SYBASE®
An SAP® Company

Sybase IQ の非構造化データ分析の概要

Sybase IQ 15.4

ドキュメント ID： DC00285-01-1540-01
改訂： 2011 年 11 月

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

目次

Sybase IQ の非構造化データ分析の概要

Sybase® IQ での非構造化データ分析について紹介し、Sybase IQ ラージ・オブジェクト・データの標準との互換性と準拠について説明します。

対象読者

このマニュアルは、Sybase IQ で非構造化データを操作するためのリファレンス資料を必要としている Sybase® IQ ユーザを対象としています。

Sybase IQ の非構造化データ分析機能に関連する、利用可能な構文、パラメータ、関数、ストアド・プロシージャ、インデックス、オプションについて説明します。このマニュアルを、Sybase IQ マニュアル・セットの他のマニュアルと一緒にリファレンスとして使用し、Sybase IQ データベース内の非構造化データの格納と取得について理解してください。

非構造化データ分析オプション

非構造化データ分析オプションは、Sybase IQ の機能を拡張することで、Sybase IQ データベース内のバイナリ・ラージ・オブジェクト (BLOB) とキャラクタ・ラージ・オブジェクト (CLOB) の格納、取得、全文検索を可能にします。

注意： この製品マニュアルで説明する非構造化データ分析機能を使用するには、正規のライセンスを取得している必要があります。

データのボリュームが増えるにつれ、リレーショナル・データベースにラージ・オブジェクト (LOB) データを格納するニーズも増大します。LOB データには次の種類があります。

- 非構造化 – データベースはデータを単に格納および抽出する。
- 半構造化 (テキストなど) – データベースはデータ構造をサポートし、操作を支援する関数 (文字列関数など) を提供する。

一般的な LOB データ・ソースとしては、イメージ、マップ、ドキュメント (PDF ファイル、ワード・プロセッサ・ファイル、プレゼンテーションなど)、オーディオ、ビデオ、XML ファイルが挙げられます。Sybase IQ では、ギガバイト (GB)、テラバイト (TB)、さらにはペタバイト (PB) のデータが含まれる個々の LOB オブジェクトを管理できます。

リレーショナル・データと非構造化データを同じロケーションに格納できるため、Sybase IQ を使用することで、同じアプリケーションとインタフェースを使用して

両方のタイプのデータにアクセスできます。Sybase IQ の全文検索機能は、非構造化データと半構造化データの処理において、テキスト・アーカイブ・アプリケーション (テキスト分析) をサポートします。

全文検索

全文検索では、**TEXT** インデックスを使用して、テーブルのローをスキャンせずに、データベース内の単語と語句を検索します。

TEXT インデックスには、インデックス・カラム内の単語の位置情報が格納されます。テキスト設定オブジェクトによって、**TEXT** インデックスの構築または更新時にインデックスに配置される単語と、全文クエリの解釈方法が制御されます。

一般的に、**TEXT** インデックスを使用して、単語または語句が含まれるローを検索する方法は、ほとんどの場合、テーブル内の各ローをスキャンする方法よりも高速です。

互換性

SQL Anywhere® Server (SA) と Adaptive Server® Enterprise (ASE) は、テキスト・ラージ・オブジェクトとバイナリ・ラージ・オブジェクトを格納します。

SQL Anywhere では、ラージ・オブジェクト (最大長 2GB) を `LONG VARCHAR` または `LONG BINARY` のデータ型のカラムに格納できます。SQL Anywhere は SQL/2003 標準に準拠して、これらのデータ型をサポートしています。SQL Anywhere は、**BYTE_LENGTH64**、**BYTE_SUBSTR64**、**BFILE**、**BIT_LENGTH**、**OCTET_LENGTH**、**CHAR_LENGTH64**、**SUBSTRING64** の各関数をサポートしていません。

Adaptive Server Enterprise では、テキスト・ラージ・オブジェクト (最大長 2GB) を `TEXT` データ型のカラムに、バイナリ・ラージ・オブジェクト (最大長 2GB) を `IMAGE` データ型のカラムにそれぞれ格納できます。Adaptive Server Enterprise は、ANSI SQL Transact-SQL® 拡張に準拠して、これらのデータ型をサポートしています。

プロキシ・テーブルの `LONG BINARY` カラムは、Microsoft SQL Server テーブルの `VARBINARY(max)` カラムにマップされます。

標準への準拠

Sybase IQ の `LONG BINARY` 機能と `LONG VARCHAR` 機能は、ISO/ANSI SQL 標準の Core レベルに準拠しています。

TEXT インデックスとテキスト設定オブジェクト

TEXT インデックスとテキスト設定オブジェクトの使用方法について説明します。

TEXT インデックスには、インデックス・カラムの単語の位置情報が格納されます。**TEXT** インデックスは、テキスト設定オブジェクトに格納されている設定を使用して作成されます。テキスト設定オブジェクトによって、無視する単語、インデックスに含める単語の最小長と最大長などの **TEXT** インデックス・データの特性が制御されます。

TEXT インデックス

全文検索では、テーブル・ローではなく、**TEXT** インデックスが検索されます。

全文検索を実行するには、検索するカラムに **TEXT** インデックスを作成する必要があります。**TEXT** インデックスには、インデックス・カラム内の単語の位置情報が格納されます。**TEXT** インデックスを使用するクエリは、ほとんどの場合、テーブル内のすべての値をスキャンする必要のあるクエリよりも高速です。

TEXT インデックスを作成するときに、**TEXT** インデックスの作成および更新時に使用するテキスト設定オブジェクトを指定できます。テキスト設定オブジェクトには、インデックスの構築方法に影響を与える設定が格納されます。テキスト設定オブジェクトを指定しない場合、データベース・サーバはデフォルトの設定オブジェクトを使用します。

TEXT インデックスを作成できるカラムのデータ型は、 CHAR、VARCHAR、LONG VARCHAR、BINARY、VARBINARY、LONG BINARY です。BINARY、VARBINARY、LONG BINARY のカラムの場合、**TEXT** インデックスで、テキスト設定と外部プレフィルタ・ライブラリを使用する必要があります。

