile.platform**
  where `platform` is: aix, hpux, sol, or nt.
  Generates required library **libtrsue_platform.ext**
  where `ext` is .so for AIX and Sun Solaris, .sl for HP-UX, and .dll for Windows.

The following libraries, depending on your platform:
- **libue_platform.a**
  Required library for testing user exits with **ue_test.platform**, where `platform` is: aix, hpux, sol, or nt.
- **libtrsue_platform.ext**
  Library containing user exits generated by makefile.platform
  where `platform` is: aix, hpux, sol, or nt, and
  `ext` is .so for AIX and Sun Solaris, .sl for HP-UX, and .dll for Windows.

The following executable, depending on your platform:
  **ue_test.platform** Test harness produced from **makeexe.platform**
  where `platform` is: aix, hpux, sol, or nt.
Configuring TRS to implement user exits

One parameter is required by `ue_test.<platform>` which specifies the dynamic library to be loaded. Following is an example:

```
ue_test.<platform> ./libtrsue_<platform><ext>
```

where `platform` is: `aix`, `hpux`, `sol`, or `nt` and
`ext` is `.so` for AIX and Sun Solaris, `.sl` for HP-UX, and `.dll` for Windows.

Configuring TRS to implement user exits

To implement TRS user exits use the following properties:

- **ProcessExitEnabled**
  
  Set to `yes` to enable the use of user exits. Only the exits that you have defined and added to your user exit library will be invoked.

- **ProcessExitFile**
  
  Full path and name of the user exit shared library that you have created. From the sample code, `libtrsue_platform.ext` is its equivalent.

- **TraceProcessUserExits**
  
  Traces entry/exit points of function call to each of the user exits that you have defined. Normal setting is `no`; however, a setting of `yes` will assist you in determining execution through your exits.

Testing user exits

To test the provided samples, simply

```
execute makefile.<platform>
```

followed by

```
makeexe.<platform>
```

Continue by executing the following:

```
ue_test.<platform> ./libtrsue_platform.ext
```

For example, on Solaris the following is required:
make -f makefile.sol
make -f makeexe.sol
ue_test.sol ./libtrsue.sol.so

Note Although exits other than Connect and Disconnect are provided in the sample $SYBASE/$SYBASE_ECON/servername/sample/trs/ue, these exits are not supported at this time.

Interface specifications

Following are the required interfaces and their descriptions for implementing the Connect and Disconnect user exits.

ue_connect

Description Connects the event user exit defining what actions are to be performed prior to TRS connecting to DB2 UDB.

Syntax

```c
TRS_RETCODE TRS_PUBLIC
ue_connect (status, serviceName, serviceNameLength, applicationName, applicationNameLength, userId, pUserIdLength, password, pPasswordLength, pOutUserId, pOutPwd)
```

```c
TRS_STATUS status;
const char* serviceName;
const int serviceNameLength;
const char* applicationName;
const int applicationNameLength
const char* userId;
int* pUserIdLength;
const char* password;
int* pPasswordLength;
char** pOutUserId;
char** pOutPwd;
```
Parameters

status

The state of the Service Library invoking `ue_connect()`. The following table describes the legal value for status:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>eTRS_FAIL</td>
<td>Indicates failure within <code>ue_connect()</code></td>
</tr>
<tr>
<td>eTRS_SUCCEED</td>
<td>Indicates success within <code>ue_connect()</code></td>
</tr>
<tr>
<td>eTRS_NOTIMPLEMENTED</td>
<td>Indicates stubbed out implementation of <code>ue_connect()</code></td>
</tr>
<tr>
<td>eTRS_DATAMODIFIED</td>
<td>Indicates that a pointer variable has been modified and implies eTRS_SUCCEED</td>
</tr>
</tbody>
</table>

serviceName

Name of the service to which the connection was made.

serviceNameLength

Length of the `serviceName`.

applicationName

Name of the application from which the connection was made.

applicationNameLength

Length of the `applicationName`.

userId

ID of the connecting user.

pUserIdLength

Pointer to length of `userId`.

password

Password associated with `userId`.

pPasswordLength

Pointer to length of `password`. 
pOutUserId
A pointer to a character string that must be allocated by this routine and may
contain a modified userId.

pOutPwd
A pointer to a character string that must be allocated by this routine and may
contain a modified password.

Usage
See syntax.

Comments

- \texttt{ue\_connect()} must return \texttt{eTRS\_DATAMODIFIED} if either \texttt{pOutUserId} or
\texttt{pOutPwd} has been allocated.

- \texttt{pUserIdLength} must be updated to reflect the length of \texttt{pOutUserId} when
\texttt{pOutUserId} has been allocated.

- \texttt{pPasswordLength} must be updated to reflect the length of \texttt{pOutPwd} when
\texttt{pOutPwd} has been allocated.

- \texttt{malloc()} should be used to allocate space for \texttt{pOutUserId} and \texttt{pOutPwd}

\underline{Warning!} By granting control to this user exit, DirectConnect has temporarily
forfeited the management of Open Server threads. The result is that
DirectConnect cannot ensure against \texttt{ue\_connect} monopolizing execution, nor
the ability of \texttt{ue\_connect} to create a deadlock. Please take precautions to
prevent this.

Example
Refer to a sample implementation at the following location:
\texttt{$SYBASE/$SYBASE\_ECON/servername/sample/trs/ue/ue\_connect.cpp}

\textbf{ue\_disconnect}

\textbf{Description}
Defines what actions are to be performed prior to a client disconnecting from
TRS.

\textbf{Syntax}

\begin{verbatim}
TRS_RETCODE TRS_PUBLIC
ue_disconnect (status, serviceName, serviceNameLength, applicationName,
applicationNameLength, userId, pUserIdLength)

TR_S_STATUS status;
const char* serviceName;
\end{verbatim}
**ue_disconnect**

const int serviceNameLength;
const char* applicationName;
const int applicationNameLength
ccnst char* userId;
int* pUserIdLength;

**Table D-3: TRS_RETCODE values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>eTRS_FAIL</td>
<td>Indicates failure within <code>ue_disconnect()</code></td>
</tr>
<tr>
<td>eTRS_SUCCEED</td>
<td>Indicates success within <code>ue_disconnect()</code></td>
</tr>
<tr>
<td>eTRS_NOTIMPLEMENTED</td>
<td>Indicates stubbed out implementation of <code>ue_disconnect()</code></td>
</tr>
</tbody>
</table>

**Parameters**

`status`
The state of the Service Library invoking `ue_disconnect()`.

The following table describes the legal value for `status`:

**Table D-4: Legal status values**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>eGood_</td>
<td>Always used</td>
</tr>
<tr>
<td>eObjNotFound_</td>
<td>Reserved for future use</td>
</tr>
<tr>
<td>eObjNotValue_</td>
<td>Reserved for future use</td>
</tr>
<tr>
<td>eFatal_</td>
<td>Reserved for future use</td>
</tr>
</tbody>
</table>

`serviceName`
Name of the service to which the connection was made.

`serviceNameLength`
Length of the `serviceName`.

`applicationName`
Name of the application from which the connection was made.

`applicationNameLength`
Length of the `applicationName`.

`userId`
ID of the connecting user.

`pUserIdLength`
Pointer to length of `userId`.

**Usage**

See syntax.
Comments

- `ue_disconnect` allows you to perform varying functions related to disconnects. Although `pUserIdLength` is defined as a pointer, its modification is meaningless with this release.

**Warning!** By granting control to this user exit, DirectConnect has temporarily forfeited the management of Open Server threads and DirectConnect cannot ensure against `ue_disconnect` monopolizing execution, nor `ue_disconnect`’s ability to create a deadlock. Please use precautions to prevent this.

Example

Refer to a sample implementation at the following location:

`$SYBASE/$SYBASE_ECON/servername/sample/trs/ue/ue_connect.cpp`
Compatibility with MDI Database Gateway and Net-Gateway

This appendix provides information regarding the compatibility between DirectConnect and both the MDI Database Gateway and Net-Gateway.

