SYBASE®

Large Objects Management

Sybase IQ

15.1

DOCUMENT ID: DC00172-01-1510-02

LAST REVISED: December 2009

Sybase, Inc., One Sybase Drive, Dublin, CA 94568.

Contents

About This Book

Subject

Sybase® IQ is a high-performance decision support server designed specifically for data warehouses and data marts. This book, *Large Objects Management in Sybase IQ*, provides reference material for working with Large Object (LOB) data in Sybase IQ. This manual is the place to look for information such as available syntax, parameters, functions, stored procedures, and options related to Sybase IQ LOB data. Read this manual to understand storage and retrieval of Binary Large Objects (BLOBs) and Character Large Objects (CLOBs) within the Sybase IQ data repository.

Audience

This manual is a reference for all users of Sybase IQ.

How to use this book

This book provides descriptions of the Large Objects Management features in Sybase IQ and is designed to be used as a reference together with the other books in the Sybase IQ documentation set.

The following table shows which chapters fit a particular interest or need.

Table 1: Guide to using this book

To learn about...	Refer to...
The Large Objects Management Option	Chapter 1, "Introduction to Large Objects Management in Sybase IQ"
The characteristics of Binary Large Object data	Chapter 2, "Binary Large Object (BLOB) data"
The characteristics of Character Large Object data	Chapter 3, "Character Large Object (CLOB) data"
Functions that support Large Object data	Chapter 4, "Function Support"
Stored procedures that support Large Object data	Chapter 5, "Stored Procedure Support"
Exporting and loading Large Object data	Chapter 6, "Moving Large Object Data"
Compatibility of Sybase IQ Large Object data	Chapter 7, "Compatibility and Conformance"
Errors and warnings specific to Large Object data	Chapter 8, "Error and Warning Messages"

Windows platforms

The Windows information in this book applies to all supported Windows platforms, unless noted otherwise. For supported Windows platforms, see the *Sybase IQ Release Bulletin for Windows*.

Related documents

The Sybase IQ 15.1 documentation set includes:

- *Release Bulletin* provides information about last-minute changes to the product and documentation.
- *Installation and Configuration Guide* provides platform-specific instructions on installing, migrating to a new version, and configuring Sybase IQ for a particular platform.
- *Advanced Security in Sybase IQ* covers the use of user encrypted columns within the Sybase IQ data repository. You need a separate license to install this product option.
- *Error Messages* lists Sybase IQ error messages referenced by Sybase error code, SQLCode, and SQLState, and SQL preprocessor errors and warnings.
- *IMSL Numerical Library User's Guide: Volume 2 of 2 C Stat Library* contains a concise description of the IMSL C Stat Library time series C functions. This book is only available to RAP – The Trading Edition™ Enterprise users.
- *Introduction to Sybase IQ* includes hands-on exercises for those unfamiliar with Sybase IQ or with the Sybase Central™ database management tool.
- *Large Objects Management in Sybase IQ* explains storage and retrieval of Binary Large Objects (BLOBs) and Character Large Objects (CLOBs) within the Sybase IQ data repository. You need a separate license to install this product option.
- *New Features in Sybase IQ 15.0* documents new features and behavior changes for version 15.0.
- *New Features Summary Sybase IQ 15.1* summarizes new features and behavior changes for the current version.
- *Performance and Tuning Guide* describes query optimization, design, and tuning issues for very large databases.
- *Quick Start* lists steps to build and query the demo database provided with Sybase IQ for validating the Sybase IQ software installation. Includes information on converting the demo database to multiplex.
- *Reference Manual* – Includes two reference guides to Sybase IQ:
  - *Reference: Building Blocks, Tables, and Procedures* describes SQL, stored procedures, data types, and system tables that Sybase IQ supports.

  - *Reference: Statements and Options* describes the SQL statements and options that Sybase IQ supports.
- *System Administration Guide* – Includes two volumes:
  - *System Administration Guide: Volume 1* describes startup, connections, database creation, population and indexing, versioning, collations, system backup and recovery, troubleshooting, and database repair.
  - *System Administration Guide: Volume 2* describes writing and running procedures and batches, programming with OLAP, accessing remote data, setting up IQ as an Open Server, scheduling and event handling, programming with XML, and debugging.
- *User-Defined Functions Guide* provides information about the user-defined functions, their parameters, and possible usage scenarios.
- *Using Sybase IQ Multiplex* tells how to use multiplex capability, designed to manage large query loads across multiple nodes.
- *Utility Guide* provides Sybase IQ utility program reference material, such as available syntax, parameters, and options.

Sybase IQ and SQL Anywhere
Because Sybase IQ is an extension of SQL Anywhere Server, a component of the SQL Anywhere® package, Sybase IQ supports many of the same features as SQL Anywhere Server. The IQ documentation set refers you to SQL Anywhere documentation, where appropriate.

Documentation for SQL Anywhere includes:

- *SQL Anywhere Server – Database Administration* describes how to run, manage, and configure SQL Anywhere databases. It describes database connections, the database server, database files, backup procedures, security, high availability, and replication with Replication Server®, as well as administration utilities and options.
- *SQL Anywhere Server – Programming* describes how to build and deploy database applications using the C, C++, Java, PHP, Perl, Python, and .NET programming languages such as Visual Basic and Visual C#. This book also describes a variety of programming interfaces such as ADO.NET and ODBC.

- *SQL Anywhere Server – SQL Reference* provides reference information for system procedures, and the catalog (system tables and views). It also provides an explanation of the SQL Anywhere implementation of the SQL language (search conditions, syntax, data types, and functions).
- *SQL Anywhere Server – SQL Usage* describes how to design and create databases; how to import, export, and modify data; how to retrieve data; and how to build stored procedures and triggers.

You can also refer to the SQL Anywhere documentation in the SQL Anywhere 11.0.1 collection at Product Manuals at http://sybooks.sybase.com and in DocCommentXchange at http://dcx.sybase.com/dcx_home.php.

Documentation for Sybase Software Asset Management (SySAM) includes:

- *Sybase Software Asset Management (SySAM) 2* introduces asset management concepts and provides instructions for establishing and administering SySAM 2 licenses.
- *SySAM 2 Quick Start Guide* tells you how to get your SySAM-enabled Sybase product up and running.
- *FLEXnet Licensing End User Guide* explains FLEXnet Licensing for administrators and end users and describes how to use the tools that are part of the standard FLEXnet Licensing distribution kit from Sybase.

Other sources of information

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

- The Getting Started CD contains release bulletins and installation guides in PDF format, and may also contain other documents or updated information not included on the SyBooks CD. It is included with your software. To read or print documents on the Getting Started CD, you need Adobe Acrobat Reader, which you can download at no charge from the Adobe Web site using a link provided on the CD.
- The SyBooks CD contains product manuals and is included with your software. The Eclipse-based SyBooks browser allows you to access the manuals in an easy-to-use, HTML-based format.

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

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

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

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

Sybase certifications on the Web

Technical documentation at the Sybase Web site is updated frequently.

