Release Bulletin Mainframe Connect Server Option for CICS Version 12.6

Document ID: DC75200-01-1260-01

Last revised: May 12, 2005

Topic	Page
1. Accessing current release bulletin information	2
2. Product summary	2
2.1 Product name changes	2
2.2 Hardware and software requirements	3
2.3 Product media	4
2.4 Product documentation	4
3. Changed functionality in this version	4
3.1 Compiler upgrade	5
3.2 SYGWMAP exit	5
3.3 Abend handler	9
3.4 Client Option and Server Option application programs relink	10
3.5 Updated sample programs	11
3.6 Unicode support	11
3.7 Text and image data	17
3.8 Listener handling of concurrent connection request	27
4. Known issues	28
4.1 InstallShield license-key strings	28
4.2 Decimal loss in TDCONVRT money-to-char conversion	28
4.3 InstallShield wizard temporary space requirement	29
5. Product compatibilities	29
6. Technical support	29

Topic	Page
7. Other sources of information	29
7.1 Sybase certifications on the Web	30
7.2 Sybase EBFs and software maintenance	31

1. Accessing current release bulletin information

A more recent version of this release bulletin may be available on the Web. To check for critical product or document information added after the release of the product CD, use the Sybase® Technical Library Product Manuals Web site.

❖ **Accessing release bulletins at the Technical Library Product Manuals Web site**

1 Go to Product Manuals at http://www.sybase.com/support/manuals/.

2 Follow the links to the appropriate Sybase product.

3 Select the Release Bulletins link.

4 Select the Sybase product version from the Release Bulletins list.

5 From the list of individual documents, select the link to the release bulletin for your platform. You can either download the PDF version or browse the document online.

2. Product summary

Enclosed is the Mainframe Connect Server Option for CICS 12.6, which is a programming environment that enables you to develop mainframe applications that Open Client™ applications can execute. Mainframe-based Server Option™ applications can retrieve and update data stored on an IBM mainframe in any mainframe resource, such as VSAM files, TD queues, TS queues, and DL/1 databases, as well as in DB2 databases and other database management systems (DBMSs).

The Server Option for CICS runs on an IBM z/Series or plug-compatible mainframe computer. It uses the LU 6.2 or TCP/IP communications protocol and a CICS host transaction processor.

2.1 Product name changes

The following table describes new names for products in the 12.6 release of the Mainframe Connect IPS.

Old product names	New product name
• Open ClientConnect™ for CICS • Open ClientCONNECT for CICS	Mainframe Connect Client Option for CICS
• Open ClientConnect for IMS and MVS • Open ClientCONNECT for IMS and MVS	Mainframe Connect Client Option for IMS and MVS
• Open ServerConnect™ for CICS • Open ServerCONNECT for CICS	Mainframe Connect Server Option for CICS
• Open ServerConnect for IMS and MVS • Open ServerCONNECT for IMS and MVS	Mainframe Connect Server Option for IMS and MVS
• MainframeConnect™ for DB2 UDB • MainframeCONNECT for DB2/MVS-CICS	Mainframe Connect DB2 UDB Option for CICS
• DirectConnect™ for OS/390 • DirectCONNECT for DB2/MVS	Mainframe Connect DirectConnect for z/OS Option

The new product names are used throughout this document.

2.2 Hardware and software requirements

The following hardware and software are compatible with the Server Option for CICS 12.6:

- Hardware:
  - IBM mainframe: z/Series or plug-compatible
- Software:
  - IBM z/OS version 1.4 or later

    Note The Server Option for CICS works with earlier z/OS releases that are no longer supported by IBM.

  - CICS Transaction Server 1.3 or later
  - IBM TCP/IP

For planning, installation, and configuration information, see the Mainframe Connect Server Option for CICS *Installation and Administration Guide*.

2.3 Product media

The following table lists the Server Option for CICS 12.6 distribution media.

Table 1: Server Option for CICS 12.6 media

Media title	Media ID
Mainframe Connect Server Option for CICS 12.6	CD68187-55-1260-01
Mainframe Connect 12.6 Technical Library CD	CD00222-55-1260-01

Note For directory and file information, see the *CONTENTS* member of the *JCL* data set for your Server Option 12.6 installation.

2.4 Product documentation

The following table lists all documentation for the Server Option for CICS 12.6. Although not all documents are shipped as paper copy, all documents are available on the Web and on the Technical Library CD or the SyBooks™ CD.

Table 2: Server Option for CICS 12.6 documentation

Document title	Document ID
Mainframe Connect Server Option for CICS *Installation and Administration Guide*	DC36510-01-1260-01
Mainframe Connect Server Option *Programmer's Reference for COBOL*	DC36520-01-1260-01
Mainframe Connect Server Option *Programmer's Reference for PL/1*	DC36560-01-1260-01
Mainframe Connect Server Option *Programmer's Reference for Remote Stored Procedures*	DC35605-01-1260-01
Mainframe Connect Client Option and Server Option *Messages and Codes*	DC36450-01-1260-01
Mainframe Connect Server Option for CICS *Release Bulletin*	DC71770-01-1260-01

3. Changed functionality in this version

For information on new features and functionality in the Server Option for CICS 12.6, see the Mainframe Connect IPS *New Features* (DC00182-01-1260-01).

3.1 Compiler upgrade

The Client Option and Server Option are now built with the IBM LE/370 C compiler. These products are now compatible with the IBM Language Environment. The compiler used in the previous release, the IBM V2.1 C compiler, is no longer supported.

3.2 SYGWMAP exit

The Server Option for CICS provides a new modifiable exit routine, SYGWMAP, for dynamically changing the name of the default remote procedure called in response to a client language request. The Server Option context handler then uses functionality coded in the SYGWMAP exit to map a language request to a CICS transaction.