This appendix contains the following topics:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compatibility with MDI Database Gateway</td>
<td>175</td>
</tr>
<tr>
<td>Compatibility with Net-Gateway</td>
<td>175</td>
</tr>
</tbody>
</table>

Compatibility with MDI Database Gateway
If you have client applications for MDI Database Gateways, you can still use those applications with DB2 UDB access services.

Compatibility with Net-Gateway
If you have client applications for Net-Gateway, those applications will work with TRS. Simply follow the file-moving instructions in the next subsection.
<table>
<thead>
<tr>
<th>Move Net-Gateway files</th>
<th>To these TRS files</th>
</tr>
</thead>
<tbody>
<tr>
<td>For UNIX:</td>
<td>$SYBASE/$SYBASE_ECON (UNIX) or %SYBASE_ECON% (NT) /servername/cfg/ngcid.&lt;trs_service_library_name&gt;</td>
</tr>
<tr>
<td>$SYBASE/ngcid.&lt;msg_server_name&gt;</td>
<td></td>
</tr>
<tr>
<td>For TCP/IP on non-UNIX:</td>
<td>%SYBASE%$SYBASE_ECON (UNIX) or %SYBASE_ECON% (NT) \servername\cfg&lt;trs&gt;.ngcid</td>
</tr>
<tr>
<td>%SYBASE%ngcid.&lt;msg_server_name&gt;</td>
<td></td>
</tr>
<tr>
<td>For LU 6.2 on non-UNIX:</td>
<td>%SYBASE%$SYBASE_ECON (UNIX) or %SYBASE_ECON% (NT) \servername\cfg&lt;trs&gt;.cid</td>
</tr>
<tr>
<td>%SYBASE%&lt;msg_server_name&gt;.cid</td>
<td></td>
</tr>
<tr>
<td>For UNIX:</td>
<td>$SYBASE/$SYBASE_ECON (UNIX) or %SYBASE_ECON% (NT) /servername/cfg/ngreg.&lt;trs_service_library_name&gt;</td>
</tr>
<tr>
<td>$SYBASE/ngreg.&lt;msg_server_name&gt;</td>
<td></td>
</tr>
<tr>
<td>For TCP/IP on non-UNIX:</td>
<td>%SYBASE%$SYBASE_ECON (UNIX) or %SYBASE_ECON% (NT) \servername\cfg&lt;trs&gt;.ngreg</td>
</tr>
<tr>
<td>%SYBASE%ngreg.&lt;msg_server_name&gt;</td>
<td></td>
</tr>
<tr>
<td>For LU 6.2 on non-UNIX:</td>
<td>%SYBASE%$SYBASE_ECON (UNIX) or %SYBASE_ECON% (NT) \servername\cfg&lt;trs&gt;.reg</td>
</tr>
<tr>
<td>%SYBASE%&lt;msg_server_name&gt;.reg</td>
<td></td>
</tr>
<tr>
<td>For UNIX:</td>
<td>$SYBASE/$SYBASE_ECON (UNIX) or %SYBASE_ECON% (NT) /servername/cfg/nggrp.&lt;trs_service_library_name&gt;</td>
</tr>
<tr>
<td>$SYBASE/nggrp.&lt;msg_server_name&gt;</td>
<td></td>
</tr>
<tr>
<td>For TCP/IP on non-UNIX:</td>
<td>%SYBASE%$SYBASE_ECON (UNIX) or %SYBASE_ECON% (NT) \servername\cfg&lt;trs&gt;.nggrp</td>
</tr>
<tr>
<td>%SYBASE%nggrp.&lt;msg_server_name&gt;</td>
<td></td>
</tr>
<tr>
<td>For LU 6.2 on non-UNIX:</td>
<td>%SYBASE%$SYBASE_ECON (UNIX) or %SYBASE_ECON% (NT) \servername\cfg&lt;trs&gt;.grp</td>
</tr>
<tr>
<td>%SYBASE%&lt;msg_server_name&gt;.grp</td>
<td></td>
</tr>
</tbody>
</table>
## APPENDIX E  Compatibility with MDI Database Gateway and Net-Gateway

<table>
<thead>
<tr>
<th>Move Net-Gateway files</th>
<th>To these TRS files</th>
</tr>
</thead>
<tbody>
<tr>
<td>For UNIX:</td>
<td></td>
</tr>
<tr>
<td>$SYBASE/ngrpc.&lt;msg_server_name&gt;</td>
<td>$SYBASE/$SYBASE_ECON (UNIX) or %SYBASE_ECON% (NT)</td>
</tr>
<tr>
<td></td>
<td>/&lt;servername&gt;/cfg/ngrpc.&lt;trs_service_library_name&gt;</td>
</tr>
<tr>
<td>For TCP/IP on non-UNIX:</td>
<td></td>
</tr>
<tr>
<td>%SYBASE%\ngrpc.&lt;msg_server_name&gt;</td>
<td>%SYBASE%$SYBASE_ECON (UNIX) or %SYBASE_ECON% (NT)</td>
</tr>
</tbody>
</table>
|                        | %SYBASE%\nservername\cfg\<trs>\ngrpc |%
| For LU 6.2 on non-UNIX:|                   |
| %SYBASE%\<msg_server_name>.rpc | %SYBASE%\$SYBASE_ECON (UNIX) or %SYBASE_ECON% (NT) |
|                        | %SYBASE%\nservername\cfg\<trs>.ngrpc |%
| For UNIX:              |                   |
| $SYBASE/ngact.<msg_server_name> | $SYBASE/$SYBASE_ECON (UNIX) or %SYBASE_ECON% (NT) |
|                        | /<servername>/log/ngact.<trs_service_library_name> |
| For TCP/IP on non-UNIX:|                   |
| %SYBASE%\ngact.<msg_server_name> | %SYBASE%\$SYBASE_ECON (UNIX) or %SYBASE_ECON% (NT) |
|                        | %SYBASE%\nservername\cfg\<trs>\ngact |%
| For LU 6.2 on non-UNIX:|                   |
| %SYBASE%\<msg_server_name>.act | %SYBASE%\$SYBASE_ECON (UNIX) or %SYBASE_ECON% (NT) |
|                        | %SYBASE%\nservername\cfg\<trs>.ngact |%
| For UNIX:              |                   |
| $SYBASE/ngtds.<msg_server_name> | $SYBASE/$SYBASE_ECON (UNIX) or %SYBASE_ECON% (NT) |
|                        | /<servername>/log/ngtds.<trs_service_library_name> |
| For TCP/IP on non-UNIX:|                   |
| %SYBASE%\ngtds.<msg_server_name> | %SYBASE%\$SYBASE_ECON (UNIX) or %SYBASE_ECON% (NT) |
|                        | %SYBASE%\nservername\cfg\<trs>.tds |%
| For LU 6.2 on non-UNIX:|                   |
| %SYBASE%\<msg_server_name>.tds | %SYBASE%\$SYBASE_ECON (UNIX) or %SYBASE_ECON% (NT) |
|                        | %SYBASE%\nservername\cfg\<trs>.tds |%
### Compatibility with Net-Gateway

<table>
<thead>
<tr>
<th>Move Net-Gateway files</th>
<th>To these TRS files</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>For UNIX:</strong></td>
<td></td>
</tr>
<tr>
<td>$SYBASE/nglog.&lt;msg_server_name&gt;</td>
<td>$SYBASE/$SYBASE_ECON (UNIX) or %SYBASE_ECON% (NT) /servername/log/servername.log</td>
</tr>
<tr>
<td><strong>For TCP/IP on non-UNIX:</strong></td>
<td></td>
</tr>
<tr>
<td>%SYBASE%\nglog.&lt;msg_server_name&gt;</td>
<td>%SYBASE%$SYBASE_ECON (UNIX) or %SYBASE_ECON% (NT) \servername\log&lt;dcon_server_name&gt;.log</td>
</tr>
<tr>
<td><strong>For LU 6.2 on non-UNIX:</strong></td>
<td></td>
</tr>
<tr>
<td>%SYBASE%&lt;msg_server_name&gt;.log</td>
<td>%SYBASE%$SYBASE_ECON (UNIX) or %SYBASE_ECON% (NT) \servername\log&lt;dcon_server_name&gt;.log</td>
</tr>
</tbody>
</table>
**Glossary**

**accept** Establishment of a Open ServerConnect–DirectConnect SNA or TCP/IP connection.

**access service** The named set of properties, used with a DirectConnect Access Service Library, to which clients connect. Each DirectConnect server can have multiple services.