❖ Finding the latest information on product certifications

1. Point your Web browser to Technical Documents at http://certification.sybase.com/ucr/search.do.
2. Either select the product family and product under Search by Base Product; or select the platform and product under Search by Platform.
3. Select Search to display the availability and certification report for the selection.

❖ Finding the latest information on component certifications

1. Point your Web browser to Availability and Certification Reports at http://certification.sybase.com/.
2. Either select the product family and product under Search by Base Product; or select the platform and product under Search by Platform.
3. Select Search to display the availability and certification report for the selection.

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

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

1. Point your Web browser to Technical Documents at http://www.sybase.com/support/techdocs/.
2. Click MySybase and create a MySybase profile.

Sybase EBFs and software maintenance

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

1. Point your Web browser to the Sybase Support Page at http://www.sybase.com/support.
2. Select EBFs/Maintenance. If prompted, enter your MySybase user name and password.
3. Select a product.
4. Specify a time frame and click Go. A list of EBF/Maintenance releases is displayed.

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

Syntax conventions

This documentation uses these conventions in syntax descriptions:

- **Keywords** SQL keywords are shown in UPPERCASE. However, SQL keywords are case-insensitive, so you can enter keywords in any case; SELECT, Select, and select are equivalent.
- **Placeholders** Items that must be replaced with appropriate identifiers or expressions are shown in *italics*.
- **Continuation** Lines beginning with an ellipsis (...) are a continuation of the statements from the previous line.
- **Repeating items** Lists of repeating items are shown with an element of the list followed by an ellipsis (...). One or more list elements are allowed. If multiple elements are specified, they must be separated by commas.
- **Optional portions** Optional portions of a statement are enclosed by square brackets. For example:

  ```
  RELEASE SAVEPOINT [ savepoint-name ]
  ```

  The square brackets indicate that the *savepoint-name* is optional. Do not type the brackets.

- **Options** When none or only one of a list of items must be chosen, the items are separated by vertical bars and the list enclosed in square brackets. For example:

  ```
  [ ASC | DESC ]
  ```

  The square brackets indicate that you can choose ASC, DESC, or neither. Do not type the brackets.

- **Alternatives** When precisely one of the options must be chosen, the alternatives are enclosed in curly braces. For example:

  ```
  QUOTES { ON | OFF }
  ```

  The curly braces indicate that you must include either ON or OFF. Do not type the brackets.

Typographic conventions

Table 2 lists the typographic conventions used in this documentation.

Table 2: Typographic conventions

Item	Description
`Code`	SQL and program code appears in a monospaced (fixed-width) font.
`User entry`	Text entered by the user is shown in a monospaced (fixed-width) font.
emphasis	Emphasized words are shown in italic.
file names	File names are shown in italic.
database objects	Names of database objects, such as tables and procedures, are shown in bold, sans serif type in print, and in italic online.

The demo database

Sybase IQ includes scripts to create a demo database (*iqdemo.db*). Many of the queries and code samples in this document use the demo database as a data source.

The demo database contains internal information about a small company (employees, departments, and financial data), as well as product (products), and sales information (sales orders, customers, and contacts).

See the Sybase IQ installation guide for your platform or talk to your system administrator for more information about the demo database.

Accessibility features

This document is available in an HTML version that is specialized for accessibility. You can navigate the HTML with an adaptive technology such as a screen reader, or view it with a screen enlarger.

Sybase IQ 15.1 and the HTML documentation have been tested for compliance with U.S. government Section 508 Accessibility requirements. Documents that comply with Section 508 generally also meet non-U.S. accessibility guidelines, such as the World Wide Web Consortium (W3C) guidelines for Web sites.

Configuring your accessibility tool

You might need to configure your accessibility tool for optimal use. Some screen readers pronounce text based on its case; for example, they pronounce ALL UPPERCASE TEXT as initials, and MixedCase Text as words. You might find it helpful to configure your tool to announce syntax conventions. Consult the documentation for your tool for information on using screen readers.

For information about how Sybase supports accessibility, see Sybase Accessibility at http://www.sybase.com/accessibility. The Sybase Accessibility site includes links to information on Section 508 and W3C standards.

For a Section 508 compliance statement for Sybase IQ, go to Sybase Accessibility at http://www.sybase.com/products/accessibility.

If you need help

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

CHAPTER 1 Introduction to Large Objects Management in Sybase IQ

About this chapter

This chapter introduces you to large objects management.

Contents

The Large Objects Management Option

The Large Objects Management Option extends the capabilities of Sybase IQ Enterprise Edition to allow storage and retrieval of Binary Large Objects (BLOBs) and Character Large Objects (CLOBs) within the Sybase IQ data repository.

Users must be specifically licensed to use the Large Objects Management functionality described in this product documentation.

As data volumes continue to increase, the need to store Large Object (LOB) data in a relational database also increases. LOB data may be either:

- unstructured, in which case the database simply stores and retrieves the data
- structured (for example, text) in which case the database understands the data structure and provides supporting functions (for example, string functions)

Typical LOB data sources are images, maps, documents (for example, PDF), audio, video, and XML. Sizes of an individual LOB may extend into gigabytes (GB), terabytes (TB), or even petabytes (PB).

CHAPTER 2 Binary Large Object (BLOB) data

About this chapter

This chapter describes the characteristics of the LONG BINARY data type column, which stores Sybase IQ Binary Large Object data.

Contents

Large Object data types LONG BINARY and BLOB

Binary Large Object (BLOB) data in Sybase IQ is stored in columns of data type LONG BINARY or BLOB.

An individual LONG BINARY data value can have a length ranging from zero (0) to 512TB (terabytes) for an IQ page size of 128KB or 2PB (petabytes) for an IQ page size of 512KB. (The maximum length is equal to 4GB multiplied by the database page size.) The IQ database must be created with an IQ page size of at least 128KB (131072 bytes) in order to accommodate a table with LONG BINARY data.

A table or database can contain any number of LONG BINARY columns up to the supported maximum columns per table and maximum columns per database, respectively.

LONG BINARY columns can be either NULL or NOT NULL and can store zero-length values. The domain BLOB is a LONG BINARY data type that allows NULL.

A non-FP index or join index cannot be constructed on a LONG BINARY column.

A LONG BINARY column can be modified using the UPDATE, INSERT, LOAD TABLE, DELETE, TRUNCATE, SELECT...INTO and INSERT...LOCATION SQL statements. Positioned updates and deletes are not supported on LONG BINARY columns.

An Adaptive Server® Enterprise IMAGE column can be inserted into a LONG BINARY column using the INSERT...LOCATION command. All IMAGE data inserted is silently right truncated at 2147483648 bytes (2 gigabytes).

Data type conversion

There are no implicit data type conversions from the LONG BINARY data type to another non-LONG BINARY data type, except to the BINARY and VARBINARY data types for INSERT and UPDATE. There are implicit conversions to LONG BINARY data type from TINYINT, SMALLINT, INTEGER, UNSIGNED INTEGER, BIGINT, UNSIGNED BIGINT, CHAR, and VARCHAR data types. There are no implicit conversions from BIT, REAL, DOUBLE, or NUMERIC data types to LONG BINARY data type. Implicit conversion can be controlled using the CONVERSION_MODE database option.