The SYGWMAP exit can be used only for language requests in a two-tier environment. The Server Option context handler does not use SYGWMAP for requests that contain a remote procedure call, for example:

```
EXEC remote_procedure
```

where *remote_procedure* is the name of a remote procedure.

3.2.1 Changing the default language request

If the SYGWMAP exit is used to replace the name of the remote procedure called to handle a language request, all remote procedure names used by the SYGWMAP exit must be added to the *SYRP* file using the SYRP transaction. Also, all remote procedure names must be unique.

3.2.2 Errors

If the SYGWMAP exit routine returns a non-zero code, the Server Option context handler sends the following message to the client application:

```
33891 SYGWCTXH - SYGWMAP Error
```

The Server Option context handler then terminates the current client request.

3.2.3 Parameters

The parameter list passed to the SYGWMAP exit by the Server Option context handler is defined in the *SYGWMAPA* member of the *MACLIB* library. This list consists of the following parameters:

- MAP_EIB — The EXEC Interface Block (EIB) address of the Server Option context handler transaction.
- MAP_LANG_REQUEST — The address of the language request buffer.
- MAP_LANG_REQ_LENGTH — The length of the language request buffer in bytes.
- MAP_LANG_TRAN — The address of a field containing the name of the remote procedure used to handle a language request. The maximum length of this parameter value is 30 bytes. On input, the field contains the default value “Language_Request,” but the SYGWMAP exit may change the field contents to any valid remote procedure name used in handling a specific language request.
- MAP_RC — The SYGWMAP exit return code. There are two valid values for this output parameter:
  - 0 — Indicates no errors.
  - 1 — Indicates that SYGWMAP failed. The Server Option context handler reports an error message.

3.2.4 SYGWMAP code

The following code is a framework for the SYGWMAP exit. You must write additional code to parse the language request and return the Server Option remote procedure name to handle the corresponding language event.

```
         TITLE 'SYBASE CONTEXT HANDLER LANG TRAN MAPPING EXIT'
*
*-------------------------------------------------------------------*
*       Sybase     Gateway Library
*       Confidential Property of Sybase Inc.
*       (c) Copyright Sybase, Inc   2004
*       All rights reserved
*
*       This subroutine is called by the mainline SYGWCTXH
*       context handler. It passes:
*      .  Addr of context handler CICS EIB (input).
*      .  Addr of Language Requestbuffer read in by the context
*         handler (input).
*      .  Length of the language request statement (input).
*      .  Addr of a field to contain a valid RPC name to process
*         language requests (input/output). Max. length of 30 bytes.
*         Note: on input the field defaults to "Language_Request".
*         If the field is changed by this exit, then the new value
```

```
*         must also be in the SYRP file.
*      .  Return Code (output)  0=OK
*                               1=33891 SYGWCTXH SYGWMAP Error
*
* History:
*
*-------------------------------------------------------------------------*
*      EQUATES
*-------------------------------------------------------------------------*
R00       EQU   0                    GENERAL REGISTERS
R01       EQU   1
R02       EQU   2
R03       EQU   3
R04       EQU   4
R05       EQU   5
R06       EQU   6
R07       EQU   7
R08       EQU   8
R09       EQU   9
R10       EQU   10
R11       EQU   11
R12       EQU   12
R13       EQU   13
R14       EQU   14
R15       EQU   15
*
*-------------------------------------------------------------------------
*     Parameter dsect for SYGWMAP call
*-------------------------------------------------------------------------
PARMS    DSECT
         COPY SYGWMAPA
*-------------------------------------------------------------------------
*     WORKING STORAGE SECTION
*-------------------------------------------------------------------------*
DFHEISTG  DSECT
uSERID    DS    8C   user id to be returned here
*
*
*-------------------------------------------------------------------------*
*        PROGRAM ENTRY
*-------------------------------------------------------------------------*
SYGWMAP DFHEIENT CODEREG=(12) EIBREG=11
SYGWMAP AMODE 31
SYGWMAP RMODE ANY
*-------------------------------------------------------------------------
* address the parameter area and do some initialization
```

```
*-------------------------------------------------------------------
         USING PARMS,R09
         LR    R09,R01
         L     R04,MAP_EIB              Get Addr of EIB
         L     R05,MAP_LANG_REQUEST     Get Addr of Lang Req
         L     R06,MAP_LANG_REQ_LENGTH  Get Length of Lang Req
         L     R07,MAP_LANG_TRAN        Get Addr of name of Lang Req
         MVC   MAP_RC,ZERO Init Return code to OK
*
*        ADD CODE TO PARSE SQL STATEMENT POINTED TO BY R05
*
*        If you need Client's User Id, uncomment the following line
*        EXEC CICS ASSIGN USERID(USERID)
*        CLC   USERID(8),.................
*
NOCHANGE B     RETURN              Default
*
CHANGE1  MVC   0(30,R07),=CL30'Language_Request2'
         B     RETURN
CHANGE2  MVC   0(30,R07),=CL30'Language_Request3'
         B     RETURN
*
RETURN   DS    0H
*
         DFHEIRET
*-------------------------------------------------------------------*
*      CONSTANTS
*-------------------------------------------------------------------*
ZERO        DC    F'0'
ONE         DC    F'1'
*
            LTORG
            END
```