**Access Service Library** A service library that provides access to non-Sybase data contained in a database management system or other type of repository. Each such repository is called a “target.” Each Access Service Library interacts with exactly one target and is named accordingly. See also service library.

**ACSLIB** See access service library.

**Administrative Service Library** A service library that provides remote management capabilities and server-side support. It supports a number of remote procedures, invoked as RPC requests, that enable remote DirectConnect management. See also remote procedure call, service library.

**ADMLIB** See Administrative Service Library.

**Advanced Interactive Executive** The IBM implementation of the UNIX operating system. The RISC System/6000, among other workstations, runs the AIX operating system. See also UNIX.

**advanced program-to-program communication** Hardware and software that characterize the LU 6.2 architecture and its implementations in products. See also logical unit 6.2.

**AIX** See Advanced Interactive Executive.

**AMD2** The component of MainframeConnect for DB2 UDB that allows clients to submit SQL statements to DB2 UDB. It is a CICS transaction that receives SQL statements sent from DirectConnect and submits them to DB2 UDB, using the DB2 UDB dynamic SQL facility. It also receives the results and messages from DB2 UDB and returns them to DirectConnect.

**API** See application program interface.

**APPC** See advanced program-to-program communication.
<table>
<thead>
<tr>
<th>Term</th>
<th>Definition/Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>application program interface</td>
<td>The programming language interface between the user and Open ClientConnect or Open ServerConnect. The API for Open ClientConnect is Client-Library. The API for Open ServerConnect is Gateway-Library.</td>
</tr>
<tr>
<td>ASE/CIS</td>
<td>Adaptive Server Enterprise / Component Integration Services (formerly OmniConnect).</td>
</tr>
<tr>
<td>batch</td>
<td>A group of records or data processing jobs brought together for processing or transmission.</td>
</tr>
<tr>
<td>bind</td>
<td>In the Sybase environment, this term has different meanings depending on the context:</td>
</tr>
<tr>
<td></td>
<td>• In CICS, it is an SNA command used to establish a connection between LUs, or a TCP/IP call that connects an application to a port on its system.</td>
</tr>
<tr>
<td></td>
<td>• In DB2 UDB, it compiles the Database Request Module, the precompiler product that contains SQL statements in the incoming request, and produces an access plan, a machine code version of the SQL statements that specifies the optimal access strategy for each statement.</td>
</tr>
<tr>
<td></td>
<td>• In the mainframe access product set, it establishes a connection between a TRS port and a CICS or IMS region.</td>
</tr>
<tr>
<td>bulk copy</td>
<td>The utility for copying data in and out of databases.</td>
</tr>
<tr>
<td>catalog</td>
<td>A system table that contains information about objects in a database, such as tables, views, columns, and authorizations.</td>
</tr>
<tr>
<td>catalog RPC</td>
<td>A component of the DB2 UDB Access Module that allows clients to access DB2 UDB system catalogs. It uses an interface compatible with the catalog interface for the ODBC API.</td>
</tr>
<tr>
<td>catalog stored procedure</td>
<td>A procedure, used in SQL generation and application development, that provides information about tables, columns, and authorizations.</td>
</tr>
<tr>
<td>character set</td>
<td>A set of specific (usually standardized) characters with an encoding scheme that uniquely defines each character. ASCII is a common character set.</td>
</tr>
<tr>
<td>CICS</td>
<td>See Customer Information Control System.</td>
</tr>
<tr>
<td>CICS region</td>
<td>The CICS area of the computer system in which an application is running.</td>
</tr>
<tr>
<td>client</td>
<td>In client/server systems, the part of the system that sends requests to servers and processes the results of those requests. See also client/server. Compare with server.</td>
</tr>
</tbody>
</table>
### Glossary