The currently supported byte substring functions for the LONG BINARY data type are accepted as input for implicit conversion for the INSERT and UPDATE statements. See the section "Function support of LONG BINARY data type" on page 11 for more information on functions that support LONG BINARY.

The LONG BINARY data type can be explicitly converted to BINARY or VARBINARY. No other explicit data type conversions (for example, using the CAST or CONVERT function) exist either to or from the LONG BINARY data type.

Truncation of LONG BINARY data during conversion of LONG BINARY to BINARY or VARBINARY is handled the same way the truncation of BINARY and VARBINARY data is handled. If the STRING_RTRUNCATION option is ON, then any right truncation (of any values, not just non-space characters) on INSERT or UPDATE of a binary column results in a truncation error and a rollback.

LONG BINARY columns in queries

In WHERE clauses of the SELECT statement, LONG BINARY columns can only be used in IS NULL and IS NOT NULL expressions, in addition to the BYTE_LENGTH64, BYTE_SUBSTR64, BYTE_SUBSTR, BIT_LENGTH, OCTET_LENGTH, CHARINDEX, and LOCATE functions.

LONG BINARY columns cannot be used in the SELECT statement clauses ORDER BY, GROUP BY, and HAVING or with the DISTINCT keyword.

LIKE predicates are not supported on LONG BINARY (BLOB) columns. If you attempt to search for a pattern in a LONG BINARY column using a LIKE predicate, the error `“Invalid data type comparison in predicate”` is returned.

See “Function support of LONG BINARY data type” on page 11 for more information on LONG BINARY data and functions.

Variables of LONG BINARY data type

An inbound LONG BINARY variable (a host variable or SQL variable used by IQ) is limited to a length of 32767 (32K-1) bytes and is handled by IQ as VARBINARY data. An error is raised, if the inbound variable length is greater than 32767 bytes.

An outbound LONG BINARY variable (a variable set by IQ) has a maximum length of 2GB.

Monitoring performance of LONG BINARY columns

The Sybase IQ performance monitor displays performance data for LONG BINARY columns.

CHAPTER 3

Character Large Object (CLOB) data

About this chapter

This chapter describes the characteristics of the LONG VARCHAR data type column, which stores Sybase IQ Character Large Object data.

Contents

Large Object data types LONG VARCHAR and CLOB

Character Large Object (CLOB) data in Sybase IQ is stored in columns of data type LONG VARCHAR or CLOB.

An individual LONG VARCHAR data value can have a length ranging from zero (0) to 512TB (terabytes) for an IQ page size of 128KB or 2PB (petabytes) for an IQ page size of 512KB. (The maximum length is equal to 4GB multiplied by the database page size.) The IQ database must be created with an IQ page size of at least 128KB (131072 bytes) in order to accommodate a table with LONG VARCHAR data.

A table or database can contain any number of LONG VARCHAR columns up to the supported maximum columns per table and maximum columns per database, respectively.

Sybase IQ supports both single byte and multibyte LONG VARCHAR data.

LONG VARCHAR columns can be either NULL or NOT NULL and can store zero-length values. The domain CLOB is a LONG VARCHAR data type that allows NULL. To create a non-null LONG VARCHAR column, explicitly specify NOT NULL in the column definition.

You can create a LONG VARCHAR column using the domain CLOB, when you create a table or add a column to an existing table. For example:

```
CREATE TABLE lvtab (c1 INTEGER, c2 CLOB,
                    c3 CLOB NOT NULL);
ALTER TABLE lvtab ADD c4 CLOB;
```

A WORD (WD) index can be created on a LONG VARCHAR column. Other non-FP index types and join indexes cannot be constructed on a LONG VARCHAR column.

A LONG VARCHAR column can be modified using the UPDATE, INSERT...VALUES, INSERT...SELECT, LOAD TABLE, DELETE, TRUNCATE, SELECT...INTO and INSERT...LOCATION SQL statements. Positioned updates and deletes are not supported on LONG VARCHAR columns.

An Adaptive Server Enterprise TEXT column can be inserted into a LONG VARCHAR column using the INSERT...LOCATION command. All TEXT data inserted is silently right truncated at 2147483648 bytes (2 gigabytes).

Data type conversion

There are no implicit data type conversions from the LONG VARCHAR data type to another non-LONG VARCHAR data type, except LONG BINARY, and CHAR and VARCHAR for INSERT and UPDATE only. There are implicit conversions to LONG VARCHAR data type from CHAR and VARCHAR data types. There are no implicit conversions from BIT, REAL, DOUBLE, NUMERIC, TINYINT, SMALLINT, INT, UNSIGNED INT, BIGINT, UNSIGNED BIGINT, BINARY, VARBINARY, or LONG BINARY data types to LONG VARCHAR data type. Implicit conversion can be controlled using the CONVERSION_MODE database option.

The currently supported string functions for the LONG VARCHAR data type are accepted as input for implicit conversion for the INSERT and UPDATE statements. See the section “Function support of LONG VARCHAR data type” on page 12 for more information on functions that support LONG VARCHAR.

The LONG VARCHAR data type can be explicitly converted to CHAR and VARCHAR. No other explicit data type conversions (for example, using the CAST or CONVERT function) exist either to or from the LONG VARCHAR data type.

Truncation of LONG VARCHAR data during conversion of LONG VARCHAR to CHAR is handled the same way the truncation of CHAR data is handled. If the STRING_RTRUNCATION option is ON and string right truncation of non-spaces occurs, a truncation error is reported and a rollback is performed. Trailing partial multibyte characters are replaced with spaces on conversion.

Truncation of LONG VARCHAR data during conversion of LONG VARCHAR to VARCHAR is handled the same way the truncation of VARCHAR data is handled. If the STRING_RTRUNCTION option is ON and string right truncation of non-spaces occurs, a truncation error is reported and a rollback is performed. Trailing partial multibyte characters are truncated on conversion.

LONG VARCHAR columns in queries

In WHERE clauses of the SELECT statement, LONG VARCHAR columns can only be used in IS NULL and IS NOT NULL expressions, in addition to the BIT_LENGTH, CHAR_LENGTH, CHAR_LENGTH64, CHARINDEX, LOCATE, OCTET_LENGTH, PATINDEX, SUBSTRING64, and SUBSTRING functions.

You can use the LIKE predicate to search for a pattern on a LONG VARCHAR column. All patterns of 126 characters or less are supported. Patterns of length greater than 254 characters are not supported. Some patterns of length between 127 and 254 characters are supported, depending on the contents of the pattern.

LONG VARCHAR columns cannot be used in the SELECT statement clauses ORDER BY, GROUP BY, and HAVING or with the DISTINCT keyword (SELECT DISTINCT and COUNT DISTINCT).

See “Function support of LONG VARCHAR data type” on page 12 for more information on LONG VARCHAR data and functions.