WD インデックスと TEXT インデックスの比較

構文と機能の観点での **WD** インデックスと **TEXT** インデックスの比較

表 1 : WD インデックスと TEXT インデックス

機能	WD インデックスによるサポート	TEXT インデックスによるサポート
単語の連結	あり、構文： `tbl.col CONTAINS('great','white' ,'whale')`	あり、構文： `CONTAINS(tbl.col,'great white whale')`
一般的なブール式	あり、構文： `tbl.col CONTAINS ('great') AND ( tbl.col CONTAINS('white) OR tbl.col CONTAINS('whale') AND NOT tbl.col CONTAINS('ship'))`	あり、構文： `CONTAINS(tbl.col, 'great AND ( white OR whale AND NOT ship )')`
プレフィクスと一致する単語の検索	なし	あり、構文例： `CONTAINS (tbl.col,'whale*')`
LIKE 述部による高速化	あり、構文例： `tbl.col LIKE 'whale%'`	なし

近接する単語の検索	なし	あり、構文例：
		`CONTAINS(tbl.col, 'white BEFORE whale')`
		`CONTAINS(tbl.col, 'whale NEAR white')`
		`CONTAINS(tbl.col, ' "white whale" ')`

検索スコアに基づいた結果の順序付け	なし	あり

TEXT インデックスでは、プレフィクスと一致する単語の検索と、**LIKE** 式を使用した検索では、セマンティックが異なり、テキスト設定に応じてまったく異なる結果が返される場合があります。最小長、最大長、およびストップリストの指定によってプレフィクスの処理は制御されますが、**LIKE** のセマンティックは影響を受けません。

注意： 単語の削除が発生する場合、ブール式の意味は、**WD** インデックスと **TEXT** インデックスで異なります。これは、**TEXT** インデックスの処理では削除された単

語による影響が生じますが、**WD** インデックスではこれに相当する影響が生じないためです。

TEXT インデックスの作成 (Sybase Central)

全文検索を実行するには、検索するカラムに **TEXT** インデックスを作成する必要があります。

TEXT インデックスには、インデックス・カラム内の単語の位置情報が格納されます。

1. DBA または RESOURCE 権限のあるユーザとしてデータベースに接続します。
2. 左側のウィンドウ枠で、[Text Indexes] フォルダを右クリックし、[Text Index] > [新規作成] を選択します。
3. **TEXT** インデックスを作成するテーブルを選択します。
4. **TEXT** インデックスの名前を入力します。[次へ] をクリックします。
5. インデックスに含めるカラムを選択します。[次へ] をクリックします。
6. **TEXT** インデックスのデータを処理するときに使用するテキスト設定オブジェクトを選択します。[次へ] をクリックします。
7. SQL Anywhere テーブルの場合は、[Specify a Refresh Type] ダイアログで [次へ] をクリックします。

   Sybase IQ テーブルの場合は、このオプションは表示されません。サポートされる更新のタイプは [即時] のみです。
8. **TEXT** インデックスを格納する DB 領域を選択します。
9. [次へ] をクリックします。
10. テキスト設定について説明するコメントを入力し、[完了] をクリックします。

TEXT インデックスの作成 (Interactive SQL)

全文検索を実行するには、検索するカラムに **TEXT** インデックスを作成する必要があります。

TEXT インデックスには、インデックス・カラム内の単語の位置情報が格納されます。

1. DBA または RESOURCE 権限のあるユーザとしてデータベースに接続します。
2. **CREATE TEXT INDEX** 文を実行します。

次の例は、**TEXT** インデックス `myTxtIdx` を、`iqdemo` データベースの `Customers` テーブルの `CompanyName` カラムに作成します。`default_char` テキスト設定オブジェクトが使用されます。

```
CREATE TEXT INDEX myTxtIdx ON Customers
    ( CompanyName ) CONFIGURATION default_char
```

TEXT インデックスのサイズの見積もりに関するガイドライン

次の式を使用して、**TEXT** インデックスのメイン・ストアのサイズを見積もります。

*予想されるインデックスのバイト数 = (15+L)*U + U*PAGESIZE * R + T*

各オブジェクトの意味は、次のとおりです。

- L = 語彙における単語の平均長
- U = 語彙におけるユニークな単語の数
- R = ドキュメントの数
- T = すべてのドキュメントに含まれるすべての単語の合計数

TEXT インデックス用に必要な一時領域 (バイト単位) は、 *(M+20)* T* です。

- M = テキスト設定のための単語の最大長 (バイト単位)

注意： 必要な一時領域は、ソート・データの圧縮率によって異なります。

TEXT インデックスの制限

Sybase IQ テキスト設定オブジェクトと **TEXT** インデックスには、設計上の制限があります。

- Sybase IQ エンジンは、複数のカラムにまたがる **TEXT** インデックスをサポートしていません。
- **TEXT** インデックスの手動更新オプションや自動更新オプションはサポートされていません。
- **sp_iqrebuildindex** を使用して **TEXT** インデックスを構築できません。
- **BEGIN PARALLEL IQ...END PARALLEL IQ** 内で **TEXT** インデックスは作成できません。
- **NGRAM** 単語分割は **TEXT** インデックスに構築されます。そのため、**NGRAM** インデックスまたは **GENERICTEXT** インデックスを使用するかどうかを定義するには、テキスト設定オブジェクトの設定を使用します。
- **NGRAMTEXT** インデックス検索は、主に、単語に入力ミスが多い場合に役立ちます。Sybase IQ では、同意語や反意語のような検索はサポートされていません。

TEXT インデックスのリストの表示 (Sybase Central)

データベースのすべての **TEXT** インデックスのリストを表示します。

1. DBA または RESOURCE 権限のあるユーザとしてデータベースに接続します。
2. 左側のウィンドウ枠で、[Text Indexes] フォルダを選択します。

   すべての **TEXT** インデックスのリストが右側のウィンドウ枠に表示されます。

TEXT インデックスのリストの表示 (Interactive SQL)

