New Features Open Server™ 12.5.1 and SDK 12.5.1 for Windows, Linux, UNIX

Document ID: DC20155-01-1251-20

Last revised: July 2008

This document describes new features available for Open Server™ 12.5.1 and the Software Developer's Kit (SDK) 12.5.1. This document is revised to include new features as they become available.

Topic	Page
Product components	3
Product compatibilities	4
New feature in ESD #24	6
SSL Plus 5.2.2 + SBGSE 2.2 support	7
New features in ESDs #21, #22, and #23	8
New features in ESD #20	8
New srv_send_data routine added	8
ct_data_info enhancement	12
SSL certificate utilities work with SSL Plus 5.0.4 on all platforms	12
New features in ESD #19	12
MIT Kerberos support on HP Itanium	12
Release of 32-bit binaries for 64-bit products	13
SSL Plus 5.0.4 + SBGSE 2.0 support	13
Windows SSPI support using MIT Kerberos	15
New command line option for BCP	15
New features in ESD #18	16
New features in ESD #17	16
Extended support for OpenLDAP	16
XA 32-bit support on HP Itanium 32-bit	16
New features in ESD #16	16
New logging feature for ASE OLE DB Provider	17
New srv_send_data routine added for UNIX platforms	18
New feature in ESD #15	22
Dynamic version of bulk library available	23

Topic	Page
New features in ESD #14	23
Asynchronous execution for ODBC	23
Server-Library routines updated	24
New features in ESD #13	24
New jConnect 6.05 connection properties	25
New jConnect 5.5 connection properties	25
Documentation updates and clarifications to ESD #11	28
New features in ESD #12	28
SSL Plus support on Linux AMD64 (Opteron)/EM64T and HP Itanium 32-bit and 64-bit	29
MIT Kerberos support	29
MIT Kerberos on DB-Library	31
ESQL/COBOL support on HP Itanium 32-bit	31
ASE OLE DB Provider participation in Distributed Transactions	31
OLE DB DSN Migration tool available	33
HA failover on ASE OLE DB Provider	35
New features in ESD #11	39
MIT Kerberos support on Sun Solaris 9 x86 (32-bit)	39
Directory services sql.ini and interface file support	39
ODBC DSN Migration tool	42
Bookmark and bulk support for ODBC and OLE DB	43
Directory services sql.ini and interface file support on jConnect	44
New features in ESD #10	46
Support for HP-UX 11.11 (or HP-UX 11iv1.0) 32-bit and 64-bit platforms	46
XA 64-bit support on Linux AMD64 (Opteron)/EM64T	46
MIT Kerberos support on Microsoft Windows	46
Scrollable cursors	47
New feature in ESD #9	53
Using Distributed Transactions	53
New features in ESD #8	58
Extended support for MIT Kerberos	58
Support for new platforms and operating systems	58
LDAP functionality on IBM AIX 64-bit	59
CyberSafe Kerberos Client upgrade on Windows	59
SSLPlus 3.1.x security fixes	60
COBOL compiler upgrade on Windows 2000	60

Topic	Page
New feature in ESD #7	61
XA 64-bit support extended to additional platforms	61
New feature in ESD #6	62
New ASE OLE DB Provider by Sybase	62
New features in ESD #5	63
BCP encrypted columns	63
Korean character support enhancements	64
SSLPlus 3.1.10H security fixes	64
Support for MIT Kerberos on ODBC and OLE DB	64
Installing SDK 12.5.1 on Windows	65
Extended platform support for the ASE ODBC Driver by Sybase	65
New features in ESD #4	66
Diagnostic enhancements	66
Asynchronous communications support	67
DB-Library new configuration variable	69
Adaptive Server Enterprise ODBC driver for Linux Intel 32-bit	71
New ASE ODBC driver	71
New feature in ESD #3	72
Adaptive Server Enterprise ADO.NET Data Provider 1.1 on Microsoft Windows	72
New features in ESD #2	73
SSL Plus 3.1.5 + SBGSE support on Solaris and Microsoft Windows	73
IPV6 support on Solaris and Microsoft Windows XP and 2003	74
DB-Library LDAP functionality on Microsoft Windows	75
New feature in ESD #1	75
XA support on Sun Solaris 8 (SPARC 64-bit)	75

Product components

SDK contains the following components:

- Open Client™ (DB-Library™, Client-Library)
- Embedded SQL™/C (ESQL/C)
- Embedded SQL/COBOL (ESQL/COBOL)

- Adaptive Server® Enterprise (ASE) ODBC Driver by Sybase® (Microsoft Windows and Linux)
- Adaptive Server Enterprise (ASE) OLE DB Provider by Sybase (Microsoft Windows only)
- Adaptive Server Enterprise (ASE) ADO.NET Data Provider (Microsoft Windows only)
- Extended Architecture (XA)
- jConnect™ for JDBC™

Product compatibilities

Table 1 lists the platforms, compilers, and third-party products Open Server and SDK are built and tested on:

Table 1: Open Client and Open Server platform compatibility

Platform	Operating system level	C and C++ compilers	COBOL compilers	Kerberos version	Light-weight Directory Access (LDAP)	Secure Sockets Layer (SSL)
HP-UX 11.11 (or HP-UX 11iv1) 32-bit	HP-UX 11i v1	HP C/ANSI C B11.11.10	Micro Focus Server Express 4.0	CyberSafe Trust Broker Client 2.1 with GSS runtime lib 3.1.1, MIT 1.3.6	Netscape LDAP 4.1	Certicom SSL Plus 5.2.2 (SBGSE 2.2)
HP-UX 11.11 (or HP-UX 11iv1) 64-bit	HP-UX 11i v1	HP C/ANSI C B11.11.10	Micro Focus Server Express 4.0	MIT 1.4.3	Netscape LDAP 4.1	Certicom SSL Plus 5.2.2 (SBGSE 2.2)
HP Itanium 32-bit	HP-UX 11.23	HP aC++/ANSI C B3910B A.06.00	Micro Focus Server Express 4.0 SP2	MIT 1.4.1	Netscape LDAP 4.1	Certicom SSL Plus 5.2.2

Platform	Operating system level	C and C++ compilers	COBOL compilers	Kerberos version	Light-weight Directory Access (LDAP)	Secure Sockets Layer (SSL)
HP Itanium 64-bit	HP-UX 11.23	HP aC++/ANSI C B3910B A.06.00	Not available	MIT 1.4.1	Netscape LDAP 4.1	Certicom SSL Plus 5.2.2
HP Tru64	Digital UNIX TRU64 5.0a	C++ 6.0-010	DEC COBOL 2.7	Not available	Netscape LDAP 4.1	Certicom SSL Plus 5.2.2
IBM AIX 32-bit	AIX 4.3.3	C++ 5.0.22	Micro Focus Server Express 2.0.10	CyberSafe Trust Broker 2.1	Netscape LDAP 4.1	Certicom SSL Plus 5.2.2
IBM AIX 64-bit	AIX 5.1	C++ 5.0.22	Not available	MIT 1.4.3	Open LDAP 2.2.23	Certicom SSL Plus 5.2.2 (SBGSE 2.2)
Linux on POWER 32-bit	Red Hat Enterprise Linux AS 3.0	IBM XL C/C++ Advance Edition V7.0	Not available	Not available	Not available	Not available
Linux on POWER 64-bit	Red Hat Enterprise Linux AS 3.0	IBM XL C/C++ Advance Edition V7.0	Not available	Not available	Not available	Not available
Linux AMD64 (Opteron)/ EM64T	Red Hat Enterprise Advance Server 3.0	GCC 3.2.3 (Red Hat Linux 3.2.3 - 42)	Not available	MIT 1.2.7	Open LDAP 2.3.27	Certicom SSL Plus 5.2.2
Linux Intel 32-bit	Red Hat Linux 7.2	GCC version 2.96	Not available	MIT 1.2.2	Netscape LDAP 4.1	Certicom SSL Plus 5.2.2
Linux Intel 32-bit	Red Hat Enterprise Linux AS 3.0	GCC version 3.2.3	Not available	MIT 1.2.7	Netscape LDAP 4.1	Certicom SSL Plus 5.2.2
Linux Itanium 64-bit	Red Hat Advance Server 2.1	GCC version 2.96	Not available	Not available	Not available	Not available
Linux Itanium 64-bit	Red Hat Enterprise Linux 3.0	GCC version 3.2.3	Not available	MIT 1.2.7	Not available	Not available

Platform	Operating system level	C and C++ compilers	COBOL compilers	Kerberos version	Light-weight Directory Access (LDAP)	Secure Sockets Layer (SSL)
SGI 32-bit	IRIX 6.5.18	MIPSPro C7.3.x MIPSPro C++ 7.3.x	Not available	Not available	Netscape LDAP 4.0	Not available
SGI 64-bit	IRIX 6.5.18	MIPSPro C7.3.x MIPSPro C++ 7.3.x	Not available	Not available	Not available	Not available
Sun Solaris 8 (SPARC 32-bit)	Solaris 8	Sun C/C++ 6.2	Micro Focus Server Express 2.0.10	CyberSafe Trust Broker 2.1, MIT 1.3.4	Netscape LDAP 4.0	Certicom SSL Plus 5.2.2 (SBGSE 2.2)
Sun Solaris 8 (SPARC 64-bit)	Solaris 8	Sun C/C++ 6.2	Micro Focus Server Express 2.0.10	CyberSafe 2.1, MIT 1.3.4	Netscape LDAP 4.1	Certicom SSL Plus 5.2.2 (SBGSE 2.2)
Sun Solaris 9 x86 (32-bit)	Solaris 9	Sun C/C++ 6.2	Not available	MIT 1.4.1	Not available	Not available
Sun Solaris 10 x64 (64-bit)	Solaris 10	Studio 10 (Sun C/C++ 5.7)	Not available	MIT 1.4.1	Open LDAP 2.2.26	Certicom SSL Plus 5.2.2
Windows 2000 32-bit	Service Pack 4	MS C 6.0 (Microsoft Developers Studio; unoptimized, development only)	Micro Focus Net Express 4.0	Cybersafe Kerbero Client 4.0 with GSS runtime lib 2.0.1; MIT 2.6.5	Netscape LDAP 4.0	Certicom SSL Plus 5.2.2 (SBGSE 2.2)

Note For Open Server and SDK certifications support, see the Sybase platform certifications page at http://certification.sybase.com/ucr/search.do

New feature in ESD #24

The following feature is new for ESD #24:

SSL Plus 5.2.2 + SBGSE 2.2 support

Certicom Secure Sockets Layer (SSL) 5.2.2 support is provided for the Open Server and SDK components—Open Client (Client-Library), ESQL/C, and ESQL/COBOL—on the following platforms:

- HP Digital Unix Tru64 5.0A
- HP Itanium HP-UX 11.23 32-bit and 64-bit
- HP-UX 11.11 32-bit and 64-bit
- IBM RS/6000 AIX 4.3.3 32-bit
- IBM RS/6000 AIX 5.1 64-bit
- Linux AMD64 RHEL 3.0 64-bit
- Linux RH 7.2 32-bit
- Linux RHEL 3.0 32-bit
- Sun Solaris 8 (SPARC) 32-bit and 64-bit
- Sun Solaris 10 x64 64-bit
- Windows x86 32-bit (2000 Service Pack 4 or later)

SBGSE 2.2 is supported on the following SSL Plus 5.2.2 platforms:

- HP-UX 11.11 32-bit and 64-bit
- IBM RS/6000 AIX 5.1 64-bit
- Sun Solaris 8 (SPARC) 32-bit and 64-bit
- Windows x86 32-bit (2000 Service Pack 4 or later)

As a result of U.S. Federal regulations, Sybase is required to use FIPS-certified cipher suites. After you upgrade to SDK 12.5.1 and to Open Server 12.5.1, perform the following:

- For Sun Solaris 8 32-bit and 64-bit, HP-UX 11.11 32-bit and 64-bit, and IBM AIX 5.1 64-bit, be sure that the *lib3p* or *lib3p64* directory is in the dynamic load library path.
- In Windows, to allow the *ctlib* and *srvlib* libraries to find the Certicom cipher suites in *libsbgse2.dll*, add the following to the dynamic load library path:

  ```
  %SYBASE%\%SYBASE_OCS%\lib3p
  ```

When running any application that uses SSL, be sure that the *lib3p* directory is in its dynamic load library path.

New features in ESDs #21, #22, and #23

ESDs #21, #22, and #23 are bug-fixing releases and do not contain new features.

New features in ESD #20

The following features are new for ESD #20:

New *srv_send_data* routine added

Description

srv_send_data, the new API routine, allows Open Server applications to transfer rows containing multiple text/image columns to clients. It allows Open Server applications to send text and image data in chunks, preventing excessive utilization of memory.

Syntax

```
CS_RETCODE srv_send_data(spp, column, buf, buflen)

SRV_PROC *spp;
CS_INT *column;
CS_BYTE *buf;
CS_INT buflen;
```

Parameters

spp

A pointer to an internal thread control structure.

column

The number of the column in a row set.

buf

A pointer to a buffer containing the data to send to the client.

buflen

The length of the **buf* buffer.

Return Value

Table 2: Return values (srv_send_data)

Returns	To indicate
CS_SUCCEED	The routine completed successfully.
CS_FAIL	The routine failed.