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>client application</strong></td>
<td>Software responsible for the user interface that sends requests to applications acting as servers. See also <strong>client/server</strong>.</td>
</tr>
<tr>
<td><strong>Client-Library</strong></td>
<td>A library of routines that is part of Open ClientConnect. The Open ClientConnect Client-Library comprises a subset of the Open Client Client-Library routines.</td>
</tr>
<tr>
<td><strong>client request</strong></td>
<td>An RPC or language request sent by a client to a server.</td>
</tr>
<tr>
<td><strong>client/server</strong></td>
<td>An architecture in which the client is an application that handles the user interface and local data manipulation functions, and the server is an application providing data processing access and management. See also <strong>client application</strong>.</td>
</tr>
<tr>
<td><strong>Client Services Application</strong></td>
<td>A customer-written CICS program initiated on the host that uses the Sybase API to invoke MainframeConnect for DB2 UDB as a client to DirectConnect or to SQL Server. See also <strong>application program interface</strong>. <strong>Client Services for CICS</strong>.</td>
</tr>
<tr>
<td><strong>Client Services for CICS</strong></td>
<td>A Sybase host API that invokes Open ServerConnect as a client to an access service for DB2 UDB or SQL Server. See also <strong>Application program interface</strong>, <strong>Customer Information Control System</strong>, <strong>Client Services Application</strong>, <strong>Open ServerConnect</strong>.</td>
</tr>
<tr>
<td><strong>commit</strong></td>
<td>A process that makes permanent all changes made to one or more database files since the initiation of the application program, the start of an interactive session, or the last commit or rollback operation. Compare with <strong>rollback</strong>.</td>
</tr>
<tr>
<td><strong>connection</strong></td>
<td>A network path between two systems. For SNA, the path connects a logical unit (LU) on one machine to an LU on a separate machine. For TCP/IP, the path connects TCP modules on separate machines.</td>
</tr>
<tr>
<td><strong>connection router</strong></td>
<td>A program provided with Open ClientConnect that directs requests to particular remote servers. Mainframe system programmers use the connection router to define remote servers and server connections to Open ClientConnect.</td>
</tr>
<tr>
<td><strong>Connection Router Table</strong></td>
<td>A memory-resident table maintained by an Open ClientConnect system programmer that lists servers and the connections that a Client-Library transaction can use to access them.</td>
</tr>
<tr>
<td><strong>conversation-level security</strong></td>
<td>The passing of client login information to the mainframe by TRS when it allocates a conversation.</td>
</tr>
<tr>
<td><strong>CSA</strong></td>
<td>See <strong>Client Services Application</strong>.</td>
</tr>
<tr>
<td><strong>CSP</strong></td>
<td>See <strong>catalog stored procedure</strong>.</td>
</tr>
<tr>
<td>Term</td>
<td>Definition</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>cursor</td>
<td>In SQL, a named control structure used by an application program to point to a row of data.</td>
</tr>
<tr>
<td>Customer Information Control System</td>
<td>An IBM licensed program that enables transactions entered at remote terminals to be processed concurrently by user-written application programs. Open ServerConnect, MainframeConnect for DB2 UDB, and Open ClientConnect are available for CICS.</td>
</tr>
<tr>
<td>database management system</td>
<td>A computer-based system for defining, creating, manipulating, controlling, managing, and using databases.</td>
</tr>
<tr>
<td>database operation</td>
<td>A single action against the database. For DirectConnect, a database operation is usually a single SQL statement. One or more database actions can be grouped together to form a request. See also request.</td>
</tr>
<tr>
<td>Database 2</td>
<td>An IBM relational database management system.</td>
</tr>
<tr>
<td>datatype</td>
<td>A keyword that identifies the characteristics of stored information on a computer.</td>
</tr>
<tr>
<td>DB-Library</td>
<td>A Sybase and Microsoft API that allows client applications to interact with ODS applications. See also application program interface.</td>
</tr>
<tr>
<td>DB2 UDB</td>
<td>See Database 2.</td>
</tr>
<tr>
<td>default language</td>
<td>The language that displays a user’s prompts and messages.</td>
</tr>
<tr>
<td>direct request</td>
<td>A request sent directly from a client workstation through Transaction Router Service to DirectConnect without going through SQL Server. Contract with indirect request.</td>
</tr>
<tr>
<td>direct resolution</td>
<td>A type of service name resolution that relies upon a client application specifying the exact name of the service to be used. See also service name resolution. Compare with service name redirection.</td>
</tr>
<tr>
<td>DirectConnect</td>
<td>A Sybase Open Server application that provides access management for non-Sybase databases, copy management (transfer), and remote systems management. The name replaces the names MDI Database Gateway and OmniSQL Access Module. Compare with Enterprise Connect.</td>
</tr>
<tr>
<td>DirectConnect Manager</td>
<td>A Sybase Windows application that provides remote management capabilities for DirectConnect products. These capabilities include starting, stopping, creating, and copying services.</td>
</tr>
<tr>
<td>DirectConnect for OS/390</td>
<td>A Sybase LAN-based solution that communicates with mainframe host components. It incorporates the functionality of the MDI Database Gateway and the Sybase Net-Library and includes LU 6.2 and TCP/IP support.</td>
</tr>
<tr>
<td><strong>DirectConnect server</strong></td>
<td>The component that provides general management and support functions to service libraries.</td>
</tr>
<tr>
<td>--------------------------</td>
<td>--------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>DirectConnect access service</strong></td>
<td>The named set of properties, used with a DirectConnect Service Library, to which clients connect.</td>
</tr>
<tr>
<td><strong>DirectConnect Service Library</strong></td>
<td>The component that provides a set of functions within the DirectConnect Server environment.</td>
</tr>
<tr>
<td><strong>dll</strong></td>
<td>See <strong>dynamic link library</strong>.</td>
</tr>
<tr>
<td><strong>dynamic link library</strong></td>
<td>A file containing executable code and data bound to a program at load time or runtime, rather than during linking.</td>
</tr>
<tr>
<td><strong>dynamic SQL</strong></td>
<td>The preparation and processing of SQL source statements within a program while the program runs. The SQL source statements are contained in host-language variables rather than being coded directly into the application program. Compare with <strong>static SQL</strong>.</td>
</tr>
<tr>
<td><strong>end user</strong></td>
<td>A person who connects to DirectConnect using an application in order to access databases and perform transfers. See also <strong>transfer</strong>.</td>
</tr>
<tr>
<td><strong>environment variable</strong></td>
<td>A variable that describes how an operating system runs and the devices it recognizes.</td>
</tr>
<tr>
<td><strong>External Security Manager</strong></td>
<td>An add-on security package for the z/OS mainframe, licensed by Computer Associates.</td>
</tr>
<tr>
<td><strong>gateway</strong></td>
<td>Connectivity software that allows two or more computer systems with different network architectures to communicate.</td>
</tr>
<tr>
<td><strong>Gateway-Library</strong></td>
<td>A library of communication, conversion, tracing, and accounting functions supplied with Open ServerConnect.</td>
</tr>
<tr>
<td><strong>host</strong></td>
<td>The mainframe or other machine on which a database, an application, or a program resides. In TCP/IP, this is any system that is associated with at least one Internet address. See also <strong>Transmission Control Protocol/Internet Protocol</strong>.</td>
</tr>
<tr>
<td><strong>host ID</strong></td>
<td>In Open ServerConnect, the ID that the TRS passes to the mainframe with a client request. The host ID is part of the client login definition at the TRS.</td>
</tr>
<tr>
<td><strong>host password</strong></td>
<td>In Open ServerConnect, the password that the TRS passes to the mainframe with a client request. The host password is part of the client login definition at the TRS.</td>
</tr>
<tr>
<td>Glossary</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>host request library</td>
<td>A DB2 UDB table that contains host-resident SQL statements that can be executed dynamically. See also host-resident request.</td>
</tr>
<tr>
<td>host-resident request</td>
<td>A SQL request that resides on MainframeConnect in the host request library. See also host request library.</td>
</tr>
<tr>
<td>IMS</td>
<td>See Information Management System.</td>
</tr>
<tr>
<td>indirect request</td>
<td>A client request that is routed through a stored procedure on a SQL Server, which forwards the request to TRS as an RPC. Compare with direct request.</td>
</tr>
<tr>
<td>Information Management System</td>
<td>A database/data communication system that can manage complex databases and networks.</td>
</tr>
<tr>
<td>interfaces file</td>
<td>An operating system file that determines how the host client software connects to a Sybase product. An interfaces file entry contains the name of any DirectConnect server and a list of services provided by that server.</td>
</tr>
<tr>
<td>language RPC</td>
<td>The name TRS uses to represent a client’s language request. TRS treats a language request as a remote procedure call (RPC) and maps it to a language transaction at the remote server.</td>
</tr>
<tr>
<td>Integrated Product Set (IPS)</td>
<td>The Sybase Integrated Product Set that provides heterogeneous data integration.</td>
</tr>
<tr>
<td>language transaction</td>
<td>The server transaction that processes client language requests. The MainframeConnect for DB2 UDB language transaction is AMD2, which uses the DB2 UDB dynamic SQL facilities to process incoming SQL strings. The OmniSQL Access module for DB2 for IMS and z/OS uses SYRT by default.</td>
</tr>
<tr>
<td>logical unit</td>
<td>A type of network addressable unit that enables a network user to gain access to network facilities and communicate remotely. A connection between a TRS and a CICS region is a connection between logical units.</td>
</tr>
<tr>
<td>logical unit 6.2</td>
<td>A type of logical unit that supports general communication between programs in a distributed processing environment. See also advanced program-to-program communication.</td>
</tr>
<tr>
<td>login ID</td>
<td>In Open ServerConnect, the ID that a client user uses to log in to the system.</td>
</tr>
<tr>
<td>login packet</td>
<td>Client information made available to Open ServerConnect. The client program sets this information in a login packet and sends it to the TRS, which forwards it to the mainframe.</td>
</tr>
<tr>
<td>Term</td>
<td>Definition</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>long-running transaction</strong></td>
<td>A transaction that accepts more than one client request. Whereas short transactions end the communication after returning results to a client, a long-running transaction can await and process another request. Compare with short transaction.</td>
</tr>
<tr>
<td><strong>LU 6.2</strong></td>
<td>See logical unit 6.2.</td>
</tr>
<tr>
<td><strong>mainframe access products</strong></td>
<td>Sybase products that enable client applications to communicate with mainframes in a client/server environment. See client/server.</td>
</tr>
<tr>
<td><strong>Mainframe Connect IPS</strong></td>
<td>The Sybase Integrated Product Set that provides access to mainframe data.</td>
</tr>
<tr>
<td><strong>MainframeConnect for DB2 UDB</strong></td>
<td>A Sybase mainframe solution that provides dynamic access to DB2 UDB data. It replaces the OmniSQL Access Module for DB2 (in CICS only) and the functionality in the MDI Access Server. See also Customer Information Control System, Database 2, Multiple Virtual Storage.</td>
</tr>
<tr>
<td><strong>Multiple Virtual Storage</strong></td>
<td>An IBM operating system that runs on most System/370 and System/390 mainframes. It supports 24-bit addressing up to 16 megabytes.</td>
</tr>
<tr>
<td><strong>Net-Gateway</strong></td>
<td>A Sybase product that provides communication between a mainframe and a LAN server. Net-Gateway is the predecessor of the DirectConnect TRS.</td>
</tr>
<tr>
<td><strong>network protocol</strong></td>
<td>A set of rules governing the way computers communicate on a network.</td>
</tr>
<tr>
<td><strong>null</strong></td>
<td>Having no explicitly assigned value. NULL is not equivalent to 0 or to blank.</td>
</tr>
<tr>
<td><strong>ODBC</strong></td>
<td>See Open Database Connectivity.</td>
</tr>
<tr>
<td><strong>ODS</strong></td>
<td>See Open Data Services.</td>
</tr>
<tr>
<td><strong>OmniConnect</strong></td>
<td>A variation of Sybase ASE Server that provides a Transact-SQL interface to various sources of external data. The name replaces the names OmniSQL Gateway and OmniSQL Server. The CIS functionality of ASE has incorporated the functionality of OmniConnect and is referred to as ASE/CIS. See ASE/CIS.</td>
</tr>
<tr>
<td><strong>OmniSQL Access Module for DB2</strong></td>
<td>A Sybase mainframe solution that provides access to DB2 data. It is the predecessor of MainframeConnect for DB2 UDB.</td>
</tr>
<tr>
<td><strong>Open Client</strong></td>
<td>A Sybase product that provides customer applications, third-party products, and other Sybase products with the interfaces required to communicate with Open Client and Open Server applications.</td>
</tr>
</tbody>
</table>
Open ClientConnect  A Sybase product that allows mainframe clients to send requests to SQL Server, Open Server, MainframeConnect for DB2 UDB and Open ServerConnect, using Client-Library. Open ClientConnect provides capability for the mainframe to act as a client to LAN-based resources.