3.2.5 Link JCL

After compiling the newly created SYGWMAP exit, use the following JCL to link the SYGWMAP exit to the Server Option context handler module.

```
//JOBNAME  JOB  (ACCTNR,ACCTINFO)
//*-------------------------------------------------------------------*
//*            USE THIS JCL TO LINK YOUR SYGWMAP EXIT ROUTINE TO
//*            THE CONTEXT HANDLER LOAD MODULE.
//*-------------------------------------------------------------------*
//*            CHANGE SYSLIB DD TO POINT TO YOUR INPUT LOAD LIBRARY
//*            CHANGE EXITLIB DD TO POINT TO YOUR OBJECT LIBRARY
```

```
//*            CHANGE SYSLMOD DD TO POINT TO YOUR OUTPUT LOAD LIBRARY
//*-------------------------------------------------------------------*
//LKED      EXEC PGM=HEWL,REGION=0M,
//               PARM='LIST,RENT,XREF,AMODE=31,RMODE=ANY'
//SYSUT1   DD  UNIT=SYSDA,DISP=(NEW,DELETE),
//               SPACE=(CYL,(4,4),,CONTIG,ROUND)
//SYSLIB   DD  DISP=SHR,DSN=hlq.OSC126.CICS.LOADLIB
//EXITLIB  DD  DISP=SHR,DSN=MYLIB.DEBUG.OBJLIB
//SYSLMOD  DD  DISP=SHR,DSN=MYLIB.DEBUG.LOADLIB
//SYSPRINT DD  SYSOUT=*
//SYSLIN   DD  *
 ORDER DFHEAI
 INCLUDE EXITLIB(SYGWMAP)     EXIT PROGRAM
 INCLUDE SYSLIB(SYGWCTXH)     CONTEXT HANDLER
 ENTRY SYGWCTXH               MODULE ENT Y POINT
 NAME SYGWCTXH(R)
/*
//
```

3.3 Abend handler

The Server Option for CICS provides a generic abend handling module, SYOSABND, which can be found in the Server Option *LOADLIB*. To use the newly supplied abend handler module, do the following:

- In the CICS RDO definitions, set the TWASIZE of the RPC transaction to be at least 5 bytes.
- In the RPC application program, put the TDSPROC handle in the first 4 bytes of the TWA, and put a character Y in the 5th byte.
- In the RPC application program, issue a CICS HANDLE ABEND command for the SYOSABND program.

The following is a COBOL excerpt from a user RPC routine that calls the new CICS abend handling module, SYOSABND.

```
WORKING-STORAGE SECTION.
*Pointer field for TWA
01 WS-TWAPTR   POINTER.

*field to store the TWA length****
01 WS-TWASIZE   PIC 9(4) COMP.

LINKAGE SECTION.
*Area to store a pointer to the OSC TDSPROC and the Indicator.
*The TDS PROC is usually named "GWL-PROC" and is the first
```

```
*parameter in the OSC call such as TDACCEPT.
01 LK-TWAREA.
  05 LK-GWLPROCPTR   POINTER.
  05 LK-SWITCH   PIC X(1).

PROCEDURE DIVISION.
  EXEC CICS
  ASSIGN TWALENG(WS-TWASIZE) NOHANDLE
  END-EXEC.
*Make sure the TWA is large enough, has to be at least 5 bytes. If not,
*we have a problem.
  IF WS-TWASIZE NOT EQUAL 5
    THEN GO TO RETURN1.

  EXEC CICS
    ADDRESS TWA(WS-TWAPTR)
  END-EXEC.
*Make sure the WS-TWAPTR contains a valid address. If not, we have a
*problem.
  IF WS-TWAPTR EQUAL NULL
    THEN GO TO RETURN1.
*Move the address of the GWL-PROC and set the indicator to 'Y'.
  SET  ADDRESS  OF LK-TWAREA TO WS-TWAPTR.
  SET  LK-GWLPROCPTR TO GWL-PROC.
  MOVE 'Y'   TO LK-SWITCH.

EXEC CICS
  HANDLE ABEND PROGRAM('SYOSABND')
END-EXEC.
```

The following message is returned to the client if an abend occurs:

```
Msg 1, Level 11, State 0:
Procedure 'SYAB':
SYAB ABEND CODE ASRA
```

3.4 Client Option and Server Option application programs relink

The Client Option for CICS and Server Option for CICS version 12.50.01 and 12.50.02 changed the SYGWCAAC, SYGWCAAS, SYGWCACC, and SYGWCACS user application stubs. These stubs are linked with Client Option for CICS or Server Option for CICS user application programs. These application programs must be relinked if any of the following situations apply:

- EXEC CICS HANDLE CONDITION conditions are handled incorrectly in application programs.

- Application programs are running under any version of z/OS after having been migrated from OS/390.
- There is insufficient CICS storage (below 16MB).

Note You do not need to relink existing Client Option for CICS or Server Option for CICS applications with the 12.6 version if you have already relinked with version 12.50.02.

3.5 Updated sample programs

Sample programs, source code, and JCL compile and link modules provided with the Client Option for CICS and the Server Option for CICS have been changed to accommodate compiler changes. Sybase provides these updated *SOURCE* and *JCL* libraries.

3.6 Unicode support

The current version of the Server Option for CICS contains support for Unicode based on the Unicode support provided by IBM z/OS, including the conversion environment and conversion services. With the conversion environment and services installed and set up, the Server Option can convert character streams from one Coded Character Set Identifier (CCSID) to another. This support is provided in addition to the support for language and character sets offered in previous versions.

For details on Unicode, refer to IBM documentation.

3.6.1 Installing and enabling the IBM z/OS conversion environment and services

❖ **Installing Unicode support**

1 Create an *IMAGE* member in *SYS1.PARMLIB* using the CUNMIUTL utility.

2 Copy the *CUNIMG01* member from *WORK.IMAGE* to *SYS1.PARMLIB*.

3 The *CUNIMG01* member is loaded into z/OS using the SET UNI=01 command.

4 The DISPLAY UNI, ALL command displays the current active image and the character set conversions defined for that image.