Example

```
#include <ctpublic.h>
#include <ospublic.h>
/*
** Local Prototype.
*/
CS_RETCODE ex_srv_send_data PROTOTYPE((
    SRV_PROC *spp,
    CS_COMMAND *cmd,
    CS_INT cols
));
#define MAX_BULK 51200

/*
** EX_SRV_SEND_DATA
** Example routine to demonstrate how to write columns of data in a row set
** to a client using srv_send_data. This routine will send all the columns
** of data read from a server back to the client.

** Arguments:
** spp  A pointer to an internal thread control structure.
** cmd  The command handle for the command that is returning text data.
** cols The number of columns in a row set.

** Returns:
** CS_SUCCEED Result set sent successfully to client.
** CS_FAIL    An error was detected.
*/

CS_RETCODE ex_srv_send_data(spp, cmd, cols)
SRV_PROC *spp;
CS_COMMAND *cmd;
CS_INT cols;
{
  CS_INT   *len;           /* Length of column data. */
  CS_INT   *outlen;        /* Number of bytes received. */
  CS_BYTE  **data;         /* Column data. */
  CS_BYTE  buf[MAX_BULK]; /* Buffer for text data.*/
  CS_BOOL  ok;             /* Error control flag. */
  CS_INT   i;
CS_INT   ret;

/* Initialization. */
ok = CS_TRUE;

/*
** Transfer a row.
*/
for (i = 0; i < cols; i++)
{
  if ((fmt[i].datatype != CS_TEXT_TYPE) &&
       (fmt[i].datatype != CS_IMAGE_TYPE))
  {
    /*
    ** Transfer a non TEXT/IMAGE column.
    */
    /*
    ** Read the data of a non-text/image column from the server.
    */
    ret = ct_get_data(cmd, i+1, data[i],len[i], &outlen[i]);
    if ((ret != CS_SUCCEED)&& (ret != CS_END_DATA)&& (ret != CS_END_ITEM))
    {
        ok = CS_FALSE;
        break;
    }

    /*
    ** Write the data of a non-text/image column to client.
    */
    if (ret = srv_send_data(srvproc, i+1, NULL, 0) != CS_SUCCEED)
    {
      ok = CS_FALSE;
      break;
    }
  }
  else
  {
    /*
    ** Transfer a TEXT/IMAGE column in small trunks.
    */

    /*
    ** Read a chunk of data of a text/image column from the server.
    */
    while ((ret = ct_get_data(cmd, i+1,
            buf, MAX_BULK, &len[i])) == CS_SUCCEED)
        {
          /* Write the chunk of data to client. */
          if (ret = srv_send_data(srvproc, i+1, buf, len[i]) != CS_SUCCEED)
          {
            ok = CS_FALSE;
            break;
          }
        }
      }
    }

    switch(ret)
    {
      case CS_SUCCEED:      /* The routine completed successfully. */
      case CS_END_ITEM:     /* Reached the end of this item’s value.*/
      case CS_END_DATA:     /* Reached the end of this row’s data.*/
        break;
      case CS_FAIL:         /* The routine failed. */
      case CS_CANCELED:     /* The get data operation was cancelled.*/
      case CS_PENDING:      /* Asynchronous network I/O is in effect. */
      case CS_BUSY:         /* An asynchronous operation is pending. */
      default:
        ok = CS_FALSE;
    }
    return (ok ? CS_SUCCEED : CS_FAIL);
  }
```

Note For more information on using srv_send_data, refer to ctos_procmultitextcol in the *ctos.c* sample program.

Usage

- srv_send_data is used to send data of a row set column by column to a client.
- When sending columns with text or image data, Open Server applications must call srv_text_info before srv_send_data. This ensures the data stream is correctly set to the total length of data being sent. The application then calls srv_send_data to send the data in chunks, and continues to call the routine until there is no remaining data to be sent.
- Open Server applications can send text and image data to clients using srv_bind and srv_xferdata. However, these routines require all data columns to be sent at once. srv_send_data allows applications to send text and image data in chunks.

- Open Server applications treat text and image data streams identically, with the exception of character set conversions. These conversions are only performed on text data.

See also

Related srv_bind, srv_get_text, srv_text_info, srv_xferdata, srv_get_data, and srv_send_text routines in the Open Server 12.5.1 *Server Library/C Reference Manual*.

ct_data_info enhancement

You can now call ct_data_info to retrieve fixed I/O fields such as object name *before* a column is read. As before, the changeable fields in I/O descriptors such as pointers to text data, and length of text data are retrievable only after the column is read. This change is particularly useful when using srv_send_data since srv_send_data sends the row's data format before the whole row is read. For details, see ctos_procmultitextcol in the *ctos.c* sample program.

SSL certificate utilities work with SSL Plus 5.0.4 on all platforms

SSL Certificates generated by the certreq, certauth, certpk12 utilities on Digital Unix Tru64 5.0A, IBM RS/6000 AIX 4.3.3 32-bit, and Linux RH 7.2 32-bit platforms now work with products (SDK/Open Server/ASE) that support SSL Plus 5.0.4. For a list of platforms that support SSL Plus 5.0.4, see SSL Plus 5.0.4 + SBGSE 2.0 support.

New features in ESD #19

The following features are new for ESD #19:

MIT Kerberos support on HP Itanium

MIT Kerberos 5 version 1.4.1 is now supported on HP Itanium 32-bit and 64-bit.

Release of 32-bit binaries for 64-bit products

SDK/Open Server binaries like `isql` and `bcp`, share the same name between the 32-bit and 64-bit products. Installing ASE, SDK, or Open Server 64-bit products with other Sybase 32-bit products in the same Sybase installation, overwrites the 32-bit binaries causing issues with the peaceful coexistence of multiple products.

From SDK/Open Server 12.5.1 ESD #19 onwards, the 64-bit binaries are replaced with 32-bit binaries on all 64-bit UNIX platforms, except Sun Solaris x64.

Sun Solaris x64 64-bit will continue to release 64-bit binaries and not 32-bit binaries of Sun x86. This is because SDK/Open Server on Sun x86 32-bit does not support the Directory (LDAP) and Security (SSL) services, and neither do the 32-bit binaries like `isql` on this platform.

SSL Plus 5.0.4 + SBGSE 2.0 support

Certicom Secure Sockets Layer (SSL) Plus 5.0.4 + SBGSE 2.0 support is provided for the following Sybase products:

- Open Server
- SDK components:
    - Open Client (Client-Library)
    - ESQL/C
    - ESQL/COBOL

Support for SSL Plus 5.0.4 is available on the following platforms:

- Digital Unix Tru64 5.0A
- IBM RS/6000 AIX 5.1 64-bit
- IBM RS/6000 AIX 4.3.3 32-bit
- HP-UX 11.11 32-bit and 64-bit
- HP Itanium HP-UX 11.23 32-bit and 64-bit
- Linux RH 7.2 32-bit
- Linux RHEL 3.0 32-bit
- Linux AMD64 RHEL 3.0 64-bit

- Sun Solaris 8 (SPARC) 32-bit and 64-bit
- Sun Solaris 10 x64 64-bit
- Windows x86 32-bit (2000 Service Pack 4 or later)

SBGSE 2.0 is supported on the following SSL Plus 5.0.4 platforms:

- HP-UX 11.11 32-bit and 64-bit
- IBM RS/6000 AIX 5.1 64-bit
- Windows x86 32-bit (2000 Service Pack 4 or later)
- IBM RS/6000 AIX 4.3.3 32-bit
- Sun Solaris 8 (SPARC) 32-bit and 64-bit

As a result of U.S. Federal regulations, Sybase is required to use FIPS-certified cipher suites. After you upgrade to SDK 12.5.1 and to Open Server 12.5.1, perform the following:

- For Sun Solaris 8 32-bit and 64-bit, HP-UX 11.11 32-bit and 64-bit, and IBM AIX 5.2 32-bit and 64-bit, be sure that the *lib3p* or *lib3p64* directory is in the dynamic load library path.
- In Windows, to allow the *ctlib* and *srvlib* libraries to find the Certicom cipher suites in *libsbgse2.dll*, add the following to the dynamic load library path:

  ```
  %SYBASE%\%SYBASE_OCS%\lib3p
  ```

  When running any application that uses SSL, be sure that the *lib3p* directory is in its dynamic load library path.

Note SSL Certificates generated by the certreq, certauth, certpk12 utilities on platforms such as, Digital Unix Tru64 5.0A, IBM RS/6000 AIX 4.3.3 32-bit, and Linux RH 7.2 32-bit, will not work with products (SDK/Open Server/ASE) that support SSL Plus 5.0.4. This will be addressed in the next ESD.

Windows SSPI support using MIT Kerberos

This feature provides a Generic Security Services (GSS) library interface to Windows Security Support Provider Interface (SSPI). It uses a *.dll* to allow Kerberos security driver to use the Windows Security SSPI routines instead of the CyberSafe GSS libraries.

To use this feature, you must edit the *csfkrb5* entry in the *libtcl.cfg* file to include the *libsspiwrapper.dll*.

For example:

```
csfkrb5=LIBSKRB secbase=@REALM libgss=C:\sybase\OCS-12_5\lib3p\libsspiwrapper.dll
```

For more information on Kerberos security services, see *Open Client and Open Server Configuration Guide for Windows*.

Note Windows SSPI does not provide support for keytab files.

New command line option for BCP

You can now use the new --skiprows parameter to skip the number of rows starting from the first row of an input file. This new feature provides BCP the ability to skip a specified number of rows before starting to copy from an input file. The valid range for --skiprows is between 0 and the actual number of rows in the input file. Providing an invalid value will display an error message.

Note The new --skiprows parameter cannot co-exist with the -F option.

bcp syntax changes

The following BCP parameter is included to specify the number of rows you want to skip:

```
--skiprows nSkipRows
```

where, *nSkipRows* is the numbers of rows you want to skip.

Example

In the following example, BCP will ignore the first two rows of the input file *titles.txt* and start to copy from the third row.

```
bcp pubs2..titles in titles.txt -U username -P password
--skiprows 2
```

New features in ESD #18

ESD #18 is only a bug-fixing release and has no new features.

New features in ESD #17

The following features are new for ESD #17:

Note ESQL/COBOL is now supported on HP Itanium 32-bit.

Extended support for OpenLDAP

OpenLDAP is now supported on Linux AMD64 (Opteron)/EM64T.

You can use the LDAP directory service to create, modify, and retrieve information from network entities.

To enable LDAP, modify the *libtcl.cfg* (32-bit) or *libtcl64.cfg* (64-bit) configuration files, both available in the *$SYBASE/$SYBASE_OCS/config* directory. Then, add an LDAP server to the configured directory service.

Further configuration instructions are provided in Chapter 5, "Using a Directory Service," of the *Open Client and Open Server Configuration Guide*.

XA 32-bit support on HP Itanium 32-bit

Extended Architecture (XA) 32-bit is now supported on HP Itanium 32-bit.

New features in ESD #16

The following features are new for ESD #16:

New logging feature for ASE OLE DB Provider

The new logging feature for ASE OLE DB Provider allows you to track an application’s database access by logging calls to the OLE DB public API.

You enable the logging feature by creating a registry entry that identifies a configuration file. The configuration file is a properties file that allows you to log all calls to OLE DB, or limit logging to a specified subset of the API.

❖ **To enable the logging feature**

1 Create a properties file. For example:

```
### --- Begin oledblog.properties ---
#-----------------------------------
#Configure loggers

# by default turn everything off. Change this to "=TRACE, OLEDB_LOGFILE"
# to default to logging to a file
log4cplus.rootLogger=OFF, NULL

# We want to log the OLEDB API calls to a log file
# change this to ="OFF,NULL" if you only want to log certain methods
log4cplus.logger.com.sybase.dataaccess.oledb=TRACE, OLEDB_LOGFILE
log4cplus.additivity.com.sybase.dataaccess.oledb=false

# We don't want to log the IUnknown::AddRef and Release calls
log4cplus.logger.com.sybase.dataaccess.oledb.IUnknown.AddRef=OFF, NULL
log4cplus.additivity.com.sybase.dataaccess.oledb.IUnknown.AddRef=false
log4cplus.logger.com.sybase.dataaccess.oledb.IUnknown.Release=OFF, NULL
log4cplus.additivity.com.sybase.dataaccess.oledb.IUnknown.Release=false

# Other OLEDB interface can be turned on and off as well

# You can turn on (or off) logging of all methods on a given interface
# by configuring:
# log4cplus.logger.com.sybase.dataaccess.oledb.<InterfaceName>=???

# You can get more specific and only log (or turn off logging for) a
# specific method:
log4cplus.logger.com.sybase.dataaccess.oledb.<InterfaceName>.
<MethodName>=TRACE, OLEDB_LOGFILE


#--------------------------------------
#Configure logging appenders

# Throw away all log messages
```

```
log4cplus.appender.NULL=log4cplus::NullAppender

# Send log messages to the console
log4cplus.appender.STDOUT=log4cplus::ConsoleAppender
log4cplus.appender.STDOUT.layout=log4cplus::PatternLayout
### Modify this line to define the output format
log4cplus.appender.STDOUT.layout.ConversionPattern=%m%n

# Write the log messages to a file
log4cplus.appender.OLEDB_LOGFILE=log4cplus::FileAppender
log4cplus.appender.OLEDB_LOGFILE.layout=log4cplus::PatternLayout
log4cplus.appender.OLEDB_LOGFILE.ImmediateFlush=true
### Modify this line to define the output format
log4cplus.appender.OLEDB_LOGFILE.layout.ConversionPattern=%m%n
### Modify this line to specify the file to save the log to.
log4cplus.appender.OLEDB_LOGFILE.File=c:\temp\oledb.log


### --- End oledblog.properties ---
```

Note For more information on creating a properties file, refer to the log4cplus project page at http://log4cplus.sourceforge.net.

2 Create a registry entry:

   HKEY_CURRENT_USER\Software\Sybase\OLEDB Provider

   LogConfigFile=<*path to properties file*>

3 Restart the application you want to log. Upon restart, the application begins to log to the log file specified in the properties file.

Note To disable the logging feature, delete the registry entry created in Step 2.

New *srv_send_data* routine added for UNIX platforms

Description

The new API routine, srv_send_data, allows Open Server applications to transfer rows containing multiple columns to clients. It allows Open Server applications to send text and image data in chunks, preventing excessive utilization of memory.