Open ClientConnect for CICS/MVS  The Sybase capability for the mainframe to act as a client to LAN-based resources in the CICS environment.

Open ClientConnect for IMS™ and MVS  The Sybase capability for the mainframe to act as a client to LAN-based resources in the IMS™ and native MVS environments.

Open Data Services  A product that provides a framework for creating server applications that respond to DB-Library clients.

Open Database Connectivity  A Microsoft API that allows access to both relational and non relational databases. See also application program interface.

Open Server  A Sybase product that provides the tools and interfaces required to create a custom server. Clients can route requests to DirectConnect through an Open Server configured to meet specific needs, such as the preprocessing of SQL statements.

Open ServerConnect  A Sybase product that provides capability for programmatic access to mainframe data. It allows workstation-based clients to execute customer-written mainframe transactions remotely. See also Gateway-Library.

Open ServerConnect for CICS/MVS  The Sybase capability to provide programmatic access to mainframe data in the CICS environment.

Open ServerConnect for IMS™ and MVS  The Sybase capability to provide programmatic access to mainframe data in the IMS™ and native MVS environments.

Parameter  A variable that is given a constant value for a specified application and can denote the application. Compare with property.

Password Expiration Management  An IBM password management program with CICS Version 3.3 through an optional program temporary fix, and as an integral part of CICS with version 4.1 and higher.

PEM  See Password Expiration Management.

PL/1  See Programming Language/1.

Programming Language/1  A programming language designed for use in a wide range of commercial and scientific computer applications.
property

A setting for a server or service that defines the characteristics of the service, such as how events are logged. Compare with parameter.

protocol

The rules for requests and responses used to manage a network, transfer data, and synchronize the states of network components.

query

A request for data from a database, based upon specified conditions.

Registry

The part of the Windows operating system that holds configuration information for a particular machine.

relational database

A database in which data is viewed as being stored in tables consisting of columns (data items) and rows (units of information).

remote procedure call

A call to execute a stored procedure on a remote server. For Open ServerConnect, an RPC is a direct request from a client to TRS. For Open ClientConnect, a Client-Library transaction that calls a procedure on a remote server acts like an RPC.

remote stored procedure

A customer-written CICS program that resides on the mainframe and communicates with MainframeConnect for DB2/MVS. See also Customer Information Control System, stored procedure. Compare with Client Services Application.

remote systems management

A feature that allows a system administrator to manage multiple DirectConnect servers and multiple services from a client.

Replication Server

A Sybase SQL Server application that maintains replicated data and processes data transactions received from a data source.

request

One or more database operations an application sends as a unit to the database. Depending upon the response, the application commits or rolls back the request. See also commit, rollback, unit of work.

rollback

An instruction to a database to back out of changes requested in a unit of work. Compare with commit.

RPC

See remote procedure call.

RSP

See remote stored procedure.

server

A functional unit that provides shared services to workstations over a network. See also client/server. Compare with client.

service

A functionality available to DirectConnect applications. It is the pairing of a service library and a set of specific configuration properties.
<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>service library</td>
<td>In DirectConnect applications, a set of configuration properties that determine service functionality. See also access service library, administrative service library, transaction router service library, transfer service library.</td>
</tr>
<tr>
<td>service name redirection</td>
<td>A type of service name resolution that allows a system administrator to create an alternative mechanism to map connections with services. See also service name resolution. Compare with direct resolution.</td>
</tr>
<tr>
<td>service name resolution</td>
<td>The DirectConnect server mapping of an incoming service name to an actual service. See also direct resolution, service name redirection.</td>
</tr>
<tr>
<td>session</td>
<td>A connection between two programs or processes. In APPC communications, sessions allow transaction programs to have conversations between the partner LUs. See also advanced program-to-program communication.</td>
</tr>
<tr>
<td>short transaction</td>
<td>A mainframe transaction that ends the communication when it finishes returning results to the client. Compare with long-running transaction.</td>
</tr>
<tr>
<td>SNA</td>
<td>See Systems Network Architecture.</td>
</tr>
<tr>
<td>SQL</td>
<td>See structured query language.</td>
</tr>
<tr>
<td>sql.ini</td>
<td>The interfaces file containing definitions for each DirectConnect server to which a workstation can connect. The file must reside on every client machine that connects to SQL Servers.</td>
</tr>
<tr>
<td>SQL Server</td>
<td>The server in the Sybase Client-Server architecture. It manages multiple databases and users, tracks the actual location of data on disks, maintains mapping of logical data description to physical data storage, and maintains data and procedure caches in memory.</td>
</tr>
<tr>
<td>static SQL</td>
<td>SQL statements that are embedded within a program and prepared during the program preparation process before the program runs. Compare with dynamic SQL.</td>
</tr>
<tr>
<td>stored procedure</td>
<td>A collection of SQL statements and optional control-of-flow statements stored under a particular name. Sybase SQL Server stored procedures are called “system procedures.” See also remote stored procedure, system procedure.</td>
</tr>
<tr>
<td>structured query language</td>
<td>An IBM industry-standard language for processing data in a relational database.</td>
</tr>
<tr>
<td>SYRT</td>
<td>The component of OmniSQL Access Module for DB2 that allows clients to submit SQL language requests to DB2 through IMS TM or MVS (z/OS).</td>
</tr>
<tr>
<td>Term</td>
<td>Definition</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>system procedures</td>
<td>A stored procedure that Sybase SQL Server supplies for use in system administration. System procedures serve as shortcuts for retrieving information from system tables, or a mechanism for accomplishing database administration. See also <em>stored procedure</em>.</td>
</tr>
<tr>
<td>Systems Network Architecture (SNA)</td>
<td>An IBM proprietary plan for the structure, formats, protocols, and operational sequences for transmitting information units through networks. See also <em>advanced program-to-program communication</em>.</td>
</tr>
<tr>
<td>table</td>
<td>An array of data or a named data object that contains a specific number of unordered rows. Each item in a row can be unambiguously identified by means of one or more arguments.</td>
</tr>
<tr>
<td>Tabular Data Stream</td>
<td>A Sybase application-level protocol that defines the form and content of relational database requests and replies.</td>
</tr>
<tr>
<td>target</td>
<td>A system, program, or device that interprets, rejects, satisfies, or replies to requests received from a source.</td>
</tr>
<tr>
<td>TCP/IP</td>
<td>See <em>Transmission Control Protocol/Internet Protocol</em>.</td>
</tr>
<tr>
<td>TDS</td>
<td>See <em>Tabular Data Stream</em>.</td>
</tr>
<tr>
<td>transaction</td>
<td>A unit of processing initiated by a single request. A transaction consists of one or more application programs that, when executed, accomplish a particular action. In Open ServerConnect, a client request (RPC or language request) invokes a mainframe transaction. In Open ClientConnect, a mainframe transaction executes a stored procedure on a remote server.</td>
</tr>
<tr>
<td>transaction processing</td>
<td>A sequence of operations on a database that is viewed by the user as a single, individual operation.</td>
</tr>
<tr>
<td>Transaction Router Service</td>
<td>The DirectConnect program that accepts requests from workstation-based clients and routes them to Open ServerConnect.</td>
</tr>
<tr>
<td>Transaction Router Service Library</td>
<td>A service library that facilitates access to remote transactions, allowing customers to execute transactions from virtually any mainframe data source. See also <em>service library</em>.</td>
</tr>
<tr>
<td>transfer</td>
<td>A DirectConnect feature that allows users to move data or copies of data from one database to another.</td>
</tr>
<tr>
<td>Transfer Service Library</td>
<td>A service library that provides copy management functionality. See also <em>service library</em>.</td>
</tr>
</tbody>
</table>
Glossary

Transmission Control Protocol/Internet Protocol
A set of communication protocols that supports peer-to-peer connectivity functions for both local and wide area networks.