データベースのすべての **TEXT** インデックスのリストを表示します。

1. DBA または RESOURCE 権限のあるユーザとしてデータベースに接続します。
2. **SELECT** 文を実行します。

すべての Sybase IQ **TEXT** インデックスをリストするには、次の構文を使用します。

```
SELECT * FROM sp_iqindex() WHERE index_type = 'TEXT';
```

カタログ・テーブルの TEXT インデックスを含むすべての **TEXT** インデックスを表示するには、次の構文を使用します。

```
SELECT index_name, table_name, name FROM SYSIDX, SYSTEXTIDX,
SYSTABLE, SYSUSERS
WHERE SYSIDX.object_id=SYSTEXTIDX.index_id
AND SYSIDX.table_id=SYSTABLE.table_id
AND SYSTABLE.creator=SYSUSERS.uid;
```

TEXT インデックスの編集 (Sybase Central)

DB 領域や **TEXT** インデックスの名前などの、**TEXT** インデックスの設定を変更します。

1. DBA または RESOURCE 権限のあるユーザとしてデータベースに接続します。
2. 左側のウィンドウ枠で、[Text Indexes] フォルダを選択します。
3. [Text Indexes] のリストで、変更するオブジェクトを右クリックし、[プロパティ] を選択します。
4. [一般] タブで、必要に応じて設定を変更します。
5. [OK] をクリックします。

TEXT インデックスの編集 (Interactive SQL)

DB 領域や **TEXT** インデックスの名前などの、**TEXT** インデックスの設定を変更します。

1. DBA または RESOURCE 権限のあるユーザとしてデータベースに接続します。
2. **ALTER TEXT INDEX** 文を実行します。

TEXT インデックス `myTxtIdx` の名前を `MyTextIndex` に変更するには、次の構文を使用します。

```
ALTER TEXT INDEX MyTxtIdx
    ON Customers
RENAME AS MyTextIndex;
```

TEXT インデックスのロケーションの変更 (Sybase Central)

TEXT インデックスが格納される DB 領域を変更します。

1. DBA または SPACE ADMIN 権限のあるユーザとして、または DB 領域に対する CREATE 権限のあるテーブル所有者としてデータベースに接続します。
2. 左側のウィンドウ枠で、[Text Indexes] フォルダを選択します。
3. [Text Indexes] のリストで、変更するオブジェクトを右クリックし、[プロパティ] を選択します。
4. [一般] タブで、ドロップダウン・リストから DB 領域を選択します。
5. DB 領域が更新されたら、[OK] をクリックします。

TEXT インデックスのロケーションの変更 (Interactive SQL)

TEXT インデックスが格納される DB 領域を変更します。

1. DBA または SPACE ADMIN 権限のあるユーザとしてデータベースに接続します。
2. **MOVE TO** 句を指定して **ALTER TEXT INDEX** 文を実行します。

TEXT インデックス `MyTextIndex` を `tispace` という名前の DB 領域に移動するには、次の構文を使用します。

```
ALTER TEXT INDEX MyTextIndex ON
    GROUPO.customers MOVE TO tispace;
```

TEXT インデックスの削除 (Sybase Central)

データベースから **TEXT** インデックスを削除します。

1. DBA または RESOURCE 権限のあるユーザとしてデータベースに接続します。
2. 左側のウィンドウ枠で、[Text Indexes] フォルダを選択します。
3. [Text Indexes] のリストで、変更するオブジェクトを右クリックし、[削除] を選択します。
4. 確認ダイアログで [はい] をクリックします。

TEXT インデックスの削除 (Interactive SQL)

データベースから **TEXT** インデックスを削除します。

1. DBA または RESOURCE 権限のあるユーザとしてデータベースに接続します。
2. **DROP TEXT INDEX** 文を実行します。

TEXT インデックス `MyTextIndex` を削除するには、次の構文を使用します。

```
DROP TEXT INDEX MyTextIndex ON Customers;
```

TEXT インデックスの更新

Sybase IQ テーブルの **TEXT** インデックスでサポートされる更新のタイプは即時更新のみです。即時更新は、基になるテーブルのデータが変更されると実行されます。

Sybase IQ テーブルの **TEXT** インデックスの即時更新は、独立性レベル 3 をサポートしています。インデックスは作成時に値が設定されます。また、**INSERT** 文、**UPDATE** 文、または **DELETE** 文を使用してカラムのデータが変更されるたびに値が設定されます。初期更新の間、テーブルに排他ロックが保持されます。

TEXT_DELETE_METHOD データベース・オプション

TEXT インデックスの削除処理で使用されるアルゴリズムを指定します。

指定できる値
0 – 2

0 – 削除方法はコスト・モデルにより選択される。

1 – スモール・デリートが強制される。削除されるローの数が、テーブルの全ロー数に比べて非常に少ないときは、スモール・デリートが便利。スモール・デリートはインデックスにランダムにアクセス可能、大きいデータ・セットでキャッシュがスラッシングされる。

2 – ラージ・デリートが強制される。このアルゴリズムは、削除するロー検索のため全インデックスをスキャンする。削除されるローの数が、テーブルの全ロー数に比べてかなり多いときは、ラージ・デリートが便利。

デフォルト値
0

スコープ
このオプションを設定するために、DBA パーミッションは必要ありません。個々の接続または PUBLIC グループに temporary レベルで設定できます。すぐに有効になります。

説明
TEXT_DELETE_METHOD は、**TEXT** インデックスの削除処理で使用されるアルゴリズムを指定します。このオプションを設定しないか 0 に設定した場合、削除方法はコスト・モデルにより選択されます。コスト・モデルは、適切な削除アルゴリズムを選択する際に、CPU 関連のコストと I/O 関連のコストを考慮します。コスト・モデルでは以下の要素が考慮されます。