Syntax

```
CS_RETCODE srv_send_data(spp, column, buf, buflen)

SRV_PROC *spp;
CS_INT *column;
```

CS_BYTE *buf;
CS_INT buflen;

Parameters

spp

A pointer to an internal thread control structure.

column

The number of the column in a row set.

buf

A pointer to a buffer containing the data to send to the client. This determines the size of a section.

buflen

The length of the **buf* buffer.

Return Value

Table 3: Return values (srv_send_data)

Returns	To indicate
CS_SUCCEED	The routine completed successfully.
CS_FAIL	The routine failed.

Example

```
#include <ctpublic.h>
#include <ospublic.h>
/*
** Local Prototype.
*/
CS_RETCODE ex_srv_send_data PROTOTYPE((
   SRV_PROC *spp,
   CS_COMMAND *cmd,
   CS_INT cols
));
#define MAX_BULK 1024

/*
** EX_SRV_SEND_DATA

** Example routine to demonstrate how to write columns
** of data in a row set to a client using srv_send_data.
** This routine will send all the columns of data read
** from a server back to the client.

** Arguments:
** spp   A pointer to an internal thread control
** structure.
** cmd   The command handle for the command that is
** returning text data.
** cols  The number of columns in a row set.

** Returns:
** CS_SUCCEED Result set sent successfully to client.
** CS_FAIL An error was detected.
*/
CS_RETCODE ex_srv_send_data(spp, cmd, cols)
SRV_PROC *spp;
CS_COMMAND *cmd;
CS_INT cols;
{
   CS_INT *len;      /* Length of column data. */
   CS_INT *outlen;   /* Number of bytes received. */
   CS_BYTE **data;   /* Column data. */
   CS_BYTE buf[MAX_BULK]; /* Buffer for text data. */
   CS_BOOLok;        /* Error control flag. */
   CS_INT i;
   CS_INT ret;

   /* Initialization. */
   ok = CS_TRUE;

   /*
   ** Transfer a row.
   */
   for (i = 0; i < cols; i++)
   {
      if ((fmt[i].datatype != CS_TEXT_TYPE) &&
      (fmt[i].datatype != CS_IMAGE_TYPE))
      {
         /*
         ** Transfer a non TEXT/IMAGE column.
         */

         /*
         ** Read the data of a non-text/image column
         ** from the server.
         */
         ret = ct_get_data(cmd, i+1, data[i],
            len[i], &outlen[i]);
         if ((ret != CS_SUCCEED)
            && (ret != CS_END_DATA)
            && (ret != CS_END_ITEM))
         {
            ok = CS_FALSE;
            break;
      }

      /*
      ** Write the data of a non-text/image column
      ** to client.
      */
      if (ret = srv_send_data(srvproc, i+1, NULL,0)
         != CS_SUCCEED)
      {
         ok = CS_FALSE;
         break;
      }
   }
   else
   {
      /*
      ** Transfer a TEXT/IMAGE column in small
      ** trunks.
      */

      /*
      ** Read a chunk of data of a text/image column
      ** from the server.
      */
      while ((ret = ct_get_data(cmd, i+1, buf,
         MAX_BULK, &len[i])) == CS_SUCCEED)
      {
         /* Write the chunk of data to client. */
         if (ret = srv_send_data(srvproc, i+1, buf,
            len[i]) != CS_SUCCEED)
         {
            ok = CS_FALSE;
            break;
         }
      }
   }
}

switch(ret)
{
   case CS_SUCCEED:
   /* The routine completed successfully. */
   case CS_END_ITEM:
   /* Reached the end of this item’s value. */
   case CS_END_DATA:
   /* Reached the end of this row’s data. */
```

```
            break;
            case CS_FAIL:
            /* The routine failed. */
            case CS_CANCELED:
            /* The get data operation was cancelled. */
            case CS_PENDING:
            /* Asynchronous network I/O is in effect. */
            case CS_BUSY:
            /* An asynchronous operation is pending. */
            default:
               ok = CS_FALSE;
         }
         return (ok ? CS_SUCCEED : CS_FAIL);
      }
```

Note For more information on using srv_send_data, refer to ctos_procmultitextcol in the *ctos.c* sample program.

Usage

- srv_send_data is used to send a single column of data to a client.
- When sending columns with text or image data, Open Server applications must call srv_text_info before srv_send_data. This ensures the data stream is correctly set to the total length of data being sent. The application then calls srv_send_data to send the data in chunks, and continues to call the routine until there is no remaining data to be sent.
- Open Server applications can send text and image data to clients using srv_bind and srv_xferdata. However, these routines require all data columns to be sent at once. srv_send_data allows applications to send text and image data in chunks.
- Open Server applications treat text and image data streams identically, with the exception of character set conversions. These conversions are only performed on text data.

See also

Related srv_bind, srv_get_text, srv_text_info, srv_xferdata, srv_get_data, and srv_send_text routines in the Open Server 15.0 *Server Library/C Reference Manual*.

New feature in ESD #15

The following feature is new for ESD #15:

Dynamic version of bulk library available

Sybase now ships a dynamic version of bulk library for UNIX platforms. Bulk library is no longer directly dependent on Open Server libraries. Both the reentrant (*libblk_r.so*) and non-reentrant (*libblk.so*) libraries are available as shared libraries.

New features in ESD #14

The following features are new for ESD #14:

Note HP-UX 11.0 is not supported from ESD #14 onwards.

Asynchronous execution for ODBC

By default, drivers execute ODBC functions synchronously. That is, the application calls a function and the driver returns control to the application when execution is complete. With asynchronous execution, the driver returns control to the application after minimal processing and before execution is complete. This allows the application to execute in parallel other functions while the first function is still executing. Asynchronous execution is beneficial when a task is complex and requires a significant amount of time to execute.

For more information on asynchronous execution and its application, refer to MSDN ODBC Programmer's Reference at http://msdn2.microsoft.com/en-us/library/ms713563.aspx.

Note The ASE ODBC Driver by Sybase supports a maximum of one concurrent statement in asynchronous mode. Only one concurrent statement, synchronous or asynchronous, can be executed if server-side cursors are used or if the connection's auto-commit is disabled.

To use connection-level asynchronous execution with the ASE ODBC Driver by Sybase, call SQLSetConnectAttr and set *SQL_ATTR_ASYNC_ENABLE* to SQL_ASYNC_ENABLE_ON.

Note Calling SQLCancel when no processing is being done will not close the associated cursors. ODBC applications should explicitly call SQLFreeStmt or SQLCloseCursor to close cursors.

Server-Library routines updated

The following updates have been made to the Server-Library routines:

1. Server properties SRV_S_LOGFILE and SRV_S_TRUNCATELOG can now be set after calling srv_init.
2. After srv_init is called, setting the SRV_S_LOGFILE property with *bufp* set to an empty string ("") and *buflen* set to zero will close the log file.

New features in ESD #13

The following features are new for ESD #13:

Note Sun Solaris 10 x64 is supported from ESD #13 onwards.

New jConnect 6.05 connection properties

Table 4 lists new connection properties available for jConnect 6.05. For a complete list of connection properties, refer to Chapter 2, “Programming Information,” in the jConnect for JDBC 6.05 *Programmer’s Reference*.

Table 4: Connection properties for jConnect 6.05

Property	Description	Default value
CRC	When this property is set to *true*, the update counts that are returned are cumulative counts that include updates directly affected by the statement executed and any triggers invoked as a result of the statement being executed.	*false*
DATABASE	Use this property to specify the database name for a connection when the connection information is obtained from a Sybase *interfaces* file. The URL of an *interfaces* file cannot supply the database name.	*null*
DEFAULT_QUERY_TIMEOUT	When this connection property is set, it is used as the default query timeout for any statements created on this connection.	*0* (no timeout)
IMPLICIT_CURSOR_FETCH_SIZE	Use this property in conjunction with the SELECT_OPENS_CURSOR property to force jConnect to open a read-only cursor on every select query that is sent to the database. The cursor will have a fetch size of the value set in this property, unless overridden by the Statement.setFetchSize method.	*0*
INTERNAL_QUERY_TIMEOUT	Use this property to set the query timeout that will be used by statements internally created and executed by jConnect. This may prevent application hangs if internal commands do not complete in a reasonable time.	*0* (no timeout)
J2EE_TCK_COMPLIANT	When this property is set to *true*, the jConnect driver enables behavior that is compliant with the J2EE 1.4 TCK test suite, which causes some loss of performance. Therefore, Sybase recommends using the default value of *false*.	*false*

New jConnect 5.5 connection properties

Table 5 lists new connection properties available for jConnect 5.5. For a complete list of connection properties, refer to Chapter 2, “Programming Information,” in the jConnect for JDBC 5.5 *Programmer’s Reference.*

Table 5: Connection properties for jConnect 5.5

Property	Description	Default value
CACHE_COLUMN_METADATA	Use this property to allow jConnect to cache ResultSet Metadata on consecutive executions when DYNAMIC_PREPARE is set to *true*. This helps to improve performance.	False
CAPABILITY_TIME	Use this property only when JCONNECT_VERSION is greater than or equal to 6. When jConnect is connected to a server that supports the TIME datatype, all parameters of type java.sql.Time or escape literals {t ...} are processed as TIME. The previous versions of jConnect treat such parameters as DATETIME and prefixes "1970-01-01" to the java.sql.Time parameter. The date also gets stored in the database if the underlying datatype is datetime or smalldatetime. In the new versions of jConnect, when TIME is processed, the server converts TIME to the underlying datatype and adds its own base year as a prefix. This can result in incompatibilities between the old and new data. If you are using datetime or smalldatetime datatypes for java.sql.Time, you should leave CAPABILITY_TIME as *false* for backward compatibility. Leaving this property to be set as *false* forces jConnect to process java.sql.Time parameters or escape literals {t ...} as DATETIME regardless of the server capability of handling TIME datatype. Setting this property to *true* causes jConnect to process the java.sql.Time parameters as TIME datatype when connected to ASE 12.5.1 or later. Sybase recommends that you leave this property as *false* if you are using smalldatetime or datetime columns to store time values.	False

Property	Description	Default value
CAPABILITY_WIDETABLE	This property can be set to *false* as a performance improvement, if you do not require JDBC ResultSetMetaData like column name. This results in lesser data exchange over the network, thereby increasing performance. Sybase recommends that you use the default setting unless you are using EAServer. See "Using wide table support for ASE 12.5 and later," in Chapter 2 of the *jConnect for JDBC Programmer's Reference*.	True
CRC	When this property is set to *true*, the update counts that are returned are cumulative counts that include updates directly affected by the statement executed and any triggers invoked as a result of the statement being executed.	False
DATABASE	Use this property to specify the database name for a connection when the connection information is obtained from a Sybase *interfaces* file. The URL of an *interfaces* file cannot supply the database name.	Null
IMPLICIT_CURSOR_FETCH_SIZE	Use this property in conjunction with the SELECT_OPENS_CURSOR property to force jConnect to open a read-only cursor on every select query that is sent to the database. The cursor will have a fetch size of the value set in this property, unless overridden by the Statement.setFetchSize method.	0
QUERY_TIMEOUT_CANCELS_ALL	Use this property to force jConnect to cancel all statements on a connection when a read timeout is encountered. This behavior can be used when a client calls execute() and a timeout occurs because of a deadlock, for example, when it tries to read from a table that is being updated in another transaction. Depending on future discussions with Sun, this property may be combined with the property values affected by BE_AS_JDBC_COMPLIANT_AS_POSSIBLE property.	False

Property	Description	Default value
SERVER_INITIATED_TRANSACTIONS	This property allows the server to control transactions. By default, the property is set to *true,* and jConnect allows the server to start and control transactions by using Transact-SQL® set chained on. If the property is set to *false*, the transactions are started and controlled by jConnect using transact sql begin tran. Sybase recommends that you allow the server to control the transactions.	True
SERVERTYPE	This property can be set to "OSW" when you are connected to OpenSwitch™. This allows jConnect to send certain instructions to OpenSwitch, which allows OpenSwitch to remember initial connection settings, such as isolation level, textsize, quoted identifier, and autocommit, even when it is moved to a different server instance.	None
TEXTSIZE	Use this property to set the TEXTSIZE. By default, ASE and ASA allow 32,627 bytes to be read from an image or text column. jConnect changes the value to 2GB if you have the jConnect mda tables installed. However, setting this value when connected to OpenSwitch allows the connection to remember the settings when OpenSwitch is moved to a different server instance.	2GB

Documentation updates and clarifications to ESD #11

Documentation for the “Directory services sql.ini and interface file support” feature in ESD #11 has been updated. For more information, see “Directory services sql.ini and interface file support” on page 39.

New features in ESD #12

The following features are new for ESD #12:

SSL Plus support on Linux AMD64 (Opteron)/EM64T and HP Itanium 32-bit and 64-bit

Sybase now supports SSL Plus on the following platforms:

- SSL5.0.4m on Linux AMD64 (Opteron)/EM64T
- SSL5.0.6f on HP Itanium 32-bit
- SSL5.9.6h on HP Itanium 64-bit

The new SSL Plus support applies to the following Sybase products:

- Software Developer's Kit components:
    - Client-Library
    - ESQL/C
- Open Server

MIT Kerberos support

MIT Kerberos 5 on HP-UX 11.11 (or HP-UX 11iv1.0) 64-bit and IBM AIX 64-bit is rebased from version 1.3.6 to 1.4.3.

MIT Kerberos 5 version 1.3.6 is now supported on HP-UX 11.11 (or HP-UX 11iv1.0) 32-bit and HP-UX 11.0 32-bit.

Table 9 lists releases of MIT Kerberos version 5 on platforms currently supported by Sybase.