TRS
See Transaction Router Service.

TRS Library
See Transaction Router Service Library.

unit of work
One or more database operations grouped under a commit or rollback. A unit of work ends when the application commits or rolls back a series of requests, or when the application terminates. See also commit, rollback, transaction.

UNIX
An operating system that allows for multiple concurrent programs and users.

user ID
User identification. The ID number by which a user is known in a specific database or system.

variable
An entity that is assigned a value. DirectConnect has two kinds of variables: local and global.

Virtual Storage Access Method
An IBM licensed program that controls communication and the flow of data in an SNA network.

Virtual Telecommunications Access Method
IBM mainframe software that allows communication on an SNA network between mainframes and allows the mainframe to have multiple sessions per connection.

VSAM
See Virtual Storage Access Method.

VTAM
See Virtual Telecommunications Access Method.

Windows New Technology
A multi-tasking operating system from Microsoft Corporation.

workstation
A terminal, micocomputer, or personal computer, usually one that is connected to a mainframe or to a network, at which a user can perform tasks.
Index

Symbols
% (percent sign) as a wildcard 64
%SYBASE% environment variable 17
(double quotes)
    with parameter values 62
@ (at symbol)
    for named parameters 63
    for escape character 65

A
AccountFile
    configuration property 21
accounting
    activating 133
    reading the log 133
    status of 141
activate
    accounting 133
    connection 126
    region 128
    RPC 129
    tracing 131
add
    catalog RPCs 58
    connection 46
    connection group 105
    connection to connection group 107
    region 48
    RPC 50
    RPC to transaction group 112
    task table 39
    transaction group 111
add an LU 6.2 connection 47
administration
    permission 102
Administrative Service Library 4
ADMLIB 4
aggregate
    handling 66
AL
    transaction status 137
all option
    restarting all regions 128
    restarting connections 126
allocation 137
AMD2
    description 53
    AMD2 transaction
        request size 144
AND predicates 66
API
    Open ClientConnect 7
APPLID
    name 46
at symbol (@)
    for escape character 65
    for named parameters 63
availability
    connection 138
    RPC 139

B
batch administration commands
    TRS 37
buffer count 137
bulk insert handling 66

C
CASE support 68
catalog stored procedures 61, 91
coding 62, 64
coding examples 63
CSP parameters and DB2 64
Index

escape character 65
overview 61
parameters 62
sp_column_privileges 68
sp_columns 70
sp_databases 73
sp_datatype_info 74
sp_fkeys 76
sp_pkeys 78
sp_server_info 80
sp_special_columns 81
sp_server_info 80
sp_sproc_columns 83
sp_statistics 85
sp_table_privileges 88
sp_tables 90
supported CSPs 65
syntax 62
table_name parameter 63
table_owner parameter 63
table_qualifier parameter 63
wildcards 64
change
task table 41
transaction group 114
char set
data flag 140
character set 157
character truncation 67
CICS
listener 152, 153
security example 52, 153
client
deleting definition 103
deleting login 103
disconnect 129
login to transaction group 109
machine name 136
maximum number 140
number 136, 138
requesting the transaction called 50
requesting through SQL Server 95
status of TRS 136
client login
information file 24
Client Services Application (CSA)

Open ClientConnect 7
client_number parameter 129
ClientIdleTimeout
configuration property 34
client-level security 100
Client-Library
calls not supported 147, 148
command conventions
TRS 37, 38
command line
procedures 37
commands
gwg_help 39
commas
TRS 38
TRS security 99
con_group parameter 102
con_name parameter
addcongrp procedure 107
gwg_addcon procedure 46, 150
gwg_dropconfromgrp procedure 107
con_number parameter 126
configuration file
inging 11
format 13
ample 12
configuration property
AccountFile 21
ClientIdleTimeout 34
ConnInfoFile 22
ConQTimeout 23
DeactCon 23
description 24, 35
DirectPrevent 24, 95
displaying TRS Library settings 140
EnableAtStartup 35
LogInfoFile 24
LogTRS 25
MaxConnections 25
PEMDest 26
reference 18, 34
RegionInfoFile 28
RPCInfoFile 29
security 29
Send5701 30
TDSTraceFile 30
TraceTRS 31
TruncateLV 32
UpgradePassword 32
UpperCase 33
UseDBRPC 33
XNL 36
XNLVarChar 36
configuring
TRS 11
TRS for MainframeConnect 53
connection
activating 125
adding to connection group 107
adding to TRS 46
availability 138
deactivating 126
dedicating 45
defining 45
deleting 47
deleting from connection group 107
dropping 47
file name 141
inactive 126
name 138
number 136, 138
region 51, 153
restarting 125
seconds in queue 141
status 137, 138
testing TRS for LU 6.2 150
testing TRS for TCP/IP 151
connection group
adding 105
assigning a user 101
assigning login 98
connection-level security 105
correction-level security 105
defining 105
defining to user 102
deleting 108
deleting connection 107
connection queue
mainframe 137
connection-level security 105
ConnInfoFile
configuration property 22
ConQTimeout
configuration property 23
contention winner
parallel sessions 46
control
permission 102
correction allocated 137
correction-level security 104
count buffers 137
CQ transaction status 137
CSKL transaction 152
CSP
adding 58
dropping 59
installing 58, 59
scripts 58
see catalog stored procedures 61
testing 59
D
data truncation setting 140
datatype
long varchar 32
date functions 67
DB2
accessing 144
DB-Library
unsupported calls 146, 147
dbrpcinit statement
RPC name 136
DeactCon
configuration property 23
deactivate
accounting 133
connection 127
region 128
RPC 130
define
connection 46
connection group 105
login information 101
region 48
RPC 50
user 100
Index

delete
  connection 47
  connection from connection group 107
  connection group 108
  connections 48
  region 49
  RPC 53
  RPC from transaction group 113
  user 103
destination subsystem. See region 37
Destination_Service_Library
  parameter 17
Destination_Service_Library parameter 17
direct access to TRS
  preventing 24
direct requests
  preventing 24
  sending RPCs 145
direct RPCs 140
DirectConnect Manager
  description 5
directory structure
  locales 17
DirectPrevent configuration property 24, 95
disconnect
  idle clients 129
display
  connection group 105
  login information 101
  task table 41
  TRS command results 38
distinct option 67
draining
  connection 138
drop
  catalog RPCs 59
  connection 47
  region 49
  RPC 52
  task table 39
  transaction group 115
dropcat script
  catalog RPCs 59