- 削除されたロー
- インデックス・サイズ
- インデックス・データ型の幅
- インデックス・データのカーディナリティ
- 利用可能なテンポラリ・キャッシュ
- マシンに関連する I/O と CPU の特性
- 利用可能な CPU とスレッド

『パフォーマンス＆チューニング・ガイド』の「クエリと削除の最適化」>「削除オペレーションの最適化」を参照してください。

例
次の文では、**TEXT** インデックスからのラージ・デリートが強制されます。

```
SET TEMPORARY OPTION TEXT_DELETE_METHOD = 2
```

NGRAM TEXT インデックス

NGRAM TEXT インデックスには、カラムのテキストをテキスト値 N の n-gram に分割したテキストが格納されています。ここで、N はユーザ指定の値です。

クエリの **CONTAINS** 句で指定したテキスト値の n-gram をインデックスに格納されている n-gram に一致させることによって、**NGRAMTEXT** インデックス上で検索を実行できます。

NGRAMTEXT インデックスは、ヨーロッパ言語と非ヨーロッパ言語の両方のテキストに対するファジー検索機能を備えています。ファジー検索の詳細については、「非構造化データのクエリ」>「NGRAM TEXT インデックス検索」>「ファジー検索」を参照してください。

注意： **NGRAMTEXT** インデックス検索は、主に、単語に入力ミスが多い場合に役立ちます。Sybase IQ では、同意語や反意語のような検索はサポートされていません。

NGRAM 単語分割は **TEXT** インデックスに構築されます。そのため、**NGRAM** インデックスまたは **GENERICTEXT** インデックスを使用するかどうかを定義するには、テキスト設定オブジェクトの設定を使用します。

テキスト設定オブジェクトの設定の詳細については、『SQL Anywhere サーバ - SQL の使用法』>「データのクエリと変更」>「全文検索」>「テキスト設定オブジェクトの管理方法」>「テキスト設定オブジェクトの設定」を参照してください。

注意： このリファレンスは SQL Anywhere マニュアルにリンクされています。

NGRAM TEXT インデックスの作成

NGRAM TEXT インデックスの作成に関する参照情報を示します。

NGRAMTEXT インデックスを作成する方法については、『SQL Anywhere サーバ - SQL の使用法』>「データのクエリと変更」>「全文検索」>「全文検索を実行する方法」>「チュートリアル： NGRAM テキスト・インデックスへの全文検索の実行」および『SQL Anywhere サーバ - SQL の使用法』>「データのクエリと変更」>「全文検索」>「全文検索を実行する方法」>「チュートリアル： あいまい全文検索の実行」を参照してください。

注意： これらのリファレンスは SQL Anywhere マニュアルにリンクされています。

テキスト設定オブジェクト

テキスト設定オブジェクトによって、**TEXT** インデックスの構築または更新時にインデックスに配置される単語と、全文クエリの解釈方法が制御されます。

データベース・サーバは、**TEXT** インデックスの作成時または更新時に、**TEXT** インデックスが作成されたときに指定されたテキスト設定オブジェクトの設定を使用します。テキスト設定オブジェクトが指定されていない場合、データベース・サーバは、インデックスが作成されるカラムのデータ型に基づいて、デフォルトのテキスト設定オブジェクトを選択します。Sybase IQ データベースでは、`default_char` テキスト設定オブジェクトが常に使用されます。

テキスト設定オブジェクトによって、インデックスを作成するドキュメントから単語を生成するために使用されるプレフィルタ・ライブラリと単語分割ライブラリが指定されます。テキスト設定オブジェクトでは、**TEXT** インデックス内に格納する単語の最小長と最大長、除外する単語のリストを指定します。テキスト設定オブジェクトは次のパラメータで構成されます。

- ドキュメント・プレフィルタ – フォーマットやイメージなどの不要な情報をドキュメントから削除する。その後、フィルタされたドキュメントが他のモジュールによって選択され、さらなる処理が行われる。ドキュメント・プレフィルタはサード・パーティ・ベンダによって提供される。
- ドキュメント単語分割 – 受信バイト・ストリームを、単語セパレータによって区切られた、または指定された規則に従って区切られた単語に分割する。ドキュメント単語分割は、サーバまたはサード・パーティ・ベンダによって提供される。
- ストップリスト・プロセッサ – **TEXT** インデックスの構築中に無視する単語のリストを指定する。

デフォルトのテキスト設定オブジェクト

Sybase IQ には、デフォルトのテキスト設定オブジェクトが用意されています。

デフォルトのテキスト設定オブジェクト `default_char` は、非 `NCHAR` データとともに使用されます。この設定は、テキスト設定オブジェクトまたは **TEXT** インデックスを初めて作成するときに作成されます。

テキスト設定オブジェクト `default_nchar` では、**IN SYSTEM** テーブルの **TEXT** インデックスの `NCHAR` での使用がサポートされています。Sybase IQ テーブルの **TEXT** インデックスで `default_nchar` テキスト設定は使用できません。

表「デフォルトのテキスト設定オブジェクトの設定」は、ほとんどの文字ベースの言語に最適な、`default_char` と `default_nchar` のデフォルト設定を示します。デフォルトのテキスト設定オブジェクトの設定を変更しないことを強くおすすめします。

表 2 : デフォルトのテキスト設定オブジェクトの設定

設定値	インストールされている値
TERM BREAKER	GENERIC
MINIMUM TERM LENGTH	1
MAXIMUM TERM LENGTH	20
STOPLIST	(空)

デフォルトのテキスト設定オブジェクトを削除した場合、次に **TEXT** インデックスまたはテキスト設定オブジェクトを作成したときに、自動的にデフォルト値で再作成されます。

テキスト設定の作成 (Sybase Central)

テキスト設定を作成し、テキスト設定プロセスに依存する **TEXT** インデックスがデータ内の単語を処理する方法を指定します。

1. DBA または RESOURCE 権限のあるユーザとしてデータベースに接続します。
2. 左側のウィンドウ枠で、[Text Configurations Objects] フォルダを右クリックし、[新規作成] > [Text Configurations Object] を選択します。
3. テキスト設定の名前を入力します。
4. テキスト設定の所有者を選択します。
5. テキスト設定のデータベース照合のタイプを選択します。[次へ] をクリックします。