Table 6: MIT Kerberos version 5 releases and supported platforms

MIT Kerberos version 5	Platform
Release 1.4.3	• HP-UX 11.11 (or HP-UX 11iv1.0) 64-bit • IBM AIX 64-bit
Release 1.4.1	• Sun Solaris 9 x86 (32-bit)
Release 1.3.6	• HP-UX 11.11 (or HP-UX 11iv1.0) 32-bit • HP-UX 11.0 32-bit • Microsoft Windows (including Windows NT, Windows 2000, Windows 2003, and Windows XP)
Release 1.3.1	• Linux Intel 32-bit • Sun Solaris 8 (SPARC 32-bit and 64-bit)
Release 1.2.7	• Linux AMD64 (Opteron)/EM64T

To use and configure MIT Kerberos security services for the above platforms, refer to the following documents:

- The chapter and appendixes in the *Open Client and Open Server Configuration Guide for UNIX*:
  - Chapter 6, “Using Security Services”
  - Appendix B, “Configuration Files”
  - Appendix E, “Kerberos Security Services”
- The chapter and appendix in the *Open Client and Open Server Programmer’s Supplement for UNIX*:
  - Chapter 1, “Open Client Client-Library/C”
  - Appendix B, “Environment Variables”
- The chapter and appendix in the *Open Client and Open Server Configuration Guide for Windows*:
  - Chapter 6, “Using Security Services”
  - Appendix B, “Configuration Files”
- The chapters in the *Open Client and Open Server Programmer’s Supplement for Windows*:
  - Chapter 1, “Building Open Client and Open Server Applications”
  - Chapter 2, “Client Library/C Example Programs”

MIT Kerberos on DB-Library

The MIT Kerberos security mechanism is now available on DB-Library, providing network and mutual authentication services. This feature allows older Sybase applications to be Kerberized to use Kerberos authentication services, with lesser need for modification and recompilation.

The following DB-Library macros were added to enable Kerberos support:

- DBSETLNETWORKAUTH: used to enable or disable network base authentication.
- DBSETLMUTUALAUTH: used to enable or disable mutual authentication of the connection's security mechanism.
- DBSETLSERVERPRINCIPAL: used to set the server's principal name, if required.

For information on installing MIT Kerberos on DB-Library, refer to the *Installation and Release Bulletin* Sybase SDK DB-Lib Kerberos Authentication Option 12.5.1.

Note DB-Library only supports network authentication and mutual authentication services in the Kerberos security mechanism.

ESQL/COBOL support on HP Itanium 32-bit

ESQL/COBOL is now supported on HP Itanium 32-bit. The COBOL compiler for HP Itanium 32-bit is Micro Focus Server Express 4.0 SP2.

ASE OLE DB Provider participation in Distributed Transactions

This feature is supported only on Windows and requires that Microsoft Distributed Transaction Coordinator (MS DTC) be the transaction coordinator managing Distributed Transactions.

Sybase supports the following programming models:

- Applications using MS DTC directly
- Applications using Microsoft Transaction Server (MTS) or (COM+)

Programming for MS DTC

❖ **To program using Microsoft Distributed Transaction Coordinator (MS DTC)**

1. Connect to MS DTC using the DtcGetTransactionManager function. For information about MS DTC, see Microsoft Distributed Transaction Coordinator documentation.
2. Get the IDBSession for each Sybase ASE connection you want to establish following the OLE DB steps.
3. Call the ITransactionDispenser::BeginTransaction function to begin an MS DTC transaction and to obtain an OLE Transaction object that represents the transaction.
4. Query ITransactionJoin from each IDBSession (OLE DB Connection) you want to enlist in the MS DTC transaction, and call JoinTransaction with the passed in parameter punkTransactionCoord as the Transaction object (obtained in Step 3). Currently, Sybase supports only the isolation level of ISOLATION LEVEL_READCOMMITTED for the distributed transaction, and does not support ITransactionOptions.
5. To update a SQL Server, follow the OLE DB steps for creating and executing the IDBCommand.
6. Call the ITransaction::Commit function to commit the MS DTC transaction. The Transaction object is no longer valid.

Programming components deployed in MTS or COM+

The following procedure describes how to create components that participate in Distributed Transactions in MTS or COM+.

❖ **To program components deployed in MTS or COM+**

1. Create an IDBSession for each ASE connection.
2. Create and execute IDBCommand for each update you would like to perform.
3. Deploy your component to MTS or COM, and configure the transaction attributes as needed.

The COM+, OLE DB Services, and OLE DB provider will take care of creating the transaction, participating in the transaction, and committing or rolling back the transaction.

OLE DB Services is needed for the Automatic Transaction Enlistment. To enable the OLE DB Services, you must follow some rules to initialize the Data Source (see the MS OLE DB documents). To enable the automatic transaction enlistment, you can set the bit DBPROPVAL_OS_TXNENLISTMENT in the *OLE_DB_SERVICES* registry and in the DBPROP_INIT_OLEDBSERVICES property value, or pass OLE DB Services = *2* in the connection string.

Connection properties for Distributed Transaction support

The following describes the connection properties:

- Distributed Transaction Protocol (DistributedTransactionProtocol) – To specify the protocol used to support the distributed transaction, either use the XA Interface standard or MS DTC OLE Native protocol, select the Distributed Transaction Protocol in the OLE DB Data Source Dialog, set the property DistributedTransactionProtocol = *OLE* in the provider string part of the connection string for OLE Native protocol, or the use default protocol *XA*.
- Tightly Coupled Transaction (TightlyCoupledTransaction) – When a distributed transaction using two resource managers points to the same ASE server, you may have a situation called “Tightly Coupled Transaction.” Under these conditions, if you do not set this property to 1, the Distributed Transaction may fail.

To summarize, if you open two database connections to the same ASE server and then enlist these connections in the same distributed transaction, Sybase recommends that you set TightlyCoupledTransaction=1. To set this property, select the Tightly Coupled Transaction in the OLE DB Data Source dialog box, or pass the property TightlyCoupledTransaction=1 in the provider string part of the connection string.

OLE DB DSN Migration tool available

The OLE DB DSN Migration tool helps you to migrate your data source definitions (DSNs) from the OLE DB Driver Kit to the OLE DB Provider by Sybase. When you migrate the DSNs, they will use the new OLE DB Provider by Sybase instead of the OLE DB Driver Kit.

Migrating to ASE OLE DB Provider by Sybase

To migrate OLE DB applications to use the ASE OLE DB Provider by Sybase, you must edit the connection string used by OLE DB client applications. The provider short name for the ASE OLE DB Provider by Sybase is "ASEOLEDB."

Known differences in behavior between the OLE DB Driver Kit and ASE OLE DB Provider by Sybase are documented in the SDK 12.5.1 *Release Bulletin* for your platform.

Note The connection string syntax for ASE OLE DB Provider by Sybase is documented in the Adaptive Server Enterprise OLE DB Provider by Sybase *User's Guide* for Microsoft Windows.

The connection string syntax differs from the syntax for the OLE DB Driver Kit. The OLE DB Provider by Sybase honors the OLE DB Driver Kit syntax, but Sybase recommends that you migrate your connection string syntax to the new syntax when possible.

Migrating Data Source Names to Sybase drivers

There are two methods to migrate Data Source Names (DSNs) from the OLE DB Driver Kit to the drivers created by Sybase:

- Using the Sybase ASE Data Source Administrator
- Using the DSN Migration tool

Using the Sybase ASE Data Source Administrator

The Sybase ASE Data Source Administrator is a GUI process that allows you to migrate existing OLE DB Driver Kit data sources and to create new data sources for the ASE OLE DB Provider.

❖ To migrate the data sources using the Data Source Administrator

1. On the main window titled "Sybase Data Source Administrator," choose the data source.
2. Click Migrate.

The Sybase Data Source Administrator allows you to add, remove, configure, or test the OLE DB data sources.

Using the DSN Migration tool

The DSN Migration tool can help you migrate the data sources from the OLE DB Driver Kit to the OLE DB Driver by Sybase.

The dsnmigrate tool uses switches to control which DSNs are migrated. You need to enter the following from the command line:

```
dsnmigrate.exe [/?|/h|/help][/oledb]
[/l|/ul|/sl][/a|/ua|/sa] [[/dsn|/udsn|/sdsn]=dsn]
[/suffix=suffix]
```

For all converted OLE DB DSNs, the new Sybase DSNs will have the same name.

Conversion switches

The following table lists and describes the switches used in the conversion.

Table 7: Conversion switches

Switches	Description of results
/?,/h,/help	Displays this message. This message is also displayed if dsnmigrate is called with no command line arguments.
/oledb	Places dsnmigrate into OLEDB-mode. By default, ODBC DSNs are migrated.
/l	Displays a list of all OLE DB Driver Kit user and system DSNs.
/ul	Displays a list of all OLE DB Driver Kit user DSNs.
/sl	Displays a list of all OLE DB Driver Kit system DSNs.
/a	Converts all OLE DB Driver Kit user and system DSNs.
/ua	Converts all OLE DB Driver Kit user DSNs.
/sa	Converts all OLE DB Driver Kit system DSNs.
/dsn	Converts specific OLE DB Driver Kit user or system DSNs.
/udsn	Converts specific OLE DB Driver Kit user DSNs.
/sdsn	Converts specific OLE DB Driver Kit system DSNs.
dsn	The name of the DSN to be converted.
/suffix	An optional switch that changes the way DSNs are named. If this switch is used, the original DSN is retained and the new DSN is named “<dsn>-<suffix>.”
suffix	The suffix that is used to name the new DSN.

HA failover on ASE OLE DB Provider

Table 8 lists the new connection parameters used for High Availability (HA) failover support on the ASE OLE DB Provider:

Table 8: HA failover connection parameters

Property names	Description	Required	Default value
HASession	Specifies if high availability is enabled. 0 indicates high availability disabled, 1 high availability enabled.	No	0
SecondaryPort	The port number of the ASE server acting as a failover server in an active-active or active-passive setup.	Yes, if HASession is set to 1.	Empty
SecondaryServer	The name or the IP address of the ASE server acting as a failover server in an active-active or active-passive setup.	Yes, if HASession is set to 1.	Empty

Using failover in HA systems

A HA cluster includes two or more machines that are configured so that if one machine (or application) is interrupted, the second machine assumes the workload of both machines. Each of these machines is 1 node of the high availability cluster. A HA cluster is used in an environment that must always be available, such as, a banking system to which clients must connect continuously, 365 days a year.

The machines are configured so that each machine can read the other machine's disks, although not at the same time (all of the disks that are failed-over should be shared disks).

For example, if Adaptive Server 1 is the primary companion server, and it crashes, Adaptive Server 2, as the secondary companion server, reads its disks (disks 1 - 4) and manages any databases on them until Adaptive Server 1 can be rebooted. Any clients connected to Adaptive Server 1 are automatically connected to Adaptive Server 2.

Failover allows Adaptive Server to work in a high availability cluster in active-active or active-passive configuration.

During failover, clients connected to the primary companion using the failover property automatically reestablish their network connections to the secondary companion. Failover can be enabled by setting the connection property HASession to "1" (default value is "0"). If this property is not set, the session failover does not occur, even if the server is configured for failover. You also must set SecondaryServer (the IP address or the machine name of the secondary ASE server) and SecondaryPort (the port number of the secondary ASE server) properties. See the ASE book, *Using Sybase Failover in a High Availability System*, for information about configuring your system for HA.

When the OLE DB Provider driver detects a connection failure with the primary ASE server, it first tries to reconnect to the primary. If it cannot reconnect, it assumes that a failover has occurred. Then, it automatically tries to connect to the secondary ASE server using the connection properties set in SecondaryServer, and SecondaryPort.

Confirming a successful failover

If a connection to the secondary ASE server is established, the ASE OLE DB Driver returns "E_FAIL" for the function return HRESULT.

To confirm a successful failover, you should examine the dwMinor field in ERRORINFO (returned from IErrorRecords::GetBasicErrorInfo), or the description returned from IErrorInfo::GetDescription. The dwMinor value should be "30130" for a successful HA failover. The description from IErrorInfo::GetDescription should be as follows, where *ASEServerName* is the server name failed over to:

```
"Sybase server is not available or has terminated your
connection, you have successfully connected to the next
available HA server ASEServerName. All transactions has
been rolled back."
```

Note Sybase recommends that you check the code returned by dwMinor to determine the success of the failover rather than through examination of the error description.

The client must then reapply the failed transaction with the new connection. If failover happens while a transaction is open, only the changes that were committed to the database before failover are retained.

Verifying an unsuccessful failover

If the connection to the secondary server is not established, the ASE OLE DB Driver also returns “E_FAIL” for the function return HRESULT. However, the dwMinor field in ERRORINFO (returned from IErrorRecords::GetBasicErrorInfo) should be “30131”, and the description returned from IErrorInfo::GetDescription should be:

```
“Connection to Sybase server has been lost, connection
to the next available HA server also failed. All
transactions have been rolled back.”
```

Sample code for checking failover

The following code snippet shows how to code for a failover:

```
/* Declare required variables */
...
/* Open Database connection */
...
/* Perform a transaction */
...
/*Check HRESULT  and dwMinor in
  ERRORINFO, handle failover */
if (FAILED(hr))
{
   IErrorInfo* pIErrorInfo;
   GetErrorInfo(0, &pIErrorInfo);
   IErrorRecords * pIErrorRecords;
   HRESULT hr1 = pIErrorInfo->QueryInterface
      (IID_IErrorRecords,(void **)&pIErrorRecords);

   if (SUCCEEDED(hr1))
   {
      ERRORINFO errorInfo;
      pIErrorRecords->GetBasicErrorInfo(0,
         &errorInfo);
      pIErrorRecords->Release();
      if (errorInfo.dwMinor == 30130)
      {
         //successful failover,
         //retry the transaction
      }
   }
}
```

New features in ESD #11

The following features are new for ESD #11:

MIT Kerberos support on Sun Solaris 9 x86 (32-bit)

MIT Kerberos 5 version 1.4.1 is now supported on Sun Solaris 9 x86 (32-bit).

Table 9 lists releases of MIT Kerberos version 5 on platforms currently supported by Sybase.