To enable Unicode support, set the USEIBMUNICODE configuration parameter to Y. The USEIBMUNICODE is specified in the SYGWMCST macro in the *SYGWXCPH* customization module. The Server Option uses the newly defined unichar, univarchar, and unitext internal datatypes and performs conversions between UTF-8, UTF-16, and other CCSIDs.

For information on installing Unicode support for IBM z/OS, see "Support for Unicode Using Conversion Services" (SA22-7649-01).

3.6.2 SYGWXCPH customization module changes

The character set translation routines in the Server Option use tables in the *SYGWXCPH* customization module for the conversion of character sets. Because IBM Unicode support requires the CCSIDs of the character sets involved in conversion, the translation tables in the *SYGWXCPH* customization module and the SYGWMCXL macro have been modified to contain CCSIDs.

SYGWMCST

The USEIBMUNICODE parameter has been added to the SYGWMCST customization macro. The following are valid values for the USEIBMUNICODE parameter:

- Y – Use IBM support for character set conversions.
- N – Use the original Server Option support.

SYGWMCXL

The SYGWMCXL macro has been modified to include the following parameters, which are used for character conversion:

- CCSID – the CCSID for the character set.
- CHARSETTYPE – the character set type. A indicates ASCII, and E indicates EBCDIC.
- CHARSIZE – the maximum length of a character, between 1 and 4 bytes.
- PAD – the padding character. This parameter value depends on the type of character set. For ASCII, the padding character is 20. For EBCDIC, the padding character is 40.

Example 1

```
SYGWMCXL TYPE=ENTRY,
      CHARSET=cp939,CHARSETBYTES=D,
      CCSID=939,CHARTYPE=E,CHARSIZE=2,PAD=40
```

Example 2

```
SYGWMCXL TYPE=ENTRY,
      CHARSET=Russian,CHARSETBYTES=S,
      CCSID=1025,CHARTYPE=E,CHARSIZE=1,PAD=40
```

3.6.3 New datatypes for Unicode support

Components of the Mainframe Connect IPS have two new datatypes using the UTF-16 encoding of the Unicode character. The new unichar and univarchar datatypes are independent of the existing char and varchar datatypes but behave similarly. Like the char datatype, unichar is a fixed-width, non-nullable datatype. Like the varchar datatype, univarchar is a variable-width, nullable datatype. Each unichar or univarchar character requires 2 bytes of storage, so a unichar or univarchar column consists of 16-bit Unicode values.

Note Components of the Mainframe Connect IPS also have a unitext datatype defined, but there is no special support for it.

3.6.4 Unicode support in the Server Option for CICS

The unichar, univarchar, and unitext datatypes have been added for Unicode support in the Server Option. These three datatypes are mapped to TDS_LONGBINARY with a user type of 34, 35, or 36, as shown in Table 3.

Table 3: Unicode datatype mappings

SQL datatype	TDS datatype	User type	Comment
unichar	TDS_LONGBINARY	34	Fixed-length UTF-16 data
univarchar	TDS_LONGBINARY	35	Variable-length UTF-16 data
unitext	TDS_LONGBINARY	36	UTF-16 encoded data

The Server Option has the following three datatypes to support unichar, univarchar, and unitext:

- TDSUNICHAR – Internal type 26
- TDSUNIVARCHAR – Internal type 27
- TDSUNITEXT – Internal type 28

Note Currently, there is no special support for TDSUNITEXT.

The following API calls have been changed in the Server Option to accommodate support for Unicode:

- TDPROPS
- TDESCRIB

TDPROPS

The TDPROPS API call maintains character set conversion properties.

Syntax

```
COPY SYGWCOB

01  TDSPROC            PIC S9(9) COMP.
01  RETCODE            PIC S9(9) COMP.
01  OPER               PIC S9(9) COMP.
01  PROPERTY           PIC S9(9) COMP.
01  VALUE              PIC S9(9) COMP.

CALL 'TDPROPS' USING TDSPROC RETCODE OPER PROPERTY
VALUE.
```

Arguments

TDPROC – (I) Handle for this client-server connection.

RETCODE – (O) Variable where the result of function execution is returned.

OPER – (I) Value must be TDS-GET to retrieve the property specified by PROPERTY or TDS-SET to change the property specified by PROPERTY.

PROPERTY – (I). TDPROPS supports the following properties:

- TDS_CLIENT_CCSID – defines the CCSID to which the Server Option converts server data. The value of this property defaults to the CCSID of the character set negotiated between the client and the server at login.
- TDS_SERVER_CCSID – defines the CCSID to which the Server Option converts client data. The value of this property defaults to the CCSID of the character set negotiated between the client and the server at login. A UTF-8 connection is established in the case when the client-requested character set at login is UTF-8, and Unicode support is enabled for the Server Option.
- TDS_PROG_CCSID – controls the conversion of data between the character sets of the server and the server application. For example, if a Server Option application sets TDS_PROG_CCSID to 1025 (Russian EBCDIC, CCSID=1025), and data received from the server is in UTF-8 (CCSID=1208), a parameter retrieved to a character variable in a TDRCVPRM call will be implicitly converted from CCSID=1208 to CCSID=1025.

- TDS_DATA_CCSID – controls the conversion of metadata. For example, if an OSC application program sets TDS_DATA_CCSID property value to 1025 (Russian EBCDIC, CCSID=1025) and data received from the server is in UTF-8 (CCSID=1208), then the column names will be retrieved for the application program after being implicitly converted from CCSID=1208 to CCSID=1025.

VALUE – (I) The value of the property specified in PROPERTY.