   注意： NCHAR 照合を使用するテキスト設定は、Sybase IQ **TEXT** インデックスではサポートされません。
6. [GENERIC 単語分割] アルゴリズムを選択します。
7. 単語の最小長と最大長を入力します。
8. 外部の単語分割ライブラリを使用する場合は、[Use an external term breaker] を選択して、外部の単語分割関数およびライブラリを指定します。

   関数とライブラリを、`function-name@library-file-name` の形式で指定します。
9. [次へ] をクリックします。
10. 外部のプレフィルタ・ライブラリを使用する場合は、[Use an external prefilter] を選択して、外部のプレフィルタ関数およびライブラリを指定します。

    関数とライブラリを、`function-name@library-file-name` の形式で指定します。
11. このテキスト設定を使用して **TEXT** インデックスを構築するときに無視する単語をストップリストに追加します。単語をスペースで区切ります。

    このリストに含まれている単語は、クエリでも無視されます。
12. [次へ] をクリックします。
13. テキスト設定について説明するコメントを入力し、[完了] をクリックします。

テキスト設定の作成 (Interactive SQL)

テキスト設定を作成し、テキスト設定プロセスに依存する **TEXT** インデックスがデータ内の単語を処理する方法を指定します。

1. DBA または RESOURCE 権限のあるユーザとしてデータベースに接続します。
2. **CREATE TEXT CONFIGURATION** 文を実行します。

`default_char` テキスト設定オブジェクトをテンプレートとして使用して、`myTxtConfig` という名前のテキスト設定オブジェクトを作成するには、次の構文を使用します。

```
CREATE TEXT CONFIGURATION myTxtConfig FROM default_char;
```

テキスト設定オブジェクトの設定

テキスト設定オブジェクトの設定と、それらがインデックスの作成対象にどのような影響を与えるか、また全文検索クエリがどのように解釈されるかについて説明します。

テキスト設定オブジェクトと、それらが **TEXT** インデックスと全文検索に与える影響の例については、「テキスト設定オブジェクトの設定の解釈」を参照してください。

参照：

- テキスト設定オブジェクトの設定の解釈 (20 ページ)

単語分割アルゴリズム (TERM BREAKER)

TERM BREAKER 設定は、文字列を単語に分割するために使用されるアルゴリズムを指定します。

Sybase IQ では、単語の格納に関して **GENERIC** (デフォルト) と **NGRAM** がサポートされています。

注意： **NGRAM** 単語分割には、n-gram が格納されます。n-gram は、長さ *n* の文字のグループです。*n* は、**MAXIMUM TERM LENGTH** の値です。

指定する単語分割にかかわらず、データベース・サーバは、単語が **TEXT** インデックスに挿入されるときに、単語の元の位置情報を **TEXT** インデックスに記録します。n-gram の場合は、元の単語の位置情報ではなく、n-gram の位置情報が格納されます。

表 3 : TERM BREAKER の影響

TEXT インデックスに対して	クエリ単語に対して
GENERIC TEXT インデックス – **GENERICTEXT** インデックス (デフォルト) を構築する場合、英数字以外の文字の間の一連の英数字は、データベース・サーバによって単語として処理されます。単語の定義後に、単語の長さの設定を超える単語と、ストップリストに含まれている単語は、カウントはされますが、**TEXT** インデックスには挿入されません。 **GENERICTEXT** インデックスのパフォーマンスは、**NGRAMTEXT** インデックスより高速です。ただし、**GENERICTEXT** インデックスではファジー検索は実行できません。	GENERIC TEXT インデックス – **GENERICTEXT** インデックスに対してクエリする場合、クエリ文字列内の単語は、インデックスが作成される場合と同じように処理されます。クエリ単語と **TEXT** インデックスに含まれる単語を比較して照合が実行されます。
NGRAM TEXT インデックス – **NGRAMTEXT** インデックスを構築する場合、英数字以外の文字の間の一連の英数字は、データベース・サーバによって 1 つの単語として処理されます。単語が定義されると、データベース・サーバが単語を n-gram に分割します。こうすることで、n よりも短い単語と、ストップリストに含まれている n-gram は破棄されます。 たとえば、**MAXIMUM TERM LENGTH** が 3 の **NGRAMTEXT** インデックスの場合、文字列 'my red table' は、**TEXT** インデックスで red tab abl ble の n-gram として表されます。	NGRAM TEXT インデックス – **NGRAMTEXT** インデックスに対してクエリする場合、クエリ文字列内の単語は、インデックスが作成される場合と同じように処理されます。クエリ単語の n-gram とインデックスが付けられた単語の n-gram を比較して照合が実行されます。

単語の最小長の設定 (MINIMUM TERM LENGTH)

MINIMUM TERM LENGTH 設定で、インデックスに挿入される、または全文クエリで検索される単語の最小長 (文字数) を指定します。

MINIMUM TERM LENGTH は、**NGRAMTEXT** インデックスには関係しません。

MINIMUM TERM LENGTH は、特にプレフィクス検索に関係します。**MINIMUM TERM LENGTH** の値は、0 よりも大きくする必要があります。**MAXIMUM TERM LENGTH** よりも大きい値に設定すると、**MAXIMUM TERM LENGTH** は **MINIMUM TERM LENGTH** と等しい値に自動的に調整されます。

MINIMUM TERM LENGTH のデフォルト値は、デフォルトのテキスト設定オブジェクトの設定から取得されます。通常は 1 です。

表 4 : MINIMUM TERM LENGTH の影響

TEXT インデックスに対して	クエリ単語に対して
GENERIC TEXT インデックス – **GENERICTEXT** インデックスの場合、**TEXT** インデックスには、**MINIMUM TERM LENGTH** よりも短い単語は格納されません。	GENERIC TEXT インデックス – **GENERICTEXT** インデックスに対してクエリする場合、**MINIMUM TERM LENGTH** よりも短いクエリ単語は、**TEXT** インデックスに存在している可能性がないため無視されます。
NGRAM TEXT インデックス – **NGRAMTEXT** インデックスの場合、この設定は無視されます。	NGRAM TEXT インデックス – **MINIMUM TERM LENGTH** 設定は、**NGRAMTEXT** インデックスに対する全文クエリには影響しません。