Table 9: MIT Kerberos version 5 releases and supported platforms

MIT Kerberos version 5	Platform
Release 1.4.1	• Sun Solaris 9 x86 (32-bit)
Release 1.3.6	• IBM AIX 64-bit • HP-UX 11.11 (or HP-UX 11iv1.0) 64-bit • Microsoft Windows (including Windows NT, Windows 2000, Windows 2003, and Windows XP)
Release 1.3.1	• Linux Intel 32-bit • Sun Solaris 8 (SPARC 32-bit and 64-bit)
Release 1.2.7	• Linux AMD64 (Opteron)/EM64T

Directory services sql.ini and interface file support

This new feature supports the use of the *sql.ini* file (for Windows) and the *interfaces* file (for UNIX) to provide server information. It is supported for the following drivers and providers:

- ADO.NET Data Provider
- ODBC Driver
- OLE DB Provider

Currently, to connect to the driver or provider you must specify several properties, such as server name or IP address and port number of an ASE server. By using the *sql.ini* or *interfaces* file, enterprises can centralize the information about the services available in the enterprise networks including, ASE server information.

Connection string

The following must be added to the connection string to identify the *sql.ini* or *interfaces* file. You can connect to a single Directory Services URL (DSURL) or to multiple DSURLs.

Connection string for a single DSURL

For a single DSURL for ADO.NET, ODBC, and the OLE DB drivers and providers, you must add the following properties to the connection string in the following format:

```
DSURL=file://[path]<filename>[?][servicename]
```

where:

- *path* (optional) is the path to the interfaces file. If not specified, *%SYBASE%\ini* is used as the default path on Windows and *$SYBASE* on Linux and Mac OS X.
- *filename* is the name of the interface file.
    - On Windows, the interfaces file is typically named *sql.ini*.
    - On UNIX, the interfaces file is typically named *interfaces*.
- *servicename* (optional) is the name of the service defined by the DSURL, also known as the server name.
    - If not defined in the DSURL, the server property in the connection string is used.
    - If both service name and server exist, the service name will be used.
    - If neither of them exist, the driver or provider loads the server information from the *sql.ini* or *interfaces* file, if the *sql.ini* or *interfaces* file has only one entry. If more than one entry exists in the file, an error is returned.

      ODBC error example:

```
[Sybase][ODBC Driver]Getting more than one servers with the connect
string.
```

Therefore the servicename or server property should always be defined in the connection URL, as typically interfaces files will contain multiple server entries.

Examples of the connection string for Windows and UNIX:

Windows

Example using a default path:

```
DSURL=file://sql.ini?mango1
```

where:

- path is omitted in the above example, hence *%SYBASE%/ini* is used as the default path. However, if %SYBASE% is defined as *C:\Sybase*, then *C:\Sybase\ini* will be the path used.
- filename is *sql.ini*.
- servicename is *mango1*.

Example using explicit path:

```
DSURL=file://\\myServer\myShare\sql.ini?mango1
```

where:

- path is specified as *\\myServer\myShare*, which is a Universal Naming Convention (UNC) path that refers to a network shared directory.
- filename is *sql.ini*.
- servicename is *mango1*.

Linux and Mac OS X

Example using a default path:

```
DSURL=file://interfaces?mango1
```

where:

- path is omitted in the above example, hence $SYBASE is used as the default path. However, if $SYBASE is defined as */usr/sybase*, then */usr/sybase* will be the path used.
- filename is *interfaces*.
- servicename is *mango1*.

Example using explicit path:

```
DSURL=file:///remote/sybase/interfaces?mango1
```

where:

- path is specified as */remote/sybase*
- filename is *interfaces*.
- servicename is *mango1*.

Connection string for multiple URLs

DSURL can support multiple LDAP URLs and *interfaces* files. Sybase also supports a multiple URL with mixed LDAP URLs and file URLs. When multiple URLs are used, the driver processes the URLs one by one until it successfully opens an *interfaces* file or connects to an LDAP server.

Examples of multiple URLs

The following are examples of using multiple URLs in a connection string:

```
DSURL={file:///mils1/sybase/sql.ini;file:///test/interface}
```

```
DSURL={ldap://SYBLDAP:389/dc=sybase,dc=com??one?sybaseServername=MANGO;
file:///test/interface?MANGO}
```

Format of the interfaces file for Microsoft Windows, UNIX, and SSL

For information on the format of the interfaces file, see the *Adaptive Server Enterprise Configuration Guide* for your platform.

ODBC DSN Migration tool

The ODBC DSN Migration tool can help you migrate from the ODBC Driver Kit to the ODBC Driver by Sybase. When you migrate your DSNs, they begin using the new ODBC Driver by Sybase, instead of the ODBC Driver Kit.

Using the Migration tool

The dsnmigrate tool uses switches to control which DSNs are migrated. You need to enter the following from the command line:

```
dsnmigrate.exe [/?|/help] [l|/ul|/sl][/a|/ua|/sa]
[[/dsn|/udsn|/sdsn]=dsn] [/suffix=suffix]
```

All DSNs that are migrated are renamed to "<*dsn*>-backup" before the conversion is completed. When the new Sybase DSNs are created and the conversion is completed, the name is changed to "<*dsn*>," which will allow existing applications to continue to run without any modifications.

Conversion switches

The following table lists and describes the switches used in the conversion.

Table 10: Conversion switches

Switches	Description of results
/?,/h,/help	Displays this message. This message is also displayed if dsnmigrate is called with no command-line arguments.
/l	Displays a list of all ODBC Driver Kit users and system DSNs.
/ul	Displays a list of all ODBC Driver Kit user DSNs.
/sl	Displays a list of all ODBC Driver Kit system DSNs.
/a	Converts all ODBC Driver Kit users and system DSNs.
/ua	Converts all ODBC Driver Kit user DSNs.
/sa	Converts all ODBC Driver Kit system DSNs.
/dsn	Converts specific ODBC Driver Kit users or system DSNs.
/udsn	Converts specific ODBC Driver Kit users DSNs.
/sdsn	Converts specific ODBC Driver Kit system DSNs.
dsn	Specifies the name of the DSN to be converted.
/suffix	An optional switch that changes the way DSNs are named. If this switch is used, the original DSN is retained and the new DSN is named "<*dsn*>-<*suffix*>."
suffix	Specifies the suffix that names the new DSN.

Bookmark and bulk support for ODBC and OLE DB

Sybase supports bookmarks and SQL bulk operations for the ODBC Driver and the OLE DB Provider.

For the ODBC Driver

Bulk insertions that use SQLBulkOperations with the option of SQL_ADD and cursor positioned updates and deletes using SQLSetPos (SQL_UPDATE, SQL_DELETE, SQL_POSITION). For instructions on using SQL_ADD and SQLSetPos, refer to the Microsoft Developer Network library's *Programmer's Reference*:

- SQL_ADD at http://msdn2.microsoft.com/en-us/library/ms712550.aspx
- SQLSetPos at http://msdn2.microsoft.com/en-us/library/ms713507.aspx

For the OLE DB Provider

Bookmark operations that use the IRowsetLocate interface, which provides methods for comparing bookmarks and retrieving rows, based on bookmarks. For instructions on using IRowsetLocate, refer to MSDN OLE DB Programmer's Reference at http://msdn2.microsoft.com/en-us/library/ms721190.aspx.

Directory services sql.ini and interface file support on jConnect

This new feature supports the use of the *sql.ini* file (for Windows) and the *interfaces* file (for UNIX) to provide server information for jConnect for JDBC.

Currently, to connect to the driver or provider you must specify several properties, such as server name or IP address and port number of an ASE server. By using the *sql.ini* or *interfaces* file, enterprises can centralize the information about the services available in the enterprise networks including, ASE server information.

Connection string

The following must be added to the connection string to identify the *sql.ini* or *interfaces* file. On jConnect for JDBC, you may only connect to a single Directory Services URL (DSURL).

Note jConnect does not support multiple URLs.

Connection string for a single DSURL for jConnect

For a single DSURL for jConnect, add the following properties to the connection string in the following format:

Note The user must specify the path to the *sql.ini* file and the server name.

```
String url = "jdbc:sybase:jndi:file://D:/syb1252/ini/sql.ini?mango1”
```

where:

- server name = *mango1*.
- *sql.ini* file path = *file://D:/syb1252/ini/sql.ini*

If the *sql.ini* path or server name is not specified in the URL, the driver returns an error.

Format of the interfaces file for SSL

The following is the format for the *sql.ini* file for SSL:

```
[SYBSRV2]
master=nlwnsck,mango1,4100,ssl
query=nlwnsck,mango1,4100,ssl
```

```
query=nlwnsck,mango1,5000,ssl
```

Note jConnect supports multiple query entries under the same server name in the *sql.ini* file. jConnect attempts to connect to values for host or port from the query entry in the sequence, as in the *sql.ini* file. If jConnect finds an SSL in a query entry, it will require the application to be coded to handle SSL connections by specifying an application specific socket factory, or the connection may fail.

New features in ESD #10

The following features are new for ESD #10:

Support for HP-UX 11.11 (or HP-UX 11iv1.0) 32-bit and 64-bit platforms

HP-UX 11.11 (or HP-UX 11iv1.0) 32-bit and 64-bit platforms are now supported on Sybase SDK 12.5.1 and Open Server 12.5.1 products, as described in Table 11.

Table 11: SDK and Open Server 12.5.1 new platforms and supported features

Platform	Operating system level	C and C++ compilers	COBOL compilers	Kerberos version	Light-weight Directory Access (LDAP)	Secure Sockets Layer (SSL)
HP-UX 11.11 (or HP-UX 11iv1.0) 32-bit	HP-UX 11i v1	HP C/ANSI C B.11.11.10 HP ANSI C++ B3910B A.03.10	Micro Focus Server Express 4.0	CyberSafe Trust Broker 2.1, MIT 1.4.1	Netscape LDAP 4.1	Certicom SSL Plus 3.1.14
HP-UX 11.11 (or HP-UX 11iv1.0) 64-bit	HP-UX 11i v1 with Patch bundle 99OP	HP C 11.11.10 ANSI HP ANSI C++ B3910B A.03.10	Not available	MIT 1.4.1	Netscape LDAP 4.1	Certicom SSL Plus 3.1.14

XA 64-bit support on Linux AMD64 (Opteron)/EM64T

XA 64-bit is now supported on Linux AMD64 (Opteron)/EM64T.

MIT Kerberos support on Microsoft Windows

MIT Kerberos 5 version 1.3.6 is now supported on Microsoft Windows (including Windows NT, Windows 2000, Windows 2003, and Windows XP).

Scrollable cursors

The ASE ODBC Driver and the ASE OLE DB Provider now support scrollable cursors.

Using scrollable cursors

Scrollable cursors allow applications to set the current position of the cursor anywhere in the resultset by specifying appropriate scrolling options. Applications can use one of NEXT, PRIOR, FIRST, LAST, RELATIVE, or ABSOLUTE scrolling options to traverse the resultset as desired.

When scrollable cursors are requested, either a server-side or client-side scrollable cursor may be invoked. Adaptive Server must support scrollable cursors in order for a server-side cursor to be invoked. If Adaptive Server does not support scrollable cursors, the desired functionality may be mimicked by the driver using cached resultset. This is known as client-side scrollable cursors.

The ASE ODBC Driver and the ASE OLE DB Provider can support both server-side and client-side scrollable cursors on ASE version 15.0 and later. But only client-side scrollable cursors are supported on all pre-15.0 ASE versions.

The UseCursor property must be set correctly in order to obtain the desired scrollable cursor.

Warning! Client-side scrollable cursors use more memory on the client side and may generate higher network traffic.

Setting the UseCursor connection property

How you set the UseCursor connection property determines whether client-side or server-side scrollable cursors are used:

- When you set the UseCursor connection property to 1, and the ASE version is 15.0 or later, server-side scrollable cursors are used.

Note Server-side scrollable cursors are not available on all pre-15.0 ASE versions.

- When you set the UseCursor connection property to 0, client-side scrollable cursors (cached resultsets) are used, regardless of the ASE version.

For the ASE ODBC Driver

The following describes scrollable cursors for the ASE ODBC Driver.

Support for the Static Insensitive scrollable cursor

The ASE ODBC Driver supports the Static Insensitive scrollable cursor. It implements the ODBC SQLFetchScroll method to scroll and fetch rows. The SQLFetchScroll method is a standard ODBC method defined in *Microsoft Open Database Connectivity Software Development Kit Programmer's Reference, Volume 2*, which is part of the MSDN library. Go to the Microsoft Web site at http://msdn.microsoft.com/en-us/default.aspx for more information.

The ODBC driver supports the following scrolling types:

- SQL_FETCH_NEXT – return the next rowset.
- SQL_FETCH_PRIOR – return the prior rowset.
- SQL_FETCH_RELATIVE – return the rowset *n* from the start of the current rowset.
- SQL_FETCH_FIRST – return the first rowset in the result set.
- SQL_FETCH_LAST – return the last complete rowset in the result set.
- SQL_FETCH_ABSOLUTE – return the rowset starting at row *n*.

Setting scrollable cursor attributes

You must set the following attributes to use scrollable cursors:

- SQL_ATTR_CURSOR_SCROLLABLE – the type of scrollable cursor you are using. It should be set to the value of SQL_SCROLLABLE. Possible values are static, semi-sensitive, insensitive.

- SQL_ATTR_CURSOR_SENSITIVITY – the sensitivity value for this scrollable cursor.

Note The only supported value for this is SQL_INSENSITIVE.

The following are *optional* attributes when using scrollable cursors.

- SQL_ATTR_ROW_ARRAY_SIZE – the number of rows that you want returned from each call to the SQLFetchScroll() method.

Note If you do not set this value, the default value of one row is used.

- SQL_ATTR_CURSOR_TYPE – The type of scrollable cursor you are using.

Note The only supported values for this are SQL_CURSOR_FORWARD_ONLY or SQL_CURSOR_STATIC.