Note For a connection established with the UTF-8 character set, the default values for the TDS_CLIENT_CCSID, TDS_SERVER_CCSID, TDS_PROG_CCSID, and TDS_DATA_CCSID parameters are 1208, 1208, 500, and 500, respectively.

TDPROPS may specify values for these properties any time after a connection has been established. The default values for these properties depend on the character set established for the connection at login.

Note The Server Option does not reset any CCSID property values set by an application program. Once an application changes a CCSID property value, the setting remains for all API calls until it is reset by the application.

Example 1

TDS_PROG_CCSID is set to 1208 (UTF-8), and both TDS_SERVER_CCSID and TDS_CLIENT_CCSID default to 1208. The server application program calls TDRCVPRM.

- When the server reads data from the client, the data is in UTF-8.
- When a Server Option application requests data from the server, the data is retrieved in UTF-8.

Example 2

TDS_PROG_CCSID is set to 1025 (Russian EBCDIC, CCSID=1025), and both TDS_SERVER_CCSID and TDS_CLIENT_CCSID default to 1208. The application calls TDRCVPRM:

- When data is read by the server from the client, the data is in UTF-8.
- When data is requested by the Server Option application from the server, the data is retrieved in EBCDIC CCSID=1025.

Example 3

TDS_PROG_CCSID is set to 939 (Japanese EBCDIC, CCSID=939), and both TDS_SERVER_CCSID and TDS_CLIENT_CCSID default to sjis (CCSID=943). The application calls TDRCVPRM:

- When data is read by the server from the client, the data is in sjis.

- When data is requested by the Server Option application from the server, the data is retrieved in EBCDIC CCSID=939.

TDESCRIB

The TDESCRIB API call now allows use of the TDSUNICHAR, TDSUNIVARCHAR, and TDSUNITEXT datatypes.

Table 4 lists new datatype conversions supported.

Table 4: New datatype mappings

Datatype	Datatype
TDSCHAR	TDSUNICHAR
TDSCHAR	TDSUNIVARCHAR
TDSCHAR	TDSUNITEXT
TDSVARCHAR	TDSUNICHAR
TDSVARCHAR	TDSUNIVARCHAR
TDSTEXT	TDSUNITEXT

3.6.5 Remote procedure calls in the Server Option for CICS

The following constraints apply to remote procedure calls made from isql, or applications similar to isql, through a UTF-8 connection in a two-tier environment:

- Remote procedure call names must consist solely of single-byte UTF-8 characters represented by code points 1-127.
- If the remote procedure call uses in-stream parameters, all multiple-byte parameters must be enclosed in single or double quotes.
- Parameter names must consist of single-byte UTF-8 characters.

Example 1

The file named *RuslangparmUTF8.txt* contains the following:

```
exec rpcuni "Это utf8 параметр", "hello"'
go
```

The following is a valid remote procedure call using in-stream parameters:

```
%> isql -SmyOSC -UmyUser -PmyPass -Jutf8 -
iRuslangparmUTF8.txt
```

Example 2

The file named *NamelangparmUTF8.txt* contains the following:

```
exec rpcuni @P1="Это utf8 параметр", @P2= "hello"
go
```

The following is a valid remote procedure call using in-stream parameters:

```
%> isql -SmyOSC -UmyUser -PmyPass -Jutf8 -
iNamelangparmUTF8.txt
```

3.7 Text and image data

Client applications send text and image data to the Server Option in a writetext stream. To process writetext stream data, a Server Option application cannot employ functions normally used to process parameter data. Instead, a Server Option application must use special text and image functions.

A Server Option application can send text or image data to a client application in either of the following ways:

- *data stream* – If the row of returned data contains one column of text or image data, the row may be sent as a data stream. The length of the data is between 0 and 2 gigabytes.
- *describe/send row* – If the row of returned data contains columns in addition to a text or image column, the text or image data may be sent using the describe/send row method. The length of the data cannot exceed 32KB.

The following subsections describe text and image issues for the Server Option:

- CS_IODESC structure
- Retrieving data from a client
- Returning data to a client
- Text and image functions

3.7.1 CS_IODESC structure

The CS_IODESC structure describes text or image data and is used to pass information between a Server Option application and the API functions that process this data.

The general structure for a CS_IODESC, regardless of programming language, is shown in Table 5.

Table 5: CS_IODESC structure

This field	Contains this information
IOTYPE	Indicates the type of input or output to perform. For text and image operations, *IOTYPE* always has the value CS_IODATA.
DATATYPE	The datatype of the data object. The only legal values for *DATATYPE* are TDSTEXT and TDSIMAGE.
LOCALE	Not used in the Server Option. Set this to NULL.
USERTYPE	Not used in the Server Option.
TOTAL_TXTLEN	In bytes, the total length of the text or image value.
OFFSET	Reserved for future use.
LOG_ON_UPDATE	Determines whether the update to this text or image value should be logged. This field is not used by the Server Option.
NAME	The name of the text or image column.
NAMELEN	In bytes, the length of *NAME*.
TIMESTAMP	The text timestamp of the column. A text timestamp marks the time of the last modification to a text or image column.
TIMESTAMPLEN	Not used by the Server Option.
TEXTPTR	A text pointer to a table row ID.
TEXTPTRLEN	In bytes, the length of *TEXTPTR*. This length is currently set at 16.

The CS_IODESC structure is defined in the *SYGWCOB* copy book for COBOL (under the name CS-IODESC) and in the *SYGWPLI* INCLUDE member for PL/1.

When receiving text or image data from a client application, a Server Option application invokes the TDINFTXT function with the *ACTION* parameter set to TDS_GET. The Server Option application must provide the correct text or image DATATYPE field value *before* TDINFTXT is invoked so that the Server Option can translate incoming text data. Only the value of the TOTAL_TXTLEN field is provided by TDINFTXT here.