単語の最大長の設定 (MAXIMUM TERM LENGTH)

MAXIMUM TERM LENGTH 設定で、インデックスに挿入される、または全文クエリで検索される単語の最大長 (文字数) を指定します。

MAXIMUM TERM LENGTH 設定は、単語分割アルゴリズムに応じて異なります。**MAXIMUM TERM LENGTH** の値は、60 以下にする必要があります。**MAXIMUM TERM LENGTH** を **MINIMUM TERM LENGTH** よりも小さい値に設定すると、**MINIMUM TERM LENGTH** は **MAXIMUM TERM LENGTH** と等しい値に自動的に調整されます。

この設定のデフォルト値は、デフォルトのテキスト設定オブジェクトの設定から取得されます。通常は 20 です。

表 5 : MAXIMUM TERM LENGTH の影響

TEXT インデックスに対して	クエリ単語に対して
GENERIC TEXT インデックス – **GENERICTEXT** インデックスの場合、**MAXIMUM TERM LENGTH** は、**TEXT** インデックスに挿入される単語の最大長 (文字数) を指定します。	GENERIC TEXT インデックス – **GENERICTEXT** インデックスの場合、**MAXIMUM TERM LENGTH** よりも長いクエリ単語は、**TEXT** インデックスに存在している可能性がないため無視されます。
NGRAM TEXT インデックス – **NGRAMTEXT** インデックスの場合、**MAXIMUM TERM LENGTH** によって、単語が分割される n-gram の長さが決まります。**MAXIMUM TERM LENGTH** の適切な長さは、言語によって異なります。一般的な値は、英語の場合は 4 または 5 文字、中国語の場合は 2 または 3 文字です。	NGRAM TEXT インデックス – **NGRAMTEXT** インデックスの場合、クエリ単語は長さ n の n-gram に分割されます。n は **MAXIMUM TERM LENGTH** と同じです。データベース・サーバは、n-gram を使用して **TEXT** インデックスを検索します。**MAXIMUM TERM LENGTH** よりも短い単語は **TEXT** インデックス内の n-gram と一致しないため、無視されます。

ストップリストの設定 (STOPLIST)

ストップリストの設定では、インデックスを作成しない単語を指定します。

ストップリスト設定のデフォルト値は、デフォルトのテキスト設定オブジェクトの設定から取得されます。通常は、ストップリストは空です。

表 6 : STOPLIST の影響

TEXT インデックスに対して	クエリ単語に対して
GENERIC TEXT インデックス – **GENERICTEXT** インデックスの場合、ストップリストに含まれる単語は **TEXT** インデックスに挿入されません。	GENERIC TEXT インデックス – **GENERICTEXT** インデックスの場合、ストップリストに含まれるクエリ単語は、**TEXT** インデックスに存在している可能性がないため無視されます。
NGRAM TEXT インデックス – **NGRAMTEXT** インデックスの場合、**TEXT** インデックスには、ストップリストに含まれている単語から形成された n-gram は格納されません。	NGRAM TEXT インデックス – ストップリストに含まれる単語は n-gram に分割され、n-gram がストップリスト用に使用されます。同様に、クエリ単語は n-gram に分割され、ストップリストに含まれる n-gram と一致する n-gram は、**TEXT** インデックスに存在している可能性がないため削除されます。

単語をストップリストに含めるかどうかは、慎重に検討してください。特に、アポストロフィやダッシュなど、英数字以外の文字を含む単語は含めないでください。これらの文字は、単語の分割記号として機能します。たとえば、you'll という単語 ('you'll' と指定する必要があります) は、you と ll に分割され、2 つの単語としてストップリストに格納されます。以降の 'you' または 'they'll' の全文検索が悪影響を受けます。

NGRAMTEXT インデックスのストップリストによって、予期しない結果が生じる場合があります。これは、格納されるストップリストが、実際には n-gram 形式であり、指定したストップリストの単語ではないためです。たとえば、**MAXIMUM TERM LENGTH** が 3 の **NGRAMTEXT** インデックスの場合、**STOPLIST** 'there' を指定すると、 the her ere の n-gram がストップリストとして格納されます。これは、the、her、ere という n-gram を含む単語をクエリする能力に影響します。

テキスト設定のリストの表示 (Sybase Central)

データベースのすべてのテキスト設定のリストを表示します。

1. DBA または RESOURCE 権限のあるユーザとしてデータベースに接続します。
2. 左側のウィンドウ枠で、[Text Configurations Objects] を選択します。

   すべてのテキスト設定のリストが右側のウィンドウ枠に表示されます。

テキスト設定のリストの表示 (Interactive SQL)

データベースのすべてのテキスト設定のリストを表示します。

1. DBA または RESOURCE 権限のあるユーザとしてデータベースに接続します。
2. **SELECT** 文を実行します。

すべてのテキスト設定オブジェクトをリストするには、次の構文を使用します。

```
SELECT * FROM SYSTEXTCONFIG;
```

テキスト設定の変更 (Sybase Central)

DB 領域や、単語で許可する長さの範囲などの、テキスト設定オブジェクトの設定を変更します。

TEXT インデックスで使用されていないテキスト設定オブジェクトのみを変更できます。

1. DBA または RESOURCE 権限のあるユーザとしてデータベースに接続します。
2. 左側のウィンドウ枠で、[Text Configurations Objects] を選択します。
3. [Text Configurations] のリストで、変更するオブジェクトを右クリックし、[プロパティ] を選択します。
4. [設定] タブに切り替え、必要に応じて設定を変更します。
5. [OK] をクリックします。

テキスト設定の変更 (Interactive SQL)