- SQL_ATTR_ROWS_FETCHED_PTR – the address where the number of rows fetched are stored. The SQL_ATTR_ROWS_FETCHED_PTR points to a variable of data type SQLUINTEGER.
- SQL_ATTR_ROW_STATUS_PTR – the address where the row status is stored. The SQL_ATTR_ROW_STATUS_PTR points to a variable of data type SQLUSMALLINT.

Executing scrollable cursors

❖ To set up a program to execute a scrollable cursor

1. Set the scrollable cursor attributes for your environment.

   See "Setting scrollable cursor attributes" on page 48 for more information.

2. Bind the results. For example, add the following to your program:

```
res=SQLBindCol(m_StatementHandle, 2, SQL_C_DOUBLE, price, 0, NULL);
res=SQLBindCol(m_StatementHandle, 3, SQL_C_LONG, quantity, 0, NULL);
```

3. Scroll and fetch by using SQLFetchScroll(). For example, add the following to your program:

```
res = SQLSetStmtAttr(m_StatementHandle, SQL_ATTR_CURSOR_SCROLLABLE,
(SQLPOINTER)SQL_SCROLLABLE,SQL_IS_INTEGER);

res = SQLSetStmtAttr(m_StatementHandle, SQL_ATTR_CURSOR_SENSITIVITY,
SQLPOINTER)SQL_INSENSITIVE, SQL_IS_INTEGER);

res = SQLFetchScroll(m_StatementHandle, SQL_FETCH_NEXT,0);
res = SQLFetchScroll(m_StatementHandle, SQL_FETCH_PRIOR,0);
res = SQLFetchScroll(m_StatementHandle, SQL_FETCH_FIRST,0);
res = SQLFetchScroll(m_StatementHandle, SQL_FETCH_LAST,0);
res = SQLFetchScroll(m_StatementHandle, SQL_FETCH_ABSOLUTE,2);
res = SQLFetchScroll(m_StatementHandle, SQL_FETCH_ABSOLUTE,-2);
res = SQLFetchScroll(m_StatementHandle, SQL_FETCH_RELATIVE,1);
```

4 Execute the Select statement. For example, add the following to your program:

```
res = SQLExecDirect(m_StatementHandle, (SQLCHAR "select price, quantity
from book" SQL_NTS);
```

5 Close the result set and the cursor. For example, add the following to your program:

```
res = SQLFreeStmt(m_StatementHandle,SQL_CLOSE);
```

Looking at results

After you execute a scrollable cursor, you will see these results, assuming a total of *N* rows and a rowset *m* where *N* > *m*:

Result	Interpretation
Absolute 0	No row is returned, error.
Absolute 1	*m* row is returned.
Absolute *N*	1 row is returned.
Absolute *N*+1	No row is returned, error.
First	The first (1..*m*) rows are returned.
Last	The last (*N*-*m*+1 .. *N*) rows are returned.
Next	The same as SQLFetch().
Prior	Return the rowset that is before current rowset.

The following results are expected if the current cursor points to row *k* and *k*-*a* > 0, *k* + *m* + *a* < *N*, *a*>=0:

Result	Interpretation
Relative -*a*	The rows (*k*-*a*, *k*-*a* + *m* -1) are returned
Relative *a*	The rows (*k* + *a*, *k*+*a* + *m* -1) are returned

Implicit setting of scrolling cursor attributes

Certain attributes are set implicitly when your application sets specific attributes. The supported ODBC scrollable cursor attributes set implicitly are as follows:

Application sets attribute to	Other attributes set implicitly
SQL_ATTR_CONCURRENCY to SQL_CONCUR_READ_ONLY	SQL_ATTR_CURSOR_SENSITIVITY to SQL_INSENSITIVE
SQL_ATTR_CONCURRENCY to SQL_CONCUR_LOCK	SQL_ATTR_CURSOR_SENSITIVITY to SQL_SENSITIVE
SQL_ATTR_CURSOR_SCROLLABLE to SQL_NONSCROLLABLE	SQL_ATTR_CURSOR_TYPE to SQL_CURSOR_FORWARD_ONLY
SQL_ATTR_CURSOR_SENSITIVITY to SQL_INSENSITIVE	SQL_ATTR_CONCURRENCY to SQL_CONCUR_READ_ONLY SQL_ATTR_CURSOR_TYPE to SQL_CURSOR_STATIC
SQL_ATTR_CURSOR_TYPE to SQL_CURSOR_FORWARD_ONLY	SQL_ATTR_CURSOR_SCROLLABLE to SQL_NONSCROLLABLE
SQL_ATTR_CURSOR_TYPE to SQL_CURSOR_STATIC	SQL_ATTR_CURSOR_SCROLLABLE to SQL_SCROLLABLE

For the ASE OLE DB Provider

The following describes scrollable cursors for the ASE OLE DB Provider.

Using scrollable cursors

The OLE DB Data Provider supports the following scrolling types:

- Next – return the next row.
- Prior – return the prior row.
- Relative *n* rows – return the row, *n* rows from the current rowset.

Setting scrollable cursor attributes

You must set the following attributes to use scrollable cursors:

- DBPROP_CANSCROLLBACKWARDS – if set to VARIANT_TRUE, the rowset allows the lRowsOffset parameter of GetNextRows to be negative.
- DBPROP_CANFETCHBACKWARDS – if set to VARIANT_TRUE, the rowset will allow the cRows parameter of GetNextRows to be negative.

Executing scrollable cursors

❖ **To set up a program to execute a scrollable cursor**

1 Set the scrollable cursor properties on the rowset:

```
DBPROP RowsetProperties[2];
for(int i = 0; i < 2; i++)

        VariantInit(&RowsetProperties[i].vValue);


RowsetProperties[0].dwPropertyID = DBPROP_CANFETCHBACKWARDS;
RowsetProperties[0].vValue.vt     = VT_BOOL;
RowsetProperties[0].vValue.boolVal= VARIANT_TRUE;
RowsetProperties[0].dwOptions = DBPROPOPTIONS_REQUIRED;
RowsetProperties[0].colid         = DB_NULLID;
RowsetProperties[1].dwPropertyID  = DBPROP_CANSCROLLBACKWARDS;
RowsetProperties[1].vValue.vt     = VT_BOOL;
RowsetProperties[1].vValue.boolVal= VARIANT_TRUE;
RowsetProperties[1].dwOptions     = DBPROPOPTIONS_REQUIRED;
RowsetProperties[1].colid         = DB_NULLID;

DBPROPSET rgRowsetPropSet[1];
rgRowsetPropSet[0].guidPropertySet = DBPROPSET_ROWSET;
rgRowsetPropSet[0].cProperties    = 2;
rgRowsetPropSet[0].rgProperties = RowsetProperties;
```

2 Open the rowset:

```
IRowset* pIRowset = ds.OpenRowset("book", 1, rgRowsetPropSet);
```

3 Fetch the rows forward:

```
DBCOUNTITEM cRowsReturned;
HROW hRow[3];
HROW* pRows = hRow;
hr = pIRowset->GetNextRows(NULL, 0, 3, &cRowsReturned, &pRows);
```

4 Release the rows:

```
hr = pIRowset->ReleaseRows(cRowsReturned, pRows, NULL, NULL, NULL);
```

5 Fetch the rows backward:

```
DBCOUNTITEM cRowsReturned;
HROW hRow[3];
HROW* pRows = hRow;
hr = pIRowset->GetNextRows(NULL, 0, -3, &cRowsReturned, &pRows);
```

6 Release the rows:

```
hr = pIRowset->ReleaseRows(cRowsReturned, pRows, NULL, NULL, NULL);
```

7 Release the rowset:

```
pIRowset->Release()
```

Looking at results

To identify the results and the result set interpretation, after you execute a scrollable cursor, refer to the Microsoft MSDN library.

Example of scrollable static insensitive cursor program

For an example of a scrollable, static-insensitive cursor program refer to the previous section.

New feature in ESD #9

The following feature is new for ESD #9:

Using Distributed Transactions

This section describes how you can use the ASE ODBC driver and the ASE ADO.NET Data Provider to participate in two-phase commit transactions.

Using the ASE ODBC driver to participate in Distributed Transactions

This feature is supported only on Microsoft Windows and requires that Microsoft Distributed Transaction Coordinator (MS DTC) be the transaction coordinator managing two-phase commit.

Sybase supports all of the following programming models:

- Applications using MS DTC directly
- Applications using Sybase Enterprise Application Server (EAServer, also known as Jaguar)
- Applications using Microsoft Transaction Server (MTS) or (COM+)

Programming for MS DTC

❖ **To program using Microsoft Distributed Transaction Coordinator (MS DTC)**

1 Connect to MS DTC by using the DtcGetTransactionManager function. For information about MS DTC, see the relevant Microsoft Distributed Transaction Coordinator documentation.

2 Call SQLDriverConnect or SQLConnect once for each Sybase ASE connection you want to establish.

3 Call the ITransactionDispenser::BeginTransaction function to begin an MS DTC transaction and to obtain an OLE Transaction object that represents the transaction.

4 Call SQLSetConnectAttr one or more times for each ODBC connection you want to enlist in the MS DTC transaction. SQLSetConnectAttr must be called with an attribute of SQL_ATTR_ENLIST_IN_DTC and a ValuePtr of the Transaction object (obtained in step 3).

5 Call SQLExecDirect one or more times for each insert or update SQL statement.

6 Call the ITransaction::Commit function to commit the MS DTC transaction. The Transaction object is no longer valid.

To perform a series of MS DTC transactions, repeat steps 3 through 6.

To release the reference to the Transaction object, call the ITransaction::Release function.

To use an ODBC connection with an MS DTC transaction and then use the same connection with a local ASE Server transaction, call SQLSetConnectAttr with a ValuePtr of SQL_DTC_DONE to unenlist the connection from the transaction.

Note Also, you can call SQLSetConnectAttr and SQLExecDirect separately for each ASE Server instead of calling them as suggested in steps 4 and 5.

Programming components deployed in Sybase EAServer, MTS, or COM+

The following procedure describes how to create components that participate in Distributed Transactions in Sybase EAServer, MTS, or COM+.

❖ **To program components deployed in Sybase EAServer, MTS, or COM+**

1. Call SQLDriverConnect once for each Sybase ASE connection you want to establish.
2. Call SQLExecDirect one or more times for each insert or update SQL statement.
3. Deploy your component to MTS and configure the transaction attributes as needed.

The transaction coordinator creates a distributed transaction as needed, and the component that uses the ASE ODBC driver automatically enlists in the global transaction. Then the transaction coordinator will commit or roll back the distributed transaction.

Connection properties for Distributed Transaction support

The following describes the Connection properties:

- Distributed Transaction Protocol (DistributedTransactionProtocol) – To specify the protocol used to support the distributed transaction, XA Interface standard or MS DTC OLE Native protocol, select the Distributed Transaction Protocol in the ODBC Data Source Dialog or pass the property DistributedTransactionProtocol = *OLE* native protocol in the connection string. The default is *XA*.
- Tightly Coupled Transaction (TightlyCoupledTransaction) – When you have a distributed transaction using two resource managers that point to the same ASE server, you have a situation called a "Tightly Coupled Transaction." Under these conditions, if you do not set this property to *1*, the Distributed Transaction will fail.

  To summarize, if you open two database connections to the same ASE server and then enlist these connections in the same distributed transaction, you must set TightlyCoupledTransaction=*1*.

  To set this property select the Tightly Coupled Transaction in the ODBC Data Source dialog, or pass the property TightlyCoupledTransaction=*1* in the connection string.

Warning! Enlistment with SQLSetConnectAttr returns a SQL_ERROR if the connection has already begun a local transaction either by using SQLSetConnectAttr with the SQL_AUTOCOMMIT_OFF or by executing the BEGIN TRANSACTION statement explicitly using SQLExecDirect.

Using the ASE ADO.NET Data Provider to participate in Distributed Transactions

This feature requires the use of .NET Enterprise Services, which manages the distributed transactions.

Programming using Enterprise Services

The COM+ services infrastructure can be accessed from managed and unmanaged code. Services in unmanaged code are known as COM+ services. In .NET, these services are referred to as Enterprise Services. Working with transactions in Enterprise Services using ADO.NET is very straightforward.

❖ **To program using Enterprise Services**

1 Derive the components from *System.EntrepriseService.ServicedComponent*.

2 Specify the custom attributes (such as Transaction, AutoComplete, and others) to specify the requested services and their options. For a complete list of the attributes, refer to the Enterprise Services documentation.

Note The Timeout Option in the .NET Transaction attribute has to be explicitly set to *-1* or a very high number. .NET documentation states that the ADO.NET transaction timeout default is *0*, which means it will never time out. However, this actually causes an immediate transaction timeout, which rolls back the entire transaction.

3 Sign and build the assembly.

4 Register the assembly.

Connection properties for Distributed Transaction support

The following are the connection properties used in conjunction with Distributed Transaction support.

- Distributed Transaction Protocol (DistributedTransactionProtocol) – To specify the protocol used to support the distributed transaction, XA Interface standard, or MS DTC OLE Native protocol, by setting up the property DistributedTransactionProtocol=*OLE* native protocol in the connection string. The default protocol is *XA*.
- Tightly Coupled Transaction (TightlyCoupledTransaction) – When you have a distributed transaction using two resource managers that point to the same ASE server, you have a situation called a "Tightly Coupled Transaction." Under these conditions, if you do not set this property to *1*, the Distributed Transaction will fail.

To summarize, if you open two database connections to the same ASE server and enlist these connections in the same distributed transaction, you must set TightlyCoupledTransaction=*1*.

- Enlist – The AseConnection object automatically enlists in an existing distributed transaction if it determines that a transaction is active. Automatic transaction enlistment occurs when the connection is opened or retrieved from the connection pool. You can disable this auto-enlistment by specifying Enlist=*0* as a connection string parameter for an AseConnection.