E
EnableAtStartup
  configuration property 35
environment variables
  %SYBASE% 17
  LC_ALL 159
  LC_CTYPE 159
  LC_MESSAGE 159
  LC_TIME 160
error files
  TRS 31
error log 130
examples
  activating a connection (LU 6.2) 126
  activating a single region 128
  activating an RPC 130
  add an RPC 52
  adding a login definition to TRS 101
  adding a user to an LU 6.2 TRS 102
  adding an LU 6.2 connection 47
  adding RPCs to a tran_group 55
  changing passwords 103
  creating a transaction group 112
  deactivating a connection (LU 6.2) 127
  deactivating a region (TCP/IP) 128
  deactivating an RPC 130
  defining the connection (LU 6.2 only) for Windows NT 151
  defining the test region (TCP/IP only) 152
  defining the test RPCs 153
  deleting a transaction group 115
  deleting connections 48
  deleting RPC names from a transaction group 114
  disconnecting a client 129
  displaying existing logins 101
  displaying one transaction group 110
  dropping a region 49
  dropping an RPC 53
  dropping connections from a connection group 107
  dropping CSPs 59
  modifying a transaction group 114
  output from running the test RPC 154
  removing a user from the TRS login list 103
  specifying IDs for the mainframe 113
  testing CSPs 59
TRS configuration file 12
execute administration procedure
   TRS 38
   execute command
      catalog stored procedures and system procedures 62
      syntax 145
      TRS 37
expression handling 67

F
FMH-5 95
force option
   deactivating connection 127
Function Management Header 5 51, 95, 153

G
gateway control permission 102
gateway parameter 99
GC transaction status 137
group by 66
group connection 102
   adding 105
   adding a connection 107
   assigning a user 101
   assigning login 98
   connection-level security 105
   conversation-level security 105
   defining connection 105
ID 99
group parameter value
   RPC 113
GROUP_LOGIN parameter 111
group_name parameter
   sgw_addcongrp procedure 107
   sgw_dropconfromgrp procedure 107
   sgw_dropcongrp procedure 108
   sgw_dspcongrp procedure 105
GROUP_PWD parameter
   sgw_addtrngrp procedure 111
gwctrl parameter 102

H
help
   sgw_help command 39
host
   TCP/IP name 49, 139, 152
   transaction name 136, 138
Host Name
   status field 139
HOST_LOGIN parameter 101
HOST_PWD parameter
   sgw_addlog procedure 101
   sgw_chpwd procedure 103
Host_Tran
   status field 139
hostname
   parameter 49, 152

I
ID transaction status 137
IDENTIFY
   CICS 52
IDENTIFY, CICS 153
idle connection 34
idle transaction status 137
IN/NOT IN support 68
inactive connection
   dropping 47
   preventing 126
indirect access to TRS
   routing through SQL server 24
indirect RPCs 140
initializing
   user exit 166
insert/select handling 68
installation
   catalog RPCs 58
   test for TRS 150, 154
installing
   user exit handlers 169
interfaces file 14
   name 141
   service name 14
isql commands
   TRS administration procedures 37
Index

IT transaction status 137

J
join handling 66

L
langpwdlevel parameter 111, 112
langrpc parameter 111
language
  defining RPCs 108
  displaying password source 111
  displaying the handler 111
  login level for RPC 112
  maximum request size 143
  national 140
  RPC request name 111
  transaction 108
language events 67
level
  login ID 109
  transaction login ID 113
library calls
  unsupported 146, 148
LIKE predicates 66
Listener Transaction 51
loading
  user exit 166
locale name 157
locales
  directory 17
  file 157
locales.dat 162
localization
  .loc files 162
  character set files 162
Client-Library or Server-Library 163
conversion between client and server 158, 159
default values 163
defined 157
files 160, 162
locales file 162
locales name 157

login
  adding TRS 101
  changing 103
  client 100
  client name 136
  defining name to TRS 101
  definition 98, 100
  deleting 103
  displaying 101
  region 112
  RPC 51, 153
  system administrator 99
  transaction group 105, 109, 111
login level parameter value
  group 112
  none 112
  user 112
login parameter
  sgw_addlog procedure 101
  sgw_chpwd procedure 99, 103
  sgw_droplog procedure 103
LogInfoFile
  configuration property 24
LogTRS
  configuration property 25
long varchar datatype
  truncating 32
  truncation flag 140
long-running transaction
  client disconnect 129
LU
  remote connection definition 46, 151
LU 6.2
  connections per application 45
  security role 95

M
machine name for client 136
mainframe
  access permission 139
  TCP/IP name 49, 139, 152
  transaction name 136, 138, 139
MainframeConnect
  description 6

Mainframe Connect DirectConnect for z/OS Option
Index

overview 1
requests 143, 146
math functions 67
max_sessions parameter 47, 151
MaxConnections configuration property 25
maximum
        sessions 47, 151
maximum bytes 144
mode
        define to connection 47, 151
        name 138
        parameter 151
    mode parameter 47
modify
        passwords 102, 104
        transaction group 114

N
N RPC
        security field value 139
name
        displaying login 101
        language RPC 108
        region 49, 50
        RPC 139, 152
        TRS 140
national language 140
net password encryption 66
Net-Gateway start-up parameter
        -C 21
        -D 19
        -d 19
        -E 21
        -G 19
        -K 19
        -L 19
        -M 19
        -m 20
        -O 20
        -Q 19
        -R 20
        -s 21
        -T 20
        -t 20
-u 20
-V 20
NIS map 49
null administration procedures
        TRS 38
number connection 136, 138
numerical values
        TRS 38

O
object case sensitivity 66
ODBC
        datatypes 71
Open Client
        Open ClientConnect 7
        Open ClientConnect
description 7
Open Server
        Open ServerConnect 7
Open ServerConnect
        APIs 7
        description 7
        migration information 7
        predecessors 7
        RSPs 7
OR predicates 66
order by option 66
override TRS security 141

P
-P Net-Gateway start-up parameter
    -P 19
parallel sessions
        maximum sessions 151
        shared connection 46
parameters
        CSPs and system procedures 62
        RPC 144
        stored procedure for choosing multiple DB2s 56
        TRS administration procedures 38
passthrough security
        overriding 99
Index

password
  changing client 103
  defining client 100
  defining host 101
group 99
RPC 52
  system administrator’s account 99
user 101
PEM RPCs
  sgw_addlog 120, 123
  sgw_addtg 122, 123
  sgw_pemchgrp 121, 122, 123
  sgw_pemchpwd 120, 121, 123
  sgw_peminfogrp 120
  sgw_peminfopwd 120
PEMDest
  configuration property 26
percent sign (%) as a wildcard 64
permission
  administration 102
  connection 105
  mainframe requirements 139
transaction 108
port
  status field 139
portnumber
  parameter 49, 152
  TRS 49, 152
POSIX localization
  environment variables 159
procedures
  TRS administration 37
property values
  modifying 14
  protocol type 140
pwd parameter 101, 103

Q
queue
  connection 137
  connection wait 141
Quick-Start to configuring TRS 43
quotation marks
  TRS numerical values 38

R
reading server transaction status 137
record accounting information 132
recovery
  connections 126
  regions 128
region
  activating 128
  availability 139
  connection status 138
  deactivating 128
  defining 48
  dropping 49
  file name 141
  parameter 50, 51, 151
  restarting 128
  RPC status 139
  status 138, 139
region parameter 46
  sgw_addcon procedure 46, 151
  sgw_addrpc procedure 153
RegionInfoFile
  configuration property 28
Remote LU
  connection definition 46, 151
Remote Stored Procedures (RSPs)
  Open ServerConnect 7
REMOTE_DATATYPE value 73
requests
  direct 24
  indirect 145
  sending to TRS 143, 146
results
  TRS administration procedures 38
  TRS waiting 137
routing RPCs 95, 140
RPC
  activate 129
  addcat script 58
  adding 50
  adding to transaction group 112
  deactivating 130
  defining 104
  defining to transaction group 108
  defining to TRS 50
deleting 53

198
Index

deleting from transaction group 113
direct or indirect 140
displaying password level 110
name 136, 139
parameter size 144
routing through SQL Server 95
security 51, 52, 153
security field value 139
security status 139
sending 144
sending to mainframe 144, 146
status 139
status field 139
test for TRS 152
transaction group for language 111
RPC examples 53
RPC option
display transaction group 110
RPC parameter value
both 51, 153
none 51, 113, 153
userid 51, 153
rpc_name parameter 50, 113, 152
RPCInfoFile
configuration property 29
rpcpwdlevel parameter 113
RS transaction status 137