WD index support

The following enhancements support the WORD (WD) index on LONG VARCHAR (CLOB) columns:

- Sybase Central Java allows you to create a WD index on columns of CHAR, VARCHAR, and LONG VARCHAR data types.
- The widest column supported by the WD index increased from 32767 bytes up to the maximum width for a LOB column. (The maximum length is equal to 4GB multiplied by the database page size.)

  The maximum word length supported by Sybase IQ is still 255 bytes.
- All sp_iqcheckdb options for WD indexes over CHAR and VARCHAR columns are also supported for LONG VARCHAR (CLOB) columns, including allocation, check, and verify modes.
- The sp_iqrebuildindex stored procedure supports rebuilding a WD index over a LONG VARCHAR (CLOB) column.

Note that Chinese text or documents in a binary format still require ETL pre-processing to locate and transform the words into a form that can be parsed by the WD index.

CONTAINS predicate support

Using the CONTAINS predicate, you can now search for string constants of maximum length 255 characters on a LONG VARCHAR (CLOB) column, in addition to CHAR and VARCHAR columns.

Note that the CONTAINS predicate is not supported on LONG BINARY (BLOB) columns. If you attempt to search for a string in a LONG BINARY column using a CONTAINS predicate, an error is returned.

For more information on CONTAINS string searches, see “CONTAINS conditions” in Chapter 2, “SQL Language Elements” of the *Reference: Building Blocks, Tables, and Procedures*.

Variables of LONG VARCHAR data type

An inbound LONG VARCHAR variable (a host variable or SQL variable used by IQ) is limited to a length of 32767 (32K-1) bytes. An error is raised, if the inbound variable length is greater than 32767 bytes.

An outbound LONG VARCHAR variable (a variable set by IQ) has a maximum length of 2GB.

Monitoring performance of LONG VARCHAR columns

The Sybase IQ performance monitor displays performance data for LONG VARCHAR columns.

CHAPTER 4 Function Support

About this chapter

This chapter describes the Sybase IQ functions that support the LONG BINARY and LONG VARCHAR data types.

In addition to the functions described in this chapter, the BFILE function can be used to extract LOB data. See "Exporting large object data" on page 23 for more information on the BFILE function.

Contents

Function support of LONG BINARY data type

The functions BYTE_LENGTH64, BYTE_SUBSTR64, and BYTE_SUBSTR support LONG BINARY data.

The LONG VARCHAR functions BIT_LENGTH, CHARINDEX, LOCATE, OCTET_LENGTH, and SUBSTRING64 also support LONG BINARY data. See "Function support of LONG VARCHAR data type" on page 12 for descriptions of these functions.

BYTE_LENGTH64 function

Function

The BYTE_LENGTH64 function returns an unsigned 64 bit value containing the byte length of the LONG BINARY column parameter.

Syntax

BYTE_LENGTH64(*large-object-column*)

Parameters

large-object-column The name of a LONG BINARY column.

Usage

The BYTE_LENGTH64 function also supports the LONG VARCHAR data type.

BYTE_SUBSTR64 and BYTE_SUBSTR functions

Function

The BYTE_SUBSTR64 and BYTE_SUBSTR functions return the long binary byte substring of the LONG BINARY column parameter.

Syntax

BYTE_SUBSTR64(*large-object-column*, *start*, *length*)

BYTE_SUBSTR(*large-object-column*, *start*, *length*)

Parameters

large-object-column The name of a LONG BINARY column.

start An integer expression indicating the start of the substring. A positive integer starts from the beginning of the string, with the first byte at position 1. A negative integer specifies a substring starting from the end of the string, with the final byte at position -1.

length An integer expression indicating the length of the substring. A positive length specifies the number of bytes to return, starting at the *start* position. A negative length specifies the number of bytes to return, ending at the *start* position.

Usage

- Nesting of the functions BYTE_LENGTH64, BYTE_SUBSTR64, and BYTE_SUBSTR is not supported.
- The BYTE_SUBSTR64 and BYTE_SUBSTR functions also support the LONG VARCHAR data type.

Aggregate function support of LONG BINARY columns

Only the aggregate function COUNT (*) is supported for LONG BINARY columns. The COUNT DISTINCT parameter is not supported. An error is returned if a LONG BINARY column is used with the MIN, MAX, AVG, or SUM aggregate functions.

Function support of LONG VARCHAR data type

The functions BIT_LENGTH, CHAR_LENGTH, CHAR_LENGTH64, CHARINDEX, LOCATE, OCTET_LENGTH, SUBSTRING, SUBSTRING64, and PATINDEX support LONG VARCHAR data.

The LONG BINARY functions BYTE_LENGTH64, BYTE_SUBSTR64, and BYTE_SUBSTR also support LONG VARCHAR data. See “Function support of LONG BINARY data type” on page 11 for descriptions of these functions.

See “CHAR_LENGTH function [String],” “CHARINDEX function [String],” “LOCATE function [String],” “SUBSTRING function [String],” and “PATINDEX function [String]” in Chapter 4, “SQL Functions” of *Reference: Building Blocks, Tables, and Procedures* for full descriptions and examples of the functions that support LONG VARCHAR.

BIT_LENGTH function

Function

The BIT_LENGTH function returns an unsigned 64 bit value containing the bit length of the LONG VARCHAR column parameter. If the argument is NULL, BIT_LENGTH returns NULL.

Syntax

BIT_LENGTH(*column-name*)

Parameters

column-name The name of a LONG VARCHAR column.

Usage

The BIT_LENGTH function supports all Sybase IQ data types.

CHAR_LENGTH function

Function

The CHAR_LENGTH function returns a signed 32 bit value containing the character length of the LONG VARCHAR column parameter, including the trailing blanks.

Syntax

CHAR_LENGTH(*long-varchar-column*)

Parameters

long-varchar-column The name of a LONG VARCHAR column.

Usage

- If the argument is NULL, CHAR_LENGTH returns NULL.
- If the character length exceeds 2147483647, an error is returned.

CHAR_LENGTH64 function

Function

The CHAR_LENGTH64 function returns an unsigned 64 bit value containing the character length of the LONG VARCHAR column parameter, including the trailing blanks.

Syntax

CHAR_LENGTH64(*long-varchar-column*)

Parameters

long-varchar-column The name of a LONG VARCHAR column.

Usage

If the argument is NULL, CHAR_LENGTH64 returns NULL.

CHARINDEX function

Function

The CHARINDEX function returns a 64 bit signed integer containing the position of the first occurrence of the specified string in a LONG VARCHAR column. CHARINDEX returns a 32 bit signed integer position for CHAR and VARCHAR columns.

Syntax

CHARINDEX(*string-expression*, *long-varchar-column*)

Parameters

string-expression The string for which you are searching. This string is limited to 255 bytes.

long-varchar-column The name of the LONG VARCHAR column.

Usage

- All the positions or offsets, returned or specified, in the CHARINDEX function are always character offsets and may be different from the byte offset for multibyte data.
- If the LONG VARCHAR cell being searched contains more than one instance of the string, CHARINDEX returns only the position of the first instance.
- If the column does not contain the string, the CHARINDEX function returns zero (0).
- Searching for a string longer than 255 bytes returns NULL.
- Searching for a zero-length string returns 1.
- If any of the arguments is NULL, the result is NULL.
- CHARINDEX also supports searching LONG BINARY columns.