  If auto-enlistment is disabled, you can enlist in an existing distributed transaction by calling the EnlistDistributedTransaction method on the AseConnection with a passed-in ITransaction parameter that is a reference to an existing transaction. After calling the EnlistDistributedTransaction, all updates made using this instance of AseConnection will be made as part of this global transaction. As a result, it will be committed or rolled back when the global transaction is committed or rolled back.

  Note The AseConnection object must be open before calling EnlistDistributedTransaction.

  You can use EnlistDistributedTransaction when you pool business objects. If a business object is pooled with an open connection, automatic transaction enlistment occurs only when that connection is opened or pulled from the connection pool. If multiple transactions are performed using the pooled business object, the open connection for that object will *not* automatically enlist newly initiated transactions. In this instance, you can disable automatic transaction enlistment for the AseConnection and then enlist the AseConnection in transactions using EnlistDistributedTransaction.

  Warning! EnlistDistributedTransaction returns an exception if the AseConnection has already begun a transaction either by using BeginTransaction or by executing the BEGIN TRANSACTION statement explicitly with an AseCommand.

New features in ESD #8

The following features are new for ESD #8:

Extended support for MIT Kerberos

MIT Kerberos 5 version 1.3.6, is now supported on the following platforms:

- IBM AIX 64-bit
- HP-UX 11.11 (or HP-UX 11iv1.0) 64-bit

Sybase currently supports these versions of MIT Kerberos, on the following platforms:

- Version 1.3.1 on Linux Intel 32-bit and Sun Solaris 8 (SPARC 32-bit and 64-bit)
- Version 1.2.7 on Linux AMD64 (Opteron)/EM64T

Support for new platforms and operating systems

Table 12 lists the new platforms, operating systems, and features supported for Sybase SDK 12.5.1 and Open Server 12.5.1 products.

Table 12: SDK and Open Server 12.5.1 new platforms and supported features

Platform	Operating system level	C and C++ compilers	COBOL compilers	Kerberos version	Lightweight Directory Access (LDAP)	Secure Sockets Layer (SSL)
Linux on POWER 32-bit	Red Hat Enterprise Linux AS 3.0	IBM XL C/C++ Advance Edition V7.0	Not available	Not available	Not available	Not available
Linux on POWER 64-bit	Red Hat Enterprise Linux AS 3.0	IBM XL C/C++ Advance Edition V7.0	Not available	Not available	Not available	Not available
Linux AMD64 (Opteron) /EM64T	Red Hat Enterprise Linux AS 3.0	GCC 3.2.3 (Red Hat Linux 3.2.3 - 42)	Not available	MIT 1.2.7	Not available	Not available
Sun x86 32-bit	Solaris 9	Sun C/C++ 6.2	Not available	Not available	Not available	Not available

LDAP functionality on IBM AIX 64-bit

LDAP (OpenLDAP) is now available on IBM AIX 64-bit. You can use the LDAP directory service to create, modify, and retrieve information from network entities.

To enable OpenLDAP, modify the configuration file, *libtcl64.cfg*, available in the *$SYBASE/$SYBASE_OCS/config* directory, then add an LDAP server to the configured directory service. This is described as follows:

❖ **To enable the OpenLDAP directory service**

1 Add your platform’s environment variable to the LDAP library. For example:

```
setenv LIBPATH | \
${LIBPATH} : $SYBASE/$SYBASE_OCS/lib3p64
```

2 Select an LDAP server in the [DIRECTORY] section of *libtcl64.cfg*, then add a new keyword value pair:

Note The LDAP URL must be on a single line.

```
ldap=libdldap64.so
ldap://host.port/ditbase??scope??
bindname=username.password
```

For example:

```
[DIRECTORY]
ldap=libdldap64.so ldap://huey:11389/dc=sybase,
dc=com??one??bindname=cn=Manager, dc=sybase,
dc=com?secret
```

❖ **To add an LDAP server to the configured OpenLDAP directory service**

1 Launch the dsedit or the dscp utility in the *$SYBASE/$SYBASE_OCS/bin* directory.

2 Provide server details such as server name, and network transport details such as transport type, host name, and port number.

CyberSafe Kerberos Client upgrade on Windows

Sybase now supports CyberSafe ActiveTRUST Secure Client (Kerberos) 4.0 on Microsoft Windows (Windows NT, Windows 2000, Windows 2003, Windows XP).

CyberSafe Kerberos is supported for the following products:

- Software Developer’s Kit components:
    - Open Client (Client-Library)
    - ESQL/C
    - ESQL/COBOL
- Open Server

SSLPlus 3.1.x security fixes

The following versions of Secure Sockets Layer (SSL) fixed some SSLPlus bugs and memory leaks:

- SSL Plus 3.1.14c on Sun Solaris 8 (SPARC 32-bit and 64-bit), HP-UX 11.11 (32-bit and 64-bit), HP Tru64, IBM AIX (32-bit and 64-bit), and Linux Intel 32-bit. This version is available with ESD #7 and later.
- SSL Plus 3.1.15b on Microsoft Windows (Windows NT 32-bit, Windows 2000, Windows 2003, Windows XP). This version is available with ESD #8 and later.

The above versions of SSLPlus 3.1.x are linked with the Sybase SSL filter, *libflssl.**.

COBOL compiler upgrade on Windows 2000

The COBOL compiler is upgraded to Micro Focus Net Express 4.0 on Microsoft Windows 2000.

New feature in ESD #7

The following feature is new for ESD #7:

XA 64-bit support extended to additional platforms

Extended Architecture (XA) 64-bit, which is already available on Solaris 64-bit, has been extended to the following platforms:

- IBM AIX 64-bit
- HP-UX 11.11 (or HP-UX 11iv1.0) 64-bit
- HP Itanium 64-bit

Sybase currently supports XA 32-bit on the following 32-bit platforms:

- IBM AIX 32-bit
- HP-UX 11.11 (or HP-UX 11iv1.0) 32-bit
- Sun Solaris 8 (SPARC 32-bit)
- Microsoft Windows (Windows NT, Windows 2000, Windows 2003, Windows XP)

Overview of XA Interface

The Distributed Transaction Manager (DTM) XA Interface is the Sybase implementation of the XA Interface standard for Adaptive Server. The DTM XA Interface is one of the elements of the X/Open Distributed Transaction Processing (DTP) model, which provides an industry standard for developing DTP applications. The XA Interface accesses data that is stored on Adaptive Servers from a CICS, Encina, or TUXEDO Transaction Manager (TM).

Components of the XA Interface

The Sybase XA Interface consists of the following:

- The Sybase DTM XA Interface, which is the Sybase implementation of the XA Interface for Adaptive Server. This is a separately-licensed Adaptive Server feature.
- Adaptive Server and the DTM feature. Software installation and feature licenses are described in the Adaptive Server Enterprise *Installation Guide* for your platform.

- Sybase Open Client, which allows Client-Library calls to be part of the native interface between your application and the resource manager.
- The XA configuration file, which contains entries that define client/server connections for use with XA.
- Embedded SQL/C and Embedded SQL/COBOL, which create ESQL calls as part of the native interface between your application and the resource manager.
- A set of XA-specific dbcc commands that allow System Administrators to manage heuristic transactions.
- TM-specific configuration files and commands you can use for global recovery.

For information on how to use native Adaptive Server DTM features, see the *XA Interface Integration Guide* for CICS, Encina, and TUXEDO.

New feature in ESD #6

The following feature is new for ESD #6:

New ASE OLE DB Provider by Sybase

This release of SDK 12.5.1 introduces a new OLE DB Provider for Sybase ASE named “Adaptive Server Enterprise OLE DB Provider by Sybase”. OLE DB is a data access model from Microsoft. It uses the Component Object Model (COM) interface and, unlike ODBC, does not assume that the data source uses a SQL query processor.

Supported platforms

The ASE OLE DB Provider is designed to work with OLE DB 2.5 and later. Supported platforms include Windows NT 4.0, 2000, XP, and 2003.

Supported ASE features

The Adaptive Server Enterprise OLE DB Provider supports the following ASE specific features:

- SSL
- Directory services (LDAP)
- Password encryption

Preparing for migration

This ASE OLE DB Provider, developed by Sybase, will replace the existing OLE DB Provider in a future release. Currently, both of these providers are shipped and can co-exist on the same machine. You are encouraged to begin migrating and testing your applications with the new provider to identify any compatibility issues. The existing OLE DB Provider (provider short name "Sybase.ASEOLEDBProvider") is installed in *%SYBASE%\OLEDB* directory and currently at version 2.70. The new ASE OLE DB Provider (provider short name "ASEOLEDB") is installed in *%SYBASE%\DataAccess\OLEDB* directory and at version 12.5.1.384 or later.

For additional information, see the *User's Guide* for Adaptive Server Enterprise OLE DB Provider by Sybase.

New features in ESD #5

The following features are new for ESD #5:

BCP encrypted columns

Currently, data is retrieved in plain text when BCP is running against a table that supports encrypted columns, and the user has permission to view the data in the columns.

The new BCP command line option (-C) allows bulk movement of cipher-text data for authorized users. For this to occur, you must use the -C option, and the table located on the server must support encrypted columns. This results in a SQL command, set ciphertext=*on*, to be sent to the server before initiating any bulk library routines to produce cipher-text.

Note If the new option -C is present and the server does not support encrypted columns, no error will be generated. Instead, BCP will send the SQL command based on the presence of a table, sysencryptkeys. If a SQL command fails when this table is present, it is considered a fatal error.

Korean character support enhancements

Open Client and Open Server now supports the Korean character set cp949. Directories were added to the *charsets* directory, the table files for these charsets were added to the *utf8 charset* directory, the *utf8.cfg* file, the *locales.dat*, and the *objectid.dat* was modified to add these charsets. This Korean character set may be used in the same manner as any other charset, including having character data converted through cs_convert().

SSLPlus 3.1.10H security fixes

Secure Sockets Layer (SSL) SSLPlus 3.1.10H, the latest release of version 3.1x, is available with ESD #5 or later. This release fixed some SSLPlus bugs and memory leaks. SSLPlus 3.1.10H is linked with the Sybase SSL filter, *libflssl.**, on Sun Solaris 8 (SPARC 32-bit and 64-bit), Linux Intel 32-bit, IBM AIX 64-bit, and Microsoft Windows NT.

Support for MIT Kerberos on ODBC and OLE DB

Starting with ODBC 5.00.0034 and OLE DB 2.70.0041, the Sybase ODBC Driver and the OLE DB Provider now support MIT Kerberos, in addition to CyberSafe Kerberos. Refer to the help files released together with the products for detail information about how to use this feature.

❖ **To locate the Kerberos setup information for the Sybase ASE ODBC Driver and Sybase ASE OLE DB Provider**

1. Start the ODBC Data Source Administrator tool (*ODBCAD32.EXE*).
2. Select your Data Source and click Configure. (If you only use OLE DB Provider, click Add and then, select the Sybase ASE ODBC Driver.)
3. Go to the Connection Panel and click Help.
4. Read the "Use Kerberos" section. This provides the resource for either CyberSafe Kerberos or MIT Kerberos.

Installing SDK 12.5.1 on Windows

If you upgrade and install SDK 12.5.1 into an older SDK 12.5 Sybase directory where the 12.5 ODBC driver was installed (*%SYBASE%\odbc*), the installer updates the ODBC driver and you have immediate access to it.

Note Sybase recommends using your original ODBC driver directory location.

If you choose to install 12.5.1 in a new directory on a machine that contains the older SDK 12.5 ODBC driver (in the old Sybase directory for ODBC), you will not have access to the new ODBC driver. Data sources used during installation store a hard copy of the absolute path to where the *SYODASE.DLL* exists. There is no utility that updates a Windows registry for the ODBC driver DSNs to update the reference to the *SYODASE.DLL*. If you change the ODBC driver location, you need to either copy the *SYODASE.DLL* and its related *DLLs* and components to the older Sybase directory, or change all your DSNs to use the new driver location.

Extended platform support for the ASE ODBC Driver by Sybase

The ODBC interface is a call-based application programming interface defined by Microsoft Corporation as a standard interface for database management systems. The Adaptive Server Enterprise ODBC 12.5.1 driver allows you to write programs that access data in Adaptive Server using the ODBC call interface. In addition to the ODBC Level 2 features, the driver supports the following features:

- SSL
- HA Failover
- Password encryption

This SDK release includes additional platform support for the ASE ODBC Driver by Sybase. The driver was introduced in SDK 12.5.1 ESD #2 for the Linux platform and this release adds support for Windows and Mac OS X operating systems.

Note This driver, developed by Sybase, will replace the existing ODBC Driver Kit in a future release. Currently, both of these drivers are shipped in parallel and can co-exist on the same machine. You are encouraged to begin migrating and testing your applications with the new driver to identify any compatibility issues. The existing ODBC Driver Kit is registered with the ODBC Driver Manager as “Sybase ASE ODBC Driver” and is currently at version 5.00.0034. The ASE ODBC Driver by Sybase is registered as “Adaptive Server Enterprise” and currently at version 12.5.1.376.

New features in ESD #4

The following features are new for ESD #4:

Diagnostic enhancements

The Client-Library ct_debug routine has been revised and allows you to trace and dump the Tabular Data Stream™ (TDS) to a file, without being required to download, install and configure capture, or use the Ribo (TDS trace tool). The following are the revisions to the ct_debug routine parameters:

- To enable CS_DBG_PROTOCOL, or to set CS_DBG_PROTOCOL_FILE, an application no longer needs to use the *devlib* (debug) libraries.
- If you do not set CS_DBG_PROTOCOL_FILE on a connection, mktemp is called to generate a unique file name to contain the dump of the protocol packets. The prefix string passed to mktemp is “captur” and the resulting protocol file can be decoded by Ribo.
- It is possible for you to enable the protocol dump functionality using the *ocs.cfg* file which will cause both the normal and *devlib* libraries to create protocol dump files.