When sending text or image data to a client application, the Server Option application also invokes the TDINFTXT function with the *ACTION* parameter set to TDS_SET. The Server Option application must describe the text or image data to be sent to the client by providing values for the appropriate CS_IODESC fields *before* TDINFTXT is invoked.

3.7.2 Retrieving data from a client

A writetext stream retrieved from a client application is handled as bulk data by the Server Option application.

An application processes incoming text or image data in two steps:

1 The TDINFTXT function retrieves a description of the text or image data and places the description in a CS_IODESC structure. The TDINFTXT function call returns information including the total length of incoming data. This length enables the Server Option application to determine whether the data should be retrieved in one unit or in sections. The Server Option application also determines the size of the buffer that must be allocated to store the incoming data. TDINFTXT is invoked with the *ACTION* parameter set to TDS_GET. The DATATYPE field of the CS_IODESC structure must be provided by the Server Option application before TDINFTXT is invoked. See “TDINFTXT” for details on this function.

2 The TDGETTXT function retrieves the incoming text and image data from the client application in the specified section size and stores the data in the specified buffer. See “TDGETTXT” for details on the TDGETTXT function.

Note A call to TDINFTXT must always precede a call to TDGETTXT. The TDGETTXT routine must be called until all text has been read from the client.

Table 6 illustrates the sequence of API function calls for retrieving text or image bulk data from the client.

Table 6: API function call sequence for data retrieval

Function	Action performed
TDSQLLEN	Determines the length of the incoming writetext string

TDSRCVSQL	Retrieves a writetext string. The TDSRCVSQL function call receives a writetext bulk command, which indicates that text or image bulk data follows. The writetext bulk command occurs in the following format:
	`writetext bulk <object_name> <text_pointer> timestamp = <time_stamp> [with log | without log]`
	The parameters of the writetext bulk command are as follows:
	• <*object_name*> is the name of the object name to which data is to be sent.
	• <*text_pointer*> is a text pointer.
	• <*time_stamp*> indicates the value of the timestamp parameter.
	• The text *with log* or *without log* is not used by the Server Option.
	For example:
	`writetext bulk SYBASE.au_txt.TXT 0xa1a0bbd014a6d005060e016a20400100 timestamp = 0x0000000000000000 with log`
	The manner in which the writetext bulk parameters are used depends on the Server Option application and on the destination of the incoming text and image data.

TDSNDDON	Notifies the sender that the SQL string has been received. Use the connection option of TDS_ENDREPLY to change the communication state from *send* to *receive*.
TDINFTXT (using TDS_GET)	Returns the length of the entire text or image data stream. The Server Option translates incoming data based on the value of the DATATYPE field in the CS_IODESC structure.
TDGETTXT	Retrieves a section of the text or image data stream. TDGETTXT is invoked in a loop until all incoming data is retrieved.
TDSNDDON	Notifies the sender that all data has been received.

3.7.3 Returning data to a client

A Server Option application sends text or image data to a client application in one of two ways, depending on the number of columns in the data row.

If there is one text or image column in the row to be sent, the Server Option application proceeds as follows:

1 Using the TDESCRIB function, the Server Option application describes the format in which the client receives the text or image column.

2 Optionally, you can use the TDSETUDT function to set the user-defined datatype for the text or image column.

3 The Server Option application invokes the TDINFTXT function with the *ACTION* parameter set to TDS_SET to indicate the total length of the returning data.

4 The Server Option application invokes the TDSNDTXT function to send the data to the client in sections.

Table 7 illustrates the sequence of API function calls for sending text or image bulk data to the client.

Table 7: API function call sequence for sending bulk data only

Function	Action performed
TDESCRIB	Describes the text or image column to be sent to the client.
TDSETUDT (optional)	Sets the user-defined datatype for the column.
TDINFTXT (using TDS_SET)	Describes the text or image column to the Server Option. The Server Option application provides values for the CS_IODESC fields before invoking the TDINFTXT function. The TDINFTXT function is invoked once for every row that is to be sent to the client.
TDSNDTXT	Sends a section of the text or image data stream. The TDSNDTXT function is invoked in a loop until all the data for a given row is sent to the client.
TDSNDDON	Notifies the client that all data has been sent.

If there are other columns in addition to the text and image data in the row to be sent, the Server Option application proceeds as follows:

1 Using the TDESCRIB function, the Server Option application describes the format in which the client receives a column of data. The Server Option application invokes the TDESCRIB function once for each column of data.

2 The Server Option application invokes the TDINFTXT function with the *ACTION* parameter set to TDS_SET to provide text pointer and timestamp information. The Server Option application invokes the TDINFTXT function once for each text or image column in a row.

3 The Server Option application transfers the data to the client application using the TDSNDROW function, which is invoked once for each row of data. The text or image column size must not exceed 32KB.

Table 8 illustrates the sequence of API function calls for sending rows in which there are other columns in addition to the text or image data columns.

Table 8: API function call sequence for sending row data of varied column datatypes

Function	Action performed
TDESCRIB	Describes a column to be sent to the client. The TDESCRIB function is invoked once for each column of data to be sent to the client.

Function	Action performed
TDINFTXT (using TDS_SET)	Describes a text or image column to the Server Option. The Server Option application provides values for the CS_IODESC fields before invoking the TDINFTXT function. The TDINFTXT function is invoked in two nested loops, once for every text or image column in a row to be sent to the client, and once for every row to be sent to the client.
TDSNDROW	Sends a row of data to the client. The TDSNDROW function is invoked in a loop for every row of data to be sent to the client and preceded by a number of TDINFTXT calls describing the text and image columns in a row.
TDSNDDON	Notifies the client that all data has been sent.