See also

For a full description and an example of the CHARINDEX function, see “CHARINDEX function [String]” in Chapter 4, “SQL Functions” of *Reference: Building Blocks, Tables, and Procedures*.

LOCATE function

Function

The LOCATE function returns a 64 bit signed integer containing the position of the specified string in a LONG VARCHAR column. LOCATE returns a 32 bit signed integer position for CHAR and VARCHAR columns.

Syntax

LOCATE(*long-varchar-column*, *string-expression*
[, *numeric-expression*])

Parameters

long-varchar-column The name of the LONG VARCHAR column to search.

string-expression The string for which you are searching. This string is limited to 255 bytes.

numeric-expression The character position or offset at which to begin the search in the string. The *numeric-expression* is a 64 bit signed integer for LONG VARCHAR and LONG BINARY columns and is a 32 bit signed integer for CHAR, VARCHAR, and BINARY columns. The first character is position 1. If the starting offset is negative, LOCATE returns the last matching string offset, rather than the first. A negative offset indicates how much of the end of the string to exclude from the search. The number of characters excluded is calculated as (-1 * offset) - 1.

Usage

- All the positions or offsets, returned or specified, in the LOCATE function are always character offsets and may be different from the byte offset for multibyte data.
- If the LONG VARCHAR cell being searched contains more than one instance of the string:
  - If *numeric-expression* is specified, LOCATE starts the search at that offset in the string.
  - If *numeric-expression* is not specified, LOCATE returns only the position of the first instance.
- If the column does not contain the string, the LOCATE function returns zero (0).
- Searching for a string longer than 255 bytes returns NULL.
- Searching for a zero-length string returns 1.
- If any of the arguments is NULL, the result is NULL.
- LOCATE also supports searching LONG BINARY columns.

See also

For a full description and examples of the LOCATE function, see “LOCATE function [String]” in Chapter 4, “SQL Functions” of *Reference: Building Blocks, Tables, and Procedures*.

OCTET_LENGTH function

Function

The OCTET_LENGTH function returns an unsigned 64 bit value containing the byte length of the LONG VARCHAR column parameter.

Syntax

OCTET_LENGTH(*column-name*)

Parameters

column-name The name of a LONG VARCHAR column.

Usage

- If the argument is NULL, OCTET_LENGTH returns NULL.
- OCTET_LENGTH supports all Sybase IQ data types.

PATINDEX function

Function

The PATINDEX function returns a 64 bit unsigned integer containing the position of the first occurrence of the specified pattern in a LONG VARCHAR column. PATINDEX returns a 32 bit unsigned integer position for CHAR and VARCHAR columns.

Syntax

PATINDEX('*%pattern%*', *long-varchar-column*)

Parameters

pattern The pattern for which you are searching. This string is limited to 126 bytes for patterns with wildcards. If the leading percent wildcard is omitted, PATINDEX returns one (1) if the pattern occurs at the beginning of the column value, and zero (0) if the pattern does not occur at the beginning of the column value. Similarly, if the trailing percent wildcard is omitted, the pattern should occur at the end of the column value. The pattern uses the same wildcards as the LIKE comparison.

Patterns without wildcards (percent % or underscore _) can be up to 255 bytes in length.

long-varchar-column The name of the LONG VARCHAR column.

Usage

- All the positions or offsets, returned or specified, in the PATINDEX function are always character offsets and may be different from the byte offset for multibyte data.
- If the LONG VARCHAR cell being searched contains more than one instance of the string pattern, PATINDEX returns only the position of the first instance.
- If the column does not contain the pattern, the PATINDEX function returns zero (0).
- Searching for a pattern longer than 126 bytes returns NULL.

- Searching for a zero-length pattern returns 1.
- If any of the arguments is NULL, the result is zero (0).
- PATINDEX does not support searching LONG BINARY columns.

See also

- For a full description and examples of the PATINDEX function, see “PATINDEX function [String]” in Chapter 4, “SQL Functions”of *Reference: Building Blocks, Tables, and Procedures*.
- For more information on LIKE comparisons, see “LIKE conditions” in Chapter 2, “SQL Language Elements” of *Reference: Building Blocks, Tables, and Procedures*.

SUBSTRING function

Function

The SUBSTRING function returns a variable length character string of the LONG VARCHAR column parameter. If any of the arguments are NULL, SUBSTRING returns NULL.

Syntax

SUBSTRING(*long-varchar-column*, *start* [, *length*])

Parameters

long-varchar-column The name of a LONG VARCHAR column.

start An integer expression indicating the start of the substring. A positive integer starts from the beginning of the string, with the first character at position 1. A negative integer specifies a substring starting from the end of the string, with the final character at position -1.

length An integer expression indicating the character length of the substring. A positive length specifies the number of characters to return, starting at the *start* position. A negative length specifies the number of characters to return, ending at the *start* position.

SUBSTRING64 function

Function

The SUBSTRING64 function returns a variable length character string of the LONG VARCHAR column parameter.

Syntax

SUBSTRING64(*large-object-column*, *start* [, *length*])

Parameters

large-object-column The name of a LONG VARCHAR column.

start An 8 byte integer indicating the start of the substring. SUBSTRING64 interprets a negative or zero *start* offset as if the string were padded on the left with "non-characters." The first character starts at position 1.

length An 8 byte integer indicating the length of the substring. If *length* is negative, an error is returned.

Example

Given a column named col1 which contains the string ('ABCDEFG'), the SUBSTRING64 function returns the following values:

`SUBSTRING64( col1, 2, 4 )` returns the string 'BCDE'

`SUBSTRING64( col1, 1, 3 )` returns the string 'ABC'

`SUBSTRING64( col1, 0, 3 )` returns the string 'AB'

`SUBSTRING64( col1, -1, 3 )` returns the string 'A'

Usage

- If any of the arguments are NULL, SUBSTRING64 returns NULL.
- Nesting of the functions SUBSTRING64, SUBSTRING, BYTE_SUBSTR, and BYTE_SUBSTR64 is not supported.
- SUBSTRING64 also supports the LONG BINARY data type.

Aggregate function support of LONG VARCHAR columns

Only the aggregate function COUNT (*) is supported for LONG VARCHAR columns. The COUNT DISTINCT parameter is not supported. An error is returned if a LONG VARCHAR column is used with the MIN, MAX, AVG, or SUM aggregate functions.

CHAPTER 5 Stored Procedure Support

About this chapter

This chapter describes the stored procedure support for the LONG BINARY (BLOB) and LONG VARCHAR (CLOB) data type columns.