S
sa account 99
samples
Transaction Router 156
samples, Transaction Router 155
scripts
catalog RPCs 58
security
configuration property 29
connection-level 105
conversation-level 104
enforced at TRS 141
fields 139
file name 141
mainframe 95
not enforced at TRS 93, 99, 108, 141
override 29
overview 93, 100
RPC definition 51, 153
sa privileges 99
source RPC 52
SQL Server 95
status 141
transaction-level 108
TRS configuration property 94
user-level 100
security group
information file 24
security parameter
RPC definition 51, 104, 153
security passthrough
overriding 99
select statement
multiple DB2s 55
Send5701
configuration property 30
server name
TRS 140
service library name 13
services
creating additional 15
sessions
maximum per connection 47, 151
multiple per independent LU 47, 48
sgw prefix 37
sgw_actcon procedure 126
sgw_actregion procedure 128
sgw_actrpc procedure 129
sgw_add procedure 50
sgw_addcon procedure 46, 150
con_name parameter 46
region parameter 46
sgw_addcongrp procedure 106
sgw_addcontogrp procedure 107
sgw_addlog 120, 123
sgw_addlog procedure 101
sgw_addregion procedure 49, 152
sgw_addrpc procedure 50, 51, 152
sgw_addrpcstogrp procedure 112
sgw_addtmgrp 122, 123
sgw_addtrngrp procedure 111
sgw_chpwd procedure 99, 103

User's Guide for Transaction Router Services 199
Index

sgw_deactcon procedure 127
sgw_deactregion procedure 128
sgw_deactrpc procedure 130
sgw_disclient procedure 129
sgw_dropcon procedure 47
sgw_dropconfromgrp procedure 107
sgw_dropcongrp procedure 108
sgw_droplog procedure 103
sgw_dropregion procedure 49
sgw_droprpc procedure 53
sgw_droprpcfromgrp procedure 113
sgw_dspact procedure 133
sgw_dspcongrp procedure 105
sgw_dsplog procedure 101
sgw_dsptrngrp procedure 110
sgw_help command 39
sgw_modtmgrp procedure 114
sgw_pemchgrpwpwd 121, 122, 123
sgw_pemchpwpd 120, 121, 123
sgw_peminfogrpwpwd 120
sgw_peminfopwpd 120
sgw_shutdown parameter 133
sgw_status
   clients procedure 136
   connections procedure 137
   parameters procedure 140
   region procedure 138
   rpc procedure 139
   trace procedure 141
sgw_stopact procedure 133
sgw_stoptrace procedure 132
SH transaction status 137
shutdown
   TRS 133
site handler
   maximum allowed 140
   transaction status 137
SNA network
   connection name 138
   passing login information 52
   recovering the connection 126
   recovering the region 128
socket allocated 137
Softlink
   options 17
   Source_DirectConnect parameter 17
   Source_Service_Library parameter 17
sp_addserver procedure 137
sp_capabilities
   result set 66
sp_capabilities system procedure 66
sp_char_length
   system procedure 169, 171
sp_column privileges catalog stored procedure 68
sp_column_privileges catalog stored procedure
   result set 69
sp_columns catalog stored procedure 70
   ODBC datatypes 71
   REMOTE_DATATYPE column 73
   result set 71
sp_databases catalog stored procedure 73
   result set 74
sp_datatype_info catalog stored procedure
   result set 75
sp_fkeys catalog stored procedure 76
   result set 78
sp_pkeys catalog stored procedure 78
   result set 80
sp_server_info catalog stored procedure 80
   result set 81
sp_special_columns catalog stored procedure 81
   result set 82
sp_sproc_columns catalog stored procedure
   result set 83
sp_statistics catalog stored procedure 85
   result set 86
sp_stored_procedures catalog stored procedure
   result set 87
sp_table_privileges catalog stored procedure 88
   result set 89
sp_tables catalog stored procedure 90
   result set 91
sp_thread_props system procedure 92
   SPID status field 137
SQL compatibility 144
SQL Server
   client 137
   routing requests 95, 140, 143
   RPC name for stored procedure 136
   security 95

200

Mainframe Connect DirectConnect for z/OS Option
Index

sending requests 24
site handler 137
SQL syntax capability with sp_capabilities 66
SQLColumnPrivileges 68
SQLColumns 70
SQLForeignKeys 77
SQLGetInfo 80
SQLGetTypeInfo 74
SQLPrimaryKeys 79
SQLProcedureColumns 83
SQLProcedures 87
SQLSpecialColumns 81
SQLStatistics 85
SQLTablePrivileges 89
SQLTables 91
start
accounting 133
connection 125
tracing 131
state
transaction status 137
status
connection 137, 138
field 139
region 138, 139
RPC 139
task table 41
trace 141, 142
TRS 135, 142
stop
accounting 133
tracing 132
TRS 133
stored procedure 56
string functions 67
subquery handling 68
syntax
executing catalog stored procedures and system procedures 62
SYRT
description 53
system administrator account 99
System procedures
sp_thread_props 92
system procedures
coding 62, 64
coding examples 63
escape character 65
parameters 62
sp_capabilities 66
sp_char_length 169, 171
syntax 62
wildcards 64
system procedures parameters
property_name 92
property_value 92
T
table_name CSP parameter 63
table_owner CSP parameter 63
table_qualifier parameter 63
TCP/IP
network host name 49, 152
security role 95
TCP/IP listener 49
TDS
tracing 20
TDSTraceFile
configuration property 30
testcat script
catalog RPCs 59
testing
TRS samples 156
testing TRS samples 155
text and image handling 66
text pattern handling 66
trace
activating 131
status 141, 142
TraceTRS
configuration property 31
tran_group parameter
sgw_addlog procedure 101
sgw_addrpctogroup procedure 112
sgw_addtrngrp procedure 111
sgw_dsptngrp procedure 110
tran_id parameter 50, 153
transaction
client disconnect 129
handling 66

User’s Guide for Transaction Router Services 201
Index

idle long-running 137
language handler 108
long-running 129
state 137
time running 137
transaction group
  adding 111
  adding RPCs to 112
  assigning to a user 101
  changing 114
  defining 111
  deleting 115
  deleting RPCs 113
  listing RPCs 98
  login 109
  transaction-level security 110
transaction ID 50, 153
transaction name 136
transaction processing region
  alternate login 111
  defining connection 45
Transaction Router samples 149, 156
transaction status 137
TRS
  command conventions 37, 38
  commas 38
  configure for Quick-Start 43
  configuring for MainframeConnect 53
  controlling 125
  creating additional 15
  description 4
  error files 31
  execute administration peocedure 38
  execute command 37
  null administration procedures 38
  numerical values 38
  quick reference administration procedures 39
  quick reference to administration procedures 42
  scroll administration procedure results 38
  security 93, 115
  stopping 133
  view command results 38
TRS administration commands
  enter 37
TRS administration procedures
  isql commands 37
  parameters 38
  results 38
TRS batch administration commands 37
TRS library
  configuration file format 13
  configuration file sample 12
  configuration properties 18
  configuration property reference 34
  modifying configuration file values 14
TRS procedures
  administration 37
  task tables 39, 42
  trscopy utility 15
  truncate longvarchar data flag 140
  TruncateLV configuration property 32
T-SQL
  convert functions 67
  delete/update 67

U
-U flag, isql 101
U RPC
  security field value 139
  unattended TRS 126
  union handling 67
  unsupported calls 146
  UpgradePassword
    configuration property 32
  UpperCase
    configuration property 33
  UseDBRPC
    configuration property 33
user
  client number 136
  defining 100
  deleting 103
  login name for transaction group 109
  maximum allowed 140
user definition
  deleting 103
user Id
  defining 101
  RPC value 52
user parameter value
Index

RPC 113
utilities
  trscopy 15

V
variables
  %SYBASE% 17
VERIFY, CICS security option 52, 153
version
  TRS 140
view command results
  TRS 38
VTAM
  APPLID 153
    APPLID region parameter 46, 151, 153
    deactivate connection 127

W
WA transaction status 137
waiting transaction status 137
WC transaction status 137
wildcard
  escape 67
  examples 64
writing transaction status 137
Index