3.7.4 Text and image functions

The Server Option provides three new functions: TDINFTXT, TDGETTXT, and TDSNDTXT. These functions can be invoked from within a Server Option application written in COBOL or PL/1. The TDINFTXT, TDGETTXT, and TDSNDTXT functions are described in the following sections using COBOL syntax.

TDINFTXT

Function

Sets or gets a description of text or image data.

Syntax

```
01 TDPROC          PIC S9(9) USAGE COMP SYNC.
01 RETCODE         PIC S9(9) USAGE COMP SYNC.
01 ACTION          PIC S9(9) USAGE COMP SYNC.
01 ITEM-NUMBER     PIC S9(9) USAGE COMP SYNC.
01 CS-IODESC FROM SYGWCOB


CALL 'TDINFTXT' USING TDPROC, RETCODE, ACTION, ITEM-
NUMBER, CS-IODESC.
```

Arguments

TDPROC

(I) Handle for the client/server connection. The value here must be the same value specified in the associated TDACCEPT function call. The *TDPROC* handle corresponds to the connection and command handles in Open Client Client-Library.

RETCODE

(O) Variable to which the result of function execution is returned. The value of this variable is one of the codes listed below under “Returns.”

ACTION

(I) Action to be taken by this call. *ACTION* is an integer variable that indicates the purpose of this call.

Assign *ACTION* one of the following symbolic values:

TDS_GET (1)	The Server Option updates the CS_IODESC structure with the total length of the text or image data to be read from the client. Typically, this is followed by a call to the TDGETTXT function. The Server Option application must set the DATATYPE field in the CS-IODESC structure to TDSTEXT or TDSIMAGE before invoking TDGETTXT.
TDS_SET (2)	The Server Option sets internal Server-Library structures to describe a text or image data object. The TDINFTXT call updates a text or image column with the information contained in CS-IODESC. The Server Option application must describe the column using TDESCRIB before TDINFTXT is invoked.

ITEM-NUMBER

(I) The column number of the column being described. The first column in a row is column 1. This parameter is ignored when *ACTION* is TDS_GET.

CS-IODESC

(I) A pointer to the CS-IODESC for the application.

Comments

- TDINFTXT is used to describe text or image columns for sending a result row or retrieving a parameter.
- If *ACTION* is TDS_GET, TDINFTXT must be called prior to the first or only call to TDGETTXT for a row.
- If *ACTION* is TDS_SET, TDINFTXT must be called for each text or image datatype column in a row before TDSNDTXT or TDSNDROW is called.
- Text and image data is transferred to the client using either TDSNDTXT or TDSNDROW.

Returns

The *RETCODE* argument can contain any of the following values:

- TDS_OK (0)
- TDS_INVALID_PARAMETER (-4)
- TDS_INVALID_DATA_TYPE (-171)
- TDS_ILLEGAL_REQUEST (-5)
- TDS_INVALID_LENGTH (-173)

- TDS_RESULTS_COMPLETE (500)
- TDS_WRONG_STATE (-6)
- TDS_CONNECTION_FAILED (-4998)
- TDS_CONNECTION_TERMINATED (-4997)

See also

Related functions:

- TDSNDTXT
- TDGETTXT

TDSNDTXT

Function

Sends a subsequent part of the text or image data stream to the client.

Syntax

```
01 TDPROC                 PIC S9(9) USAGE COMP SYNC.
01 RETCODE                PIC S9(9) USAGE COMP SYNC.
01 HOST-VARIABLE-NAME     PIC X(n).
01 BUFLEN                 PIC S9(9) USAGE COMP SYNC.


CALL 'TDSNDTXT' USING TDPROC, RETCODE, HOST-VARIABLE-
NAME, BUFLEN.
```

Arguments

TDPROC

(I) Handle for the client/server connection. The value here must be the same value specified in the associated TDACCEPT function call. The *TDPROC* handle corresponds to the connection and command handles in Open Client Client-Library.

RETCODE

(O) Variable to which the result of function execution is returned. The value of this variable is one of the codes listed below under “Returns.”

HOST-VARIABLE-NAME

(I) Application program variable that contains data for this column.

BUFLEN

(I) The size in bytes of the buffer containing the data.

Comments

- TDSNDTXT is used when sending a single column of text or image data to the client.

- The Server Option application must always call TDINFTXT prior to the first call to TDSNDTXT for the data stream, in order to set the total length of the data to be sent. The application then calls TDSNDTXT to send a part of the data. TDSNDTXT is called as many times as there are sections of data in the data stream.
- The item being sent to the client must have previously been described using TDESCRIB.
- A Server Option application can also write text and image data to a client using TDSNDROW. TDSNDTXT allows the application to send the data in sections, whereas the standard TDSNDROW method requires that all the data in the column be sent in one piece.
- A column sent with TDSNDTXT must be of type text or image.
- The Server Option treats text and image data streams identically except for character set conversion, which is only performed on text data.

Returns

The *RETCODE* argument can contain any of the following values:

- TDS_OK (0)
- TDS_ILLEGAL_REQUEST (-5)
- TDS_INVALID_VAR_ADDRESS (-175)
- TDS_CANCEL_RECEIVED (-12)
- TDS_WRONG_STATE (-6)
- TDS_INVALID_LENGTH (-173)
- TDS_CONNECTION_TERMINATED (-4997)

See also

Related functions:

- TDGETTXT
- TDINFTXT

TDGETTXT

Function

Reads a subsequent part of a text or image datastream from the client.