Contents

Controlling large object data compression

The sp_iqsetcompression stored procedure controls the compression of columns of data type LONG BINARY and LONG VARCHAR when writing database buffers to disk and allows you to disable compression. This functionality saves CPU cycles, because certain data formats stored in a LONG BINARY or LONG VARCHAR column (for example, JPG files) are already compressed and gain nothing from additional compression. The sp_iqshowcompression stored procedure displays the compression setting of large object columns.

sp_iqsetcompression procedure

Function

Sets compression of data in columns of LONG BINARY (BLOB) and LONG VARCHAR (CLOB) data types.

Syntax

sp_iqsetcompression (*owner*, *table*, *column*, *on_off_flag*)

Permissions

Requires DBA authority.

Description

The sp_iqsetcompression system stored procedure provides control of compression of LONG BINARY (BLOB) and LONG VARCHAR (CLOB) data type columns. The compression setting only applies to IQ base tables.

A side effect of sp_iqsetcompression is that a COMMIT occurs after the compression setting is changed.

Table 5-1: sp_iqsetcompression parameters

Name	Description
owner	Owner of the table for which you are setting compression
table	Table for which you are setting compression
column	Column for which you are setting compression
on_off_flag	Compression setting: ON enables compression, OFF disables compression

Example

For this example, assume the following table definition:

```
CREATE TABLE USR.pixTable (picID INT NOT NULL,
picJPG LONG BINARY NOT NULL);
```

To turn off compression on the LOB column picJPG, call the sp_iqsetcompression procedure using the following command (you must have DBA permission):

```
CALL sp_iqsetcompression('USR', 'pixTable', 'picJPG',
'OFF') ;
```

This command returns no rows.

sp_iqshowcompression procedure

Function

Displays compression settings for columns of LONG BINARY (BLOB) and LONG VARCHAR (CLOB) data types.

Syntax

sp_iqshowcompression (*owner*, *table*, *column*)

Permissions

Requires DBA authority.

Description

Returns the column name and compression setting. Compression setting values are 'ON' (compression enabled) and 'OFF' (compression disabled).

Table 5-2: sp_iqshowcompression parameters

Name	Description
owner	Owner of the table for which you are setting compression
table	Table for which you are setting compression
column	Column for which you are setting compression

Example

For this example, assume the following table definition:

```
CREATE TABLE USR.pixTable (picID INT NOT NULL,
picJPG LONG BINARY NOT NULL);
```

To check the compression status of the columns in the pixTable table, call the sp_iqshowcompression procedure using the following command (you must have DBA permission):

```
CALL sp_iqshowcompression('USR', 'pixTable',
'picJPG') ;
```

This command returns one row:

```
'picJPG','ON'
```

Displaying information about large object columns

The stored procedure sp_iqindexsize displays the size of an individual LONG BINARY and LONG VARCHAR column.

Size of a LONG BINARY column

The following output shows a LONG BINARY column with approximately 42GB of data. The page size is 128KB. The largelob Info type is in the last row:

```
Username    Indexname               Type Info     KBytes    Pages    Compressed
                                                                       Pages
DBA    test10.DBA.ASIQ_IDX_T128_C3_FP FP Total     42953952  623009  622923
DBA    test10.DBA.ASIQ_IDX_T128_C3_FP FP vdo       0         0       0
DBA    test10.DBA.ASIQ_IDX_T128_C3_FP FP bt        0         0       0
DBA    test10.DBA.ASIQ_IDX_T128_C3_FP FP garray    0         0       0
DBA    test10.DBA.ASIQ_IDX_T128_C3_FP FP bm        136       2       1
DBA    test10.DBA.ASIQ_IDX_T128_C3_FP FP barray    2312      41      40
DBA    test10.DBA.ASIQ_IDX_T128_C3_FP FP dpstore   170872    2551    2549
DBA    test10.DBA.ASIQ_IDX_T128_C3_FP FP largelob  42780632  620415  620333
```

In this example, the compression ratio is 42953952/(623009*128) = 53.9%.

Size of a LONG VARCHAR column

The following output shows a LONG VARCHAR column with approximately 42GB of data. The page size is 128KB. The largelob Info type is in the last row:

```
Username    Indexname               Type Info     KBytes    Pages    Compressed
                                                                       Pages
DBA    test10.DBA.ASIQ_IDX_T128_C3_FP FP Total     42953952  623009  622923
DBA    test10.DBA.ASIQ_IDX_T128_C3_FP FP vdo       0         0       0
DBA    test10.DBA.ASIQ_IDX_T128_C3_FP FP bt        0         0       0
DBA    test10.DBA.ASIQ_IDX_T128_C3_FP FP garray    0         0       0
DBA    test10.DBA.ASIQ_IDX_T128_C3_FP FP bm        136       2       1
DBA    test10.DBA.ASIQ_IDX_T128_C3_FP FP barray    2312      41      40
DBA    test10.DBA.ASIQ_IDX_T128_C3_FP FP dpstore   170872    2551    2549
DBA    test10.DBA.ASIQ_IDX_T128_C3_FP FP largelob  42780632  620415  620333
```

In this example, the compression ratio is 42953952/(623009*128) = 53.9%.

CHAPTER 6 Moving Large Object Data

About this chapter

This chapter describes how to export and load large object data in Sybase IQ.

Contents

Exporting large object data

The IQ data extraction facility includes the BFILE function, which allows you to extract individual LONG BINARY and LONG VARCHAR cells to individual operating system files on the server. The BFILE function can be used with or without the data extraction facility.

Syntax:

BFILE(*file-name-expression*, *large-object-column*)

Parameters:

file-name-expression The name of the output file into which the LONG BINARY or LONG VARCHAR data is written. This file name can be up to (32K -1) bytes in length, but must be a valid pathname supported by the file system.

large-object-column The name of the LONG BINARY or LONG VARCHAR column.

BFILE returns the following values:

- 1, if the file is successfully written
- 0, if the file is not successfully opened or written
- NULL, if the LONG BINARY or LONG VARCHAR cell value is NULL

If the LONG BINARY or LONG VARCHAR cell value is NULL, no file is opened and no data is written.

The file path is relative to where the server was started and the open and write operations execute with the permissions of the server process. Tape devices are not supported for the BFILE output file.

LONG BINARY and LONG VARCHAR cells retrieved other than with the BFILE function (that is, retrieved through the client/server database connection later) are limited in size to a maximum length of 2GB. The SUBSTRING64 or BYTE_SUBSTR64 function must be used to retrieve LONG BINARY cells greater than 2GB using a SELECT (SELECT, OPEN CURSOR). The SUBSTRING64 function must be used to retrieve LONG VARCHAR cells greater than 2GB using a SELECT (SELECT, OPEN CURSOR). Also note that some connection drivers, for example ODBC, JDBC, and Open Client™, do not allow more than 2GB to be returned in one SELECT.

BFILE example

This example shows the use of the BFILE function to extract data from the LONG BINARY column lobcol, which is created and loaded in the “Load example” on page 26. The following command writes the data in files which can be used as secondary files in a load.

```
SELECT c1, filename, ext,
‘../myoutput/’ + TRIM(filename) + ‘.’ + TRIM(ext) fname,
BFILE(fname, lobcol)
FROM ltab
   WHERE lobcol IS NOT NULL
   AND ext IS NOT NULL
```

This command generates the file name with extension *boston.jpg* for lobcol in row 1 and the file name with extension *map_of_concord.bmp* for lobcol in row 2.