DB 領域や、単語で許可する長さの範囲などの、テキスト設定オブジェクトの設定を変更します。

TEXT インデックスで使用されていないテキスト設定オブジェクトのみを変更できます。

1. DBA または RESOURCE 権限のあるユーザとして、またはテキスト設定オブジェクトの所有者としてデータベースに接続します。
2. **ALTER TEXT CONFIGURATION** 文を実行します。

`myTxtConfig` テキスト設定オブジェクトの単語の最小長を変更するには、次の構文を使用します。

```
ALTER TEXT CONFIGURATION myTxtConfig
    MINIMUM TERM LENGTH 2;
```

ストップリストの変更 (Sybase Central)

このテキスト設定を使用して **TEXT** インデックスを構築するときに無視する単語のリストが格納されているストップリストを変更します。

TEXT インデックスで使用されていないテキスト設定オブジェクトのみを変更できます。

1. DBA または RESOURCE 権限のあるユーザとしてデータベースに接続します。
2. 左側のウィンドウ枠で、[Text Configurations Objects] を選択します。
3. [Text Configurations] のリストで、変更するオブジェクトを右クリックし、[プロパティ] を選択します。
4. [Stoplist] タブに切り替えて、必要に応じてストップリストの単語を変更します。スペースを使用して単語を区切ります。
5. ストップリストの単語のリストをアルファベット順にソートし、それらをリストに表示するには、[Sort Terms] をクリックします。
6. ストップリストが更新されたら、[OK] をクリックします。

ストップリストの変更 (Interactive SQL)

このテキスト設定を使用して **TEXT** インデックスを構築するときに無視する単語のリストが格納されているストップリストを変更します。

TEXT インデックスで使用されていないテキスト設定オブジェクトのみを変更できます。

1. DBA または RESOURCE 権限のあるユーザとしてデータベースに接続します。
2. **STOPLIST** 句を指定して **ALTER TEXT CONFIGURATION** 文を実行します。

ストップリストを `myTxtConfig` 設定オブジェクトに追加するには、次の構文を使用します。

```
ALTER TEXT CONFIGURATION myTxtConfig
    STOPLIST 'because about therefore only';
```

テキスト設定の削除 (Sybase Central)

不要なテキスト設定をデータベースから削除します。

TEXT インデックスで使用されていないテキスト設定のみを削除できます。

1. DBA または RESOURCE 権限のあるユーザとしてデータベースに接続します。
2. 左側のウィンドウ枠で、[Text Configurations Objects] を選択します。

3. [Text Configurations] のリストで、変更するオブジェクトを右クリックし、[削除] を選択します。
4. 確認ダイアログで [はい] をクリックします。

テキスト設定の削除 (Interactive SQL)

不要なテキスト設定をデータベースから削除します。

TEXT インデックスで使用されていないテキスト設定のみを削除できます。

1. DBA または RESOURCE 権限のあるユーザとしてデータベースに接続します。
2. **DROP TEXT CONFIGURATION** 文を実行します。

テキスト設定オブジェクト `myTxtConfig` を削除するには、次の構文を使用します。

```
DROP TEXT CONFIGURATION myTxtConfig;
```

テキスト設定オブジェクトの例

サンプルを確認して、テキスト設定オブジェクトの設定が **TEXT** インデックスにどのように影響するか、またインデックスがどのように解釈されるかを理解します。

テキスト設定オブジェクトの設定の解釈

以下の例では、さまざまなテキスト設定オブジェクトの設定と、それらの設定がインデックスの作成対象にどのような影響を与えるか、全文検索クエリ文字列がどのように解釈されるかを示します。

すべての例で、文字列 'I'm not sure I understand' を使用しています。

表 7 : テキスト設定オブジェクトの設定の解釈

設定	インデックスが作成される単語	クエリの解釈
TERM BREAKER：GENERIC MINIMUM TERM LENGTH： 1 MAXIMUM TERM LENGTH： 20 STOPLIST： ''	`I m not sure I understand`	`'("I m" AND not sure) AND I AND understand'`
TERM BREAKER：GENERIC MINIMUM TERM LENGTH： 2 MAXIMUM TERM LENGTH： 20 STOPLIST：'not and'	`sure understand`	`'understand'`

設定	インデックスが作成される単語	クエリの解釈
TERM BREAKER：GENERIC MINIMUM TERM LENGTH： 1 MAXIMUM TERM LENGTH： 20 STOPLIST：'not and'	`I m sure I understand`	`'"I m" AND sure AND I AND understand'`

テキスト設定オブジェクトの CONTAINS クエリ文字列の解釈

以下の例では、テキスト設定オブジェクトの文字列の設定が CONTAINS クエリでどのように解釈されるかを示します。

表「CONTAINS 文字列の解釈」の「文字列の解釈」列のカッコで囲まれた数値は、単語ごとに格納される位置情報を示しています。数値は、マニュアルで説明するためのものです。格納される実際の単語には、カッコで囲まれた数値は含まれません。

注意： テキスト・ドキュメントの位置情報の最大数は 4294967295 です。

この表に示すのは、**CONTAINS** クエリの解釈のみです。データが解析されるときに、**AND**、**NOT**、**NEAR** は、通常のトークンと見なされます。また、*****、**|** などの記号は、英数字ではないため、削除されます。

表 8 : CONTAINS 文字列の解釈

設定	文字列	文字列の解釈
TERM BREAKER：GENERIC MINIMUM TERM LENGTH： 3 MAXIMUM TERM LENGTH： 20	`'w*'`	`'"w*(1)"'`
	`'we*'`	`'"we*(1)"'`
	`'wea*'`	`'"wea*(1)"'`
	`'we* -the'`	`'"we*(1)" -"the(1)"'`
	`'for* | wonderl*'`	`'"for*(1)" | "wonderl*(1)"'`
	`'wonderlandwonderlandwonderland*'`	`''`
	`'"tr* weather"'`	`'"weather(1)"'`
	`'"tr* the weather"'`	`'"the(1) weather(2)"'`