Asynchronous communications support

This is new functionality for the Sybase ASE ODBC Driver 5.00.0022 and the Sybase ASE OLE DB Provider 2.70.0038. The products now support asynchronous communications as specified in the ODBC 3.5 specification.

Connection level asynchronous support

If the TDS ODBC driver's Cursor Mode (Select Method) is set to Cursor, the driver indicates through SQLGetInfo that it supports connection level asynchronous support. Calling SQLGetInfo with the information type of SQL_ASYNC_MODE returns an SQL_AM_CONNECTION.

Statement level asynchronous support

If the TDS ODBC driver's Cursor Mode is set to Direct, the driver indicates through SQLGetInfo that it supports statement level asynchronous support. Calling SQLGetInfo with the information type of SQL_ASYNC_MODE returns an SQL_AM_STATEMENT.

Maximum asynchronous concurrent statements

The TDS ODBC driver indicates through SQLGetInfo a specific limit of one (1) to the number of concurrent asynchronous statement. Calling SQLGetInfo with the information type of SQL_MAX_ASYNC_CONCURRENT_STATEMENTS returns one (1).

Enabling asynchronous communications support

If the TDS ODBC driver's cursor mode is set to Cursor, the TDS driver enables asynchronous communications support by calling the SQLSetConnectAttr. Calling SQLSetConnectAttr with the attribute value SQL_ATTR_ASYNC_ENABLE set to SQL_ASYNC_ENABLE_ON turns on asynchronous support. Making a SQLSetConnectAttr call with the attribute value SQL_ATTR_ASYNC_ENABLE set to SQL_ASYNC_ENABLE_OFF turns off asynchronous support.

If the TDS driver's cursor mode is set to Direct, the TDS driver will enable asynchronous communications support by calling the SQLSetStmtAttr. Calling SQLSetStmtAttr with the attribute value SQL_ATTR_ASYNC_ENABLE set to SQL_ASYNC_ENABLE_ON will turn on asynchronous support. Making a SQLSetStmtAttr call with the attribute value SQL_ATTR_ASYNC_ENABLE set to SQL_ASYNC_ENABLE_OFF will turn off asynchronous support.

Asynchronous ODBC function support

The following ODBC functions can be supported asynchronously:

SQLBulkOperations

SQLColAttribute

SQLColumnPrivileges

SQLColumns

SQLCopyDesc

SQLDescribeCol

SQLDescribeParam

SQLExecDirect

SQLExecute

SQLFetch

SQLFetchScroll

SQLForeignKeys

SQLGetData

SQLGetDescField

SQLGetDescRec

SQLGetDiagField

SQLGetDiagRec

SQLGetTypeInfo

SQLMoreResults

SQLNumParams

SQLNumResultCols

SQLParamData

SQLPrepare

SQLPrimaryKeys

SQLProcedureColumns

SQLProcedures

SQLPutData

SQLSetPos

SQLSpecialColumns

SQLStatistics

SQLTablePrivileges

SQLTables

While any of these functions are executing asynchronously they will return a SQL_STILL_EXECUTING return code. Once the given function ends, it will return the code it would have returned had it been executed asynchronously, such as SQL_SUCCESS, SQL_ERROR, or SQL_NEED_DATA.

Processing changes

Once asynchronous communications has been enabled, the TDS driver will execute the functions specified above asynchronously as detailed in the ODBC 3.5 specification. Once asynchronous communications is disabled, all functions will execute asynchronously. All other processing will operate as normally.

DB-Library new configuration variable

This new feature uses an environment variable to externally configure the DB-Library version level at runtime to change the application code by calling dbsetversion. It provides the following:

- Retrieves the environment variable at the DB-Library initialization stage
- Stores the environment variable value as the version level

Currently, in the source code, DB-Library sets the default version level to DBVERSION_46. The only way you can change this default version level is to call dbsetversion in an application. If a user wishes to use features provided by TDS protocol version 5.0 (for example, numeric and decimal types), the hard coded default version level (DBVERSION_46) needs to be changed. In the past, users had to add the call dbsetversion(DBVERSION_100) to their code. This feature allows you to decide the DB-Library version level without calling dbsetversion from an application.

Definitions

The following are definitions used in describing the new DB-Library configuration variable:

- DB-Library — a collection of C programming API routines and macros that allow an application to interact with data server and Open Server applications.
- CT-Library — a collection of C programming API routines used in writing client applications.
- Version level — a parameter in DB-Library that is used to specify the version of the TDS protocol. There are three valid values:
  - DBVERSION_46 — DB-Library is running with TDS version 4.6 protocol.

    - DBVERSION_100 — DB-Library is running with TDS version 5.0 protocol
    - DBVERSION_UNKNOWN —the version level has not been decided yet.
- dbsetversion — a DB-Library API that sets the version level of the DB-Library in the client's application.
- dbinit — an API that initializes the DB-Library. It must be called before calling any other DB-Library routines.

Configuration

To implement this feature, you must define a new environment variable SYBOCS_DBVERSION to allow the DB-Library version level to be configured externally.

The following are valid values for this environment variable:

- DBVERSION_46
- DBVERSION_100

Any other value causes the DB-Library application to fail.

Processing order

The following paragraphs describe the order for processing the environment variable and the dbsetversion in a DB-Library application:

- If an environment variable is not defined and the dbsetversion is not called in an application, the version level is DBVERSION_46, which is currently the default version in the DB_Library source code.
- If an environment variable is not defined and the dbsetversion is called with a value of DBVERSION_100 or DBVERSION_46, the version level is DBVERSION_100 or DBVERSION_46, respectively.
- If an environment variable is defined as DBVERSION_100 or DBVERSION_46 and the dbsetversion is not called in an application, the version level is DBVERSION_100 or DBVERSION_46, respectively.
- If an environment variable is defined as DBVERSION_100 and dbsetversion is called to set version level as DBVERSION_46 in the application, the version level is DBVERSION_46, since the dbsetversion overrides the environment variable and decides the final version level.

Adaptive Server Enterprise ODBC driver for Linux Intel 32-bit

This SDK release includes the Open Database Connectivity (ODBC) driver. The ODBC interface is a call-based application programming interface defined by Microsoft Corporation as a standard interface for database-management systems. The Adaptive Server Enterprise ODBC 12.5.1 driver allows you to write programs that access data in Adaptive Server using the ODBC call interface. In addition to the ODBC Level 2 features, the driver supports the following features:

- SSL
- HA Failover
- Password encryption

New ASE ODBC driver

Note Release of the ASE ODBC 5.0 driver is based on the ODBC 3.52 specification. Any application written to use the ASE ODBC 4.x driver will work without any changes for the 5.0 release.

The following is a summary of the new functionality that the ASE ODBC driver 5.0 provides:

- Support for date and time datatypes
- Support for password encryption
- Removal of the BAS\UTL\FLT shared components

New feature in ESD #3

There following feature is new for ESD #3:

Adaptive Server Enterprise ADO.NET Data Provider 1.1 on Microsoft Windows

This release includes the ASE ADO.NET Data Provider 1.1, which is an ADO.NET provider for the Adaptive Server database. It allows you to access data in Adaptive Server using any language supported by .NET, including C#, Visual Basic .NET, C++ with managed extensions, and J#. To access the new features, you must upgrade to version 1.1 as described in the following section.

Upgrading to ASE ADO.NET Data Provider 1.1

The Common Language Runtime (CLR) bytecode for applications built against previous versions of ASE ADO.NET Data Provider is not compatible with ASE ADO.NET Data Provider 1.1. To use the updated version of ASE ADO.NET Data Provider, you must rebuild your applications. Multiple versions of ASE ADO.NET Data Provider can coexist on the client machine; therefore, your old application can continue to use the previous version of ASE ADO.NET Data Provider. See the Adaptive Server Enterprise ADO.NET Data Provider *User’s Guide* for more information.

New features in ASE ADO.NET Data Provider 1.1

New features in ASE ADO.NET Data Provider 1.1 include:

- Secure Sockets Layer (SSL)
- Directory services – Lightweight Directory Access Protocol (LDAP)
- High Availability (HA) Failover
- Password encryption
- ASECommandBuilder
- ASECommandExecutXmlReader

For additional information for these new features, refer to the *User’s Guide* Adaptive Server Enterprise ADO.NET Data Provider.

New features in ESD #2

The following features are new for ESD #2:

SSL Plus 3.1.5 + SBGSE support on Solaris and Microsoft Windows

Sybase now supports Secure Sockets Layer (SSL) Plus 3.1.5 + SBGSE on Solaris 32-bit and 64-bit and Microsoft Windows on the following products:

- Software Developer's Kit components:
    - Client-Library
    - ESQL/C
    - ESQL/COBOL
- Open Server

As a result of Federal regulations, Sybase must use the FIPS-certified cipher suites. After you have upgraded to SDK 12.5.1 ESD #2 (which includes Open Client) and to Open Server 12.5.1 ESD #2 or later, for Microsoft Windows, you must initialize the Certicom libraries as described in the following procedure before you can do any SSL testing. For Solaris, you must make sure that the *lib3p* directory is in the dynamic load library path.

❖ **Initializing the Certicom libraries for Windows**

1. To allow the *ctlib* and *srvlib* libraries to find the Certicom cipher suites in *libsb.dll*, add the following to the dynamic load library path:

   ```
   %SYBASE%\%SYBASE_OCS%\lib3p
   ```

2. Run the following program:

   ```
   %SYBASE%\%SYBASE_OCS%\bin\setsslreg.exe
   ```

   This program adds an item to the system registry that is used by *libsb.dll* for a self-test. Once the value is in the registry for a particular machine, you do not need to rerun the setsslreg.exe program unless the value is deleted for some reason.

3. Run the following program:

   ```
   %SYBASE%\%SYBASE_OCS%\bin\sbgtest.exe
   ```

This program verifies that the registry entry and *libsb.dll* are properly configured (be sure that *lib3p* is in the process path). The *sbgtest.exe* is used to confirm that *libsb.dll* can get to the registry entry and passes its self-test.

The situations when you might use *sbgtest.exe* more than once are as follows:

- After running setsslreg, to confirm that the registry entry exists.
- After experiencing a complete failure using SSL, to verify that *libsb.dll* is still functioning.

Note It is very important that when you run an application that uses SSL, that the *lib3p* directory is in its dynamic load library path.

IPV6 support on Solaris and Microsoft Windows XP and 2003

Sybase now supports IPV6 on Sun Solaris 8 (SPARC 32-bit and 64-bit), and Microsoft Windows XP and Windows 2003, on the following products:

- Software Developer's Kit components:
  - Client-Library
  - ESQL/C
  - ESQL/COBOL
- Open Server

Open Client and Open Server each use getaddrinfo() to translate the host name to addresses as follows:

- Open Client attempts to establish a connection to a server by using each of the addresses returned by getaddrinfo() until a connect() succeeds.
- Open Server attempts to establish a connection to a server, depending on whether the IPV6 addresses are returned by getaddrinfo(), and behaves as follows:
  - If IPV6 addresses are *not* returned by getaddrinfo(), Open Server establishes a listener for each address that getaddrinfo() returns.

- If IPV6 addresses are returned by getaddrinfo(), Open Server creates a wildcard for the address of the listening port. If Open Server listens only on a subset of the valid IP and IPV6 addresses on a host, the actual addresses to be used must be entered as separate lines in the *interfaces* file.

Note As a result of the definition in the IPV6 specification, Open Server establishes a listener for both the IPV6 local host and the IP local host if it determines that getaddrinfo() returned the loop-back address.

DB-Library LDAP functionality on Microsoft Windows

This feature allows DB-Library (*dblib*) applications calling dbopen to connect to an Adaptive Server or an Open Server using LDAP directory services. To do so, you must modify the *libctl.cfg* file in the [DIRECTORY] section, as described in the Open Client and Open Server *Configuration Guide* for Microsoft Windows. There are no application changes required to use this feature.

The ability to access LDAP already exists in Open Client and Open Server products, and the generic directory service access code exists in *netlib*. Therefore, applications only need to link to *netlib*.

Note Because Sybase is using *netlib* for I/O, *netlib* error messages may be generated instead of the current DB-Library generated messages.

New feature in ESD #1

The following feature is new for ESD #1:

XA support on Sun Solaris 8 (SPARC 64-bit)

XA 64-bit is now available on Sun Solaris 8 (SPARC 64-bit). Although support for XA 64-bit is not available on any other 64-bit platform, Sybase currently supports XA 32-bit on the following 32-bit platforms:

- IBM AIX 32-bit
- HP-UX (or HP-UX 11iv1.0) 32-bit
- Sun Solaris 8 (SPARC 32-bit)
- Microsoft Windows

Overview of XA Interface

The DTM XA Interface is the Sybase implementation of the XA Interface standard for Adaptive Server. The DTM XA Interface is one of the elements of the X/Open Distributed Transaction Processing (DTP) model, which provides an industry standard for developing DTP applications. The XA Interface accesses data that is stored on Adaptive Servers from a CICS, Encina, or TUXEDO Transaction Manager (TM).

Components of the XA Interface

The Sybase XA Interface consists of the following:

- The Sybase DTM XA Interface, which is the Sybase implementation of the XA Interface for Adaptive Server. This is a separately-licensed Adaptive Server feature.
- Adaptive Server, and the DTM feature. Software installation and feature licenses are described in the Adaptive Server Enterprise *Installation Guide* for your platform.
- Sybase Open Client, which allows Client-Library calls to be part of the native interface between your application and the resource manager.
- The XA configuration file, which contains entries that define client/server connections for use with XA.
- Embedded SQL/C and Embedded SQL/COBOL, which create ESQL calls as part of the native interface between your application and the resource manager.
- A set of XA-specific dbcc commands that allow System Administrators to manage heuristic transactions.
- TM-specific configuration files and commands you can use for global recovery.

For information on how to use native Adaptive Server DTM features, see the *XA Interface Integration Guide* for CICS, Encina, and TUXEDO.