Loading large object data

LONG BINARY and LONG VARCHAR data can be loaded using extended syntax of the LOAD TABLE statement. You can specify a secondary load file in the primary load file. Each individual secondary data file contains exactly one LONG BINARY or LONG VARCHAR cell value.

Extended LOAD TABLE syntax

LOAD [**INTO**] **TABLE** [*owner*].*table-name*
... (*column-name load-column-specification* [, ...])
... **FROM** '*filename-string*' [, ...]
... [**QUOTES** { **ON** | **OFF** }]
... **ESCAPES OFF**
... [**FORMAT** { **ascii** | **binary** | **bcp** }]
... [**DELIMITED BY** '*string*']
...

load-column-specification:
...
| { **BINARY** | **ASCII** } **FILE**(*integer*)
| { **BINARY** | **ASCII** } **FILE** (‘*string*‘)

The keywords BINARY FILE (for LONG BINARY) or ASCII FILE (for LONG VARCHAR) specify to the load that the primary input file for the column contains the path of the secondary file (which contains the LONG BINARY or LONG VARCHAR cell value), rather than the LONG BINARY or LONG VARCHAR data itself. The secondary file pathname can be either fully qualified or relative. If the secondary file pathname is not fully qualified, then the path is relative to the directory in which the server was started. Tape devices are not supported for the secondary file. Note that IQ supports loading LONG BINARY and LONG VARCHAR values of length less than or equal to 32767 bytes in the primary load file.

For LOAD TABLE FORMAT BCP, the load specification may contain only column names, NULL, and ENCRYPTED. This means that you cannot use secondary files when loading LONG BINARY and LONG VARCHAR columns using the LOAD TABLE FORMAT BCP option. For syntax and usage details on the LOAD TABLE FORMAT BCP option, see LOAD TABLE statement in Chapter 1, “SQL Statements”of *Reference: Statements and Options*.

Load example

This example shows the SQL statements to create and load a table with LONG BINARY data.

```
CREATE TABLE ltab (c1 INT, filename CHAR(64),
    ext CHAR(6), lobcol LONG BINARY NULL);
LOAD TABLE ltab (
    c1,
    filename,
    ext NULL(‘NULL’),
    lobcol BINARY FILE (‘,’) NULL(‘NULL’)
)
FROM ‘abc.inp’
QUOTES OFF ESCAPES OFF;
```

The primary file *abc.inp* contains the following data:

```
1,boston,jpg,/s1/loads/lobs/boston.jpg,
2,map_of_concord,bmp,/s1/loads/maprs/concord.bmp,
3,zero length test,NULL,,
4,null test,NULL,NULL,
```

After the LONG BINARY data is loaded into table ltab, the first and second rows for column lobcol contain the contents of files *boston.jpg* and *concord.bmp*, respectively. The third and fourth rows contain a zero-length value and NULL, respectively.

Controlling load errors

The database option SECONDARY_FILE_ERROR allows you to specify the action of the load, if an error occurs while opening or reading from a secondary BINARY FILE or ASCII FILE.

If the option SECONDARY_FILE_ERROR is ON, the load will rollback, if an error occurs while opening or reading from a secondary BINARY FILE or ASCII FILE.

If the option SECONDARY_FILE_ERROR is OFF, the load continues, regardless of any errors that occur while opening or reading from a secondary BINARY FILE or ASCII FILE. The LONG BINARY or LONG VARCHAR cell is left with the following value:

- NULL, if the column allows nulls
- zero-length value, if the column does not allow nulls

The allowed values of the SECONDARY_FILE_ERROR option are ON and OFF. The default value is OFF. This option can be set for the PUBLIC group or temporary by any user and takes effect immediately.

When logging integrity constraint violations to the load error ROW LOG file, the information logged for a LONG BINARY or LONG VARCHAR column is:

- actual text as read from the primary data file, if the logging occurs within the first pass of the load operation
- zero-length value, if the logging occurs within the second pass of the load operation

Stripping trailing blanks

The LOAD TABLE...STRIP option has no effect on LONG VARCHAR data. Trailing blanks are *not* stripped from LONG VARCHAR data, even if the STRIP option is on.

Enclosing quotes

The LOAD TABLE...QUOTES option does not apply to loading LONG BINARY (BLOB) or LONG VARCHAR (CLOB) data from the secondary file, regardless of its setting, A leading or trailing quote is loaded as part of CLOB data. Two consecutive quotes between enclosing quotes are loaded as two consecutive quotes with the QUOTES ON option.

Truncating partial multibyte character data

Partial multibyte LONG VARCHAR data is truncated during the load according to the value of the TRIM_PARTIAL_MBC database option:

- If TRIM_PARTIAL_MBC is ON, a partial multibyte character is truncated for both primary data and the LOAD with ASCII FILE option.
- If TRIM_PARTIAL_MBC is OFF, the LOAD with ASCII FILE option handles the partial multibyte character according to the value of the SECONDARY_FILE_ERROR database option.

Table 6-1 lists how a trailing multibyte character is loaded, depending on the values of the TRIM_PARTIAL_MBC and SECONDARY_FILE_ERROR database options.

Table 6-1: Partial multibyte character on loading LONG VARCHAR with ASCII FILE option

TRIM_PARTIAL_MBC	SECONDARY_FILE_ERROR	Trailing partial multibyte character found
ON	ON/OFF	Trailing partial multibyte character truncated
OFF	ON	Cell — null, if null allowed LOAD error — rollback, if null not allowed
OFF	OFF	Cell — null, if null allowed Cell — 0 length, if null not allowed

CHAPTER 7 Compatibility and Conformance

About this chapter

This chapter describes the compatibility and conformance to standards of large object data in Sybase IQ.

Contents

Compatibility

SQL Anywhere (SA) can store large objects (up to a 2GB maximum length) in columns of data type LONG VARCHAR or LONG BINARY. The support of these data types by SQL Anywhere is SQL92 compliant. SQL Anywhere does not support the BYTE_LENGTH64, BYTE_SUBSTR64, BFILE, BIT_LENGTH, OCTET_LENGTH, CHAR_LENGTH64, and SUBSTRING64 functions.

Adaptive Server Enterprise (ASE) can store large textual objects (up to a 2GB maximum length) and large binary objects (up to a 2GB maximum length) in columns of data type TEXT or IMAGE, respectively. The support of these data types by Adaptive Server Enterprise is SQL92 compliant.

Conformance to standards

Sybase IQ LONG BINARY and LONG VARCHAR functionality conforms to the CORE level of the SQL99 standard.

CHAPTER 8 Error and Warning Messages

About this chapter

This chapter describes the error and warning messages that may be returned when you are working with LONG BINARY and LONG VARCHAR columns.