設定	文字列	文字列の解釈
	'"wonderlandwonderlandwonderland* wonderland"'	'"wonderland(1)"'
	'"wonderlandwonderlandwonderland* weather"'	'"weather(1)"'
	'"the_wonderland-wonderlandwonderland* weather"'	'"the(1) weather(3)"'
	'the_wonderland-wonderlandwonderland* weather'	'"the(1)" & "weather(1)"'
	'"light_a* the end" & tunnel'	'"light(1) the(3) end(4)" & "tunnel(1)"'
	light_b* the end" & tunnel'	'"light(1) the(3) end(4)" & "tunnel(1)"'
	'"light_at_b* end"'	'"light(1) end(4)"'
	'and-te*'	'"and(1) te*(2)"'
	'a_long_and_t* & journey'	'"long(2) and(3) t*(4)" & "journey(1)"'

MAX_PREFIX_PER_CONTAINS_PHRASE データベース・オプション

テキスト検索式で許可するプレフィクス単語の数を指定します。

指定できる値
0 – 300

0 – 検索フレーズでプレフィクス単語を制限しない。

300 – 上限 (これは、フレーズで許可する単語の合計数の総合限度)

デフォルト値
1

スコープ
このオプションを設定するために、DBA パーミッションは必要ありません。個々の接続に temporary レベルで設定できます。また、PUBLIC グループに設定できます。すぐに有効になります。

説明
MAX_PREFIX_PER_CONTAINS_PHRASE では、テキスト検索式での一定数以上のプレフィクスを禁止するためのしきい値を指定します。

このオプションを 0 に設定すると、数の制限はなくなります。Sybase IQ は、クエリの **CONTAINS** 式に、このオプションで指定した数よりも多くのプレフィックス単語を含む語句がないかどうかを確認し、見つかった場合にはエラーを報告します。

例
デフォルトの MAX_PREFIX_PER_CONTAINS_PHRASE 設定を使用する例：

```
SET MAX_PREFIX_PER_CONTAINS_PHRASE = 1
```

次の **CONTAINS** 句は有効です。

```
SELECT ch1 FROM tab1
WHERE CONTAINS(ch1, '"concord bed* in mass"')
```

デフォルトの MAX_PREFIX_PER_CONTAINS_PHRASE 設定の 1 を使用した場合、次の **CONTAINS** 句は構文エラーを返します。

```
SELECT ch1 FROM tab1
WHERE CONTAINS (ch1, '"con* bed* in mass"')
```

0 (制限なし) または 2 に MAX_PREFIX_PER_CONTAINS_PHRASE を設定した場合は、上記の **CONTAINS** 句は有効です。

外部ライブラリ

外部ライブラリを使用して、ドキュメントでプレフィルタと単語分割を提供する方法について説明します。

プレフィルタと単語分割の外部ライブラリ

Sybase IQ では、C または C++ で記述された外部のプレフィルタ・ライブラリと単語分割ライブラリを使用して、インデックスの作成中またはクエリの処理中にドキュメントをプレフィルタしてトークン化できます。これらのライブラリは、データベース・サーバのプロセス領域に動的にロードできます。

注意： 外部のプレフィルタ・ライブラリと単語分割ライブラリは、Sybase 認定パートナーから提供を受ける必要があります。認定ベンダ・ソリューションについては、Partner Certification Reports Web サイトにアクセスし、認定レポートをフィルタ処理して、Sybase IQ に関する認定を表示してください。

外部の動的にロード可能なプレフィルタ・ライブラリおよび単語分割ライブラリは、テキスト設定で指定します。また、これらのライブラリは、データベース・サーバによってロードされる必要があります。各ライブラリには、テキスト設定オブジェクトで指定されている外部関数を実装するエクスポートされた記号が含まれています。この関数は、呼び出し元が必要なタスクを実行するために使用する一連の関数記述子を返します。

外部のプレフィルタ・ライブラリと単語分割ライブラリは、特定のカラムに対する、ライブラリのロードを必要とするクエリが受信されたとき、または **TEXT** インデックスを更新する必要があるときに、最初の **CREATE TEXT INDEX** 要求で、データベース・サーバによってロードされます。

ライブラリは、**ALTER TEXT CONFIGURATION** 呼び出しが実行されてもロードされません。また、**DROP TEXT CONFIGURATION** 呼び出しが実行されても、自動的にアンロードされません。サーバが、外部ライブラリのロードを禁止するオプションを使用して起動された場合は、外部のプレフィルタ・ライブラリおよび単語分割ライブラリはロードされません。

これらの外部 C/C++ ライブラリは、サーバのプロセス領域への非サーバ・ライブラリ・コードのロードを行うので、関数の記述が不完全な場合や意図的に不正な場合、データの整合性やセキュリティ、およびサーバの堅牢性に関してリスクが発生する可能性があります。これらのリスクを管理するために、Sybase IQ サーバごとに明示的にこの機能を有効または無効にできます。「起動時の外部ライブラリの有効化と無効化」を参照してください。

ISYSTEXTCONFIG システム・テーブルには、テキスト設定オブジェクトに関連する、外部ライブラリについての情報が格納されています。『リファレンス：ビルディング・ブロック、テーブル、およびプロシージャ』の「システム・テーブルとシステム・ビュー」>「システム・ビュー」>「SYSTEXTCONFIG システム・ビュー」を参照してください。

参照：

- 起動時の外部ライブラリの有効化と無効化 (27 ページ)

外部ライブラリの制限

外部ライブラリを使用する Sybase IQ テキスト設定オブジェクトと **TEXT** インデックスには、設計上の制限があります。

- バイナリ・カラムの **TEXT** インデックスの場合、外部ベンダによって提供された外部ライブラリを使用してドキュメント変換を行う必要がある。Sybase IQ は、バイナリ・カラムに格納されているドキュメントを暗黙的に変換しない。
- 外部の単語分割を使用してドキュメントをトークン化する場合、n-gram ベースのテキスト検