Syntax

```
01 TDPROC                  PIC S9(9) USAGE COMP SYNC.
01 RETCODE                 PIC S9(9) USAGE COMP SYNC.
01 HOST-VARIABLE-NAME      PIC X(n).
01 BUFLEN                  PIC S9(9) USAGE COMP SYNC.
01 OUTLEN                  PIC S9(9) USAGE COMP SYNC.


CALL 'TDSNDTXT' USING TDPROC, RETCODE, HOST-VARIABLE
NAME, BUFLEN.
```

Arguments

TDPROC

(I) Handle for the client/server connection. The value here must be the same value specified in the associated TDACCEPT function call. The *TDPROC* handle corresponds to the connection and command handles in Open Client Client-Library.

RETCODE

(O) Variable to which the result of function execution is returned. The value of this variable is one of the codes listed below under “Returns.”

HOST-VARIABLE-NAME

(I) Application program variable to receive a subsequent part of the incoming text or image client data.

BUFLEN

(I) The size in bytes of the buffer containing the data.

OUTLEN

(O) The length in bytes of the data received.

Comments

- TDGETTXT is used to read bulk data from the client. The bulk data can be of type text or image.
- TDGETTXT must be called until all of the bulk data has been read from a client. The Server Option application must keep track of the data that remains to be read.
- A column read with TDGETTXT must be of type text or image.
- A Server Option application must call TDINFTXT prior to the first call to TDGETTXT for the data stream. The application then calls TDGETTXT to retrieve a section of data. TDGETTXT is called as many times as are necessary to read in the whole stream.
- The Server Option application must set the CS_IODESC DATATYPE field to TDSTEXT or TDSIMAGE before invoking the TDINFTXT and TDGETTXT functions. In the case in which DATATYPE is set to TDSTEXT, the Server Option translates the character set for the client data before sending the data to the Server Option application.

Returns

The *RETCODE* argument can contain any of the following values:

- TDS_OK (0)
- TDS_INVALID_VAR_ADDRESS (-175)
- TDS_INVALID_LENGTH (-173)
- TDS_ILLEGAL_REQUEST (-5)
- TDS_CONNECTION_FAILED (-4998)
- TDS_CONNECTION_TERMINATED (-4997)

See also

Related functions:

- TDSNDTXT
- TDINFTXT

3.8 Listener handling of concurrent connection request

The Sybase Listener distributed with the Server Option can handle up to 128 concurrent connection requests. Sybase can provide a code zap for use in rare cases, when the number of concurrent connection requests exceeds 128, to enable the Server Option to handle up to 300 concurrent connection requests.

The volume of concurrently managed connection requests can also be increased by the activation of more listeners on different ports, and different context handler transaction names can be associated with different listener names.

Careful tuning may be required for a high volume of concurrent connection requests in a two-tier environment where the context handler task owns the socket for a connection and is active for the life of the user RPC task. The CICS MXT parameter should be set accordingly, and the TCLASS parameter can be used to restrict the number of concurrent context handler tasks for a listener.

4. Known issues

The following section describes known issues in the Server Option for CICS 12.6.

4.1 InstallShield license-key strings

License keys containing sequences of multiple consecutive dollar signs ($$$) entered in the InstallShield installation wizard are rendered in the resulting install job, *IxLIC*, with only one dollar sign instead of a sequence.

For example, a license-key string entered in the InstallShield installation wizard as `A$$$B$$C` is rendered in the install job as `A$B$C`.

To correct your license key, edit the license string in the *IxLIC* install job after you have run the InstallShield installation wizard.

4.2 Decimal loss in TDCONVRT money-to-char conversion

The TDCONVRT API function of the Server Option handles conversion of data from the money datatype to the char datatype based on the value of the NUM-DECIMAL-PLACES input parameter. This parameter defines the number of positions following the decimal point in the output data. If NUM-DECIMAL-PLACES is set to 0, there are no positions following the decimal point. In this case, TDCONVRT does not default to 2 positions after the decimal point, as with other products.

This issue concerns CR #352910.

4.3 InstallShield wizard temporary space requirement

The InstallShield wizard, which runs only on Windows, requires a maximum of 800KB of free disk space for temporary files.

5. Product compatibilities

For full functionality with the current release, use these Sybase components, as available at your site:

Table 9: Sybase product release compatibility

Component	Release level
Mainframe Connect Client Option	12.6
Mainframe Connect Server Option	12.6
Mainframe Connect DB2 UDB Option	12.6
DirectConnect Option for z/OS	12.6

6. Technical support

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 have any questions about this installation or if you need assistance during the installation process, ask the designated person to contact Sybase Technical Support or the Sybase subsidiary in your area.

7. Other sources of information

Use the Sybase Getting Started CD, the SyBooks CD, and the Sybase 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 SyBooks CD. It is included with your software. To read or print documents on the Getting Started CD, you need Adobe Acrobat Reader, which you can download at no charge from the Adobe Web site using a link provided on the CD.
- The SyBooks CD contains product manuals and is included with your software. The Eclipse-based SyBooks browser allows you to access the manuals in an easy-to-use, HTML-based format.

Some documentation may be provided in PDF format, which you can access through the PDF directory on the SyBooks CD. To read or print the PDF files, you need Adobe Acrobat Reader.

Refer to the *SyBooks Installation Guide* on the Getting Started CD, or the *README.txt* file on the SyBooks CD for instructions on installing and starting SyBooks.

- The Sybase Product Manuals Web site is an online version of the SyBooks 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.

  To access the Sybase Product Manuals Web site, go to Product Manuals at http://www.sybase.com/support/manuals/.

7.1 Sybase certifications on the Web

Technical documentation at the Sybase Web site is updated frequently.

❖ **Finding the latest information on product certifications**

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

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.

7.2 Sybase EBFs and software maintenance

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

1 Point your Web browser to the Sybase Support Page at http://www.sybase.com/support.

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.