Contents

Error 1000195

Message text

LOAD specification ‘%2’ only valid for column(s) having datatype ‘%3’. %1

Item	Value
SQLCode	-1000195L
Constant	EMSG_BINARYFILE
SQLState	QDB95
ODBC 2 State	ERROR
ODBC 3 State	ERROR
Sybase Error Code	20855
Severity Code	14
Parameter 1	location of the exception
Parameter 2	type of load specification
Parameter 3	data type of column

Probable cause

The named load specification in a LOAD TABLE statement is only valid for columns with the given data type.

Error 1000198

Message text

Cannot create join index with table(s) having column(s) of datatype %2. %1

Item	Value
SQLCode	-1000198L
Constant	EMSG_CANNOT_CREATE_JOIN_INDEX
SQLState	QDB98
ODBC 2 State	ERROR
ODBC 3 State	ERROR
Sybase Error Code	20858
Severity Code	14
Parameter 1	location of the exception
Parameter 2	data type of column

Probable cause

This error is reported when you attempt to create a join index on a table that has one or more LONG VARCHAR or LONG BINARY data type columns. The JOIN INDEX functionality is supported for most data types. There are a few data types, however, for which this functionality is not supported (for example, LONG BINARY and LONG VARCHAR).

Error 1001013

Message text

Invalid data type comparison %2, %1

Item	Value
SQLCode	-1001013L
Constant	EMSG_TYPECOMPAREERROR
SQLState	QFA13
ODBC 2 State	ERROR
ODBC 3 State	ERROR
Sybase Error Code	20522
Severity Code	14
Parameter 1	location of the exception

Probable cause

This error is reported if you attempt to search for a pattern in a LONG BINARY column using a LIKE predicate. LIKE predicates are not supported on LONG BINARY (BLOB) columns.

Error 1001051

Message text

Query returns %2 data > 2GB. Use %3 %1

Item	Value
SQLCode	-1001051L
Constant	EMSG_LOB_OVER_2G_W_ARG
SQLState	QFA47
ODBC 2 State	ERROR
ODBC 3 State	ERROR
Sybase Error Code	21097
Severity Code	14
Parameter 1	SA parse source code line
Parameter 2	function recommended
Parameter 3	long binary or long varchar data type

Probable cause

This error is reported when a query attempts to return a LONG BINARY or LONG VARCHAR value greater than 2 gigabytes.

Error 1001052

Message text

Parameter %2 must be long binary/varchar type. %3 %1

Item	Value
SQLCode	-1001052L
Constant	EMSG_ONLY_SUPPORT_LOB_W_ARG
SQLState	QFA48
ODBC 2 State	ERROR
ODBC 3 State	ERROR
Sybase Error Code	21098
Severity Code	14
Parameter 1	SA parse source code line
Parameter 2	LOB argument name
Parameter 3	recommended function name

Probable cause

This error is reported when an invalid data type is used for a Large Object (LOB) function parameter.

Error 1001053

Message text

Wrong number of parameters to function %2 %1

Item	Value
SQLCode	-1001053L
Constant	EMSG_WRONG_NUM_PARAMS_W_ARG
SQLState	QFA49
ODBC 2 State	ERROR
ODBC 3 State	ERROR
Sybase Error Code	21099
Severity Code	14
Parameter 1	SA parse source code line
Parameter 2	function name

Probable cause

This error is reported when a Large Object (LOB) function is passed an incorrect number of arguments.

Error 1001054

Message text

You cannot specify long binary/varchar column in the ORDER/GROUP by clause or in an aggregate function. %1

Item	Value
SQLCode	-1001054L
Constant	EMSG_LOB_NOT_ALLOWED_GROUP
SQLState	QFA50
ODBC 2 State	ERROR
ODBC 3 State	ERROR
Sybase Error Code	21100
Severity Code	14
Parameter 1	location of the exception

Probable cause

This error is reported when you attempt to use a LONG BINARY column in an ORDER BY, GROUP BY, or aggregation clause.

Warning 1001055

Message text

An error occurred loading %1 column, %2, for %3, rowid %4.

Item	Value
SQLCode	1001055L
Constant	EMSG_LOB_LOAD_ERROR_WARN
SQLState	QFA51
ODBC 2 State	OK
ODBC 3 State	OK
Sybase Error Code	21101
Severity Code	10
Parameter 1	long binary or long varchar data type
Parameter 2	FP index name
Parameter 3	secondary file name
Parameter 4	rowid

Probable cause

This warning message is returned when an error is encountered either opening or reading a LONG BINARY or LONG VARCHAR secondary file during a load operation. This warning message is returned in the server log and the IQ message file when the SECONDARY_FILE_ERROR option is OFF and an error occurs.

Warning 1001056

Message text

An error occurred extracting %1 column, %2, for %3.

Item	Value
SQLCode	1001056L
Constant	EMSG_LOB_EXTRACT_ERROR_WARN
SQLState	QFA52
ODBC 2 State	OK
ODBC 3 State	OK
Sybase Error Code	21102
Severity Code	10
Parameter 1	long binary or long varchar data type
Parameter 2	FP index name
Parameter 3	secondary file name

Probable cause

This warning message is returned when you attempt to extract a LONG BINARY or LONG VARCHAR column and an error is encountered during the extract operation. This warning message is returned in the server log and the IQ message file when the SECONDARY_FILE_ERROR option is OFF and an error occurs.

Error 1001057

Message text

You must use BFILE() to extract %2 column. %1

Item	Value
SQLCode	-1001057L
Constant	EMSG_LOB_EXTRACT_USE_BFILE
SQLState	QFA53
ODBC 2 State	ERROR
ODBC 3 State	ERROR
Sybase Error Code	21103
Severity Code	14
Parameter 1	location of the exception
Parameter 2	long binary or long varchar data type

Probable cause

This error is reported when you execute a query containing a LONG BINARY or LONG VARCHAR column with the database option TEMP_EXTRACT_NAME1 set ON and you did not specify the BFILE function.

Error 1001058

Message text

The secondary file name, %2, is too long. %1

Item	Value
SQLCode	-1001058L
Constant	EMSG_LOB_SECONDARY_FILE_TOOLONG
SQLState	QFA54
ODBC 2 State	OK
ODBC 3 State	OK
Sybase Error Code	21104
Severity Code	14
Parameter 1	location of the exception
Parameter 2	secondary file name

Probable cause

This error is reported when the length of the LOAD TABLE secondary file pathname exceeds the pathname length limit of the operating system. The action taken when this error is reported depends on the value of the SECONDARY_FILE_ERROR database option.

Error 1012030

Message text

for long binary/varchar Column '%2', database page size of (%3) must be greater than %4. %1

Item	Value
SQLCode	-1012030
Constant	EMSG_CAT_PAGESIZETOOSMALL
SQLState	QUA30
ODBC 2 State	ERROR
ODBC 3 State	ERROR
Sybase Error Code	20953
Severity Code	14
Parameter 1	location of the exception
Parameter 2	column number
Parameter 3	requested page size
Parameter 4	minimum allowed page size

Probable cause

The database page size is too small to create a LONG BINARY or LONG VARCHAR column. The database page size must be 128K or greater to create a LONG BINARY or LONG VARCHAR column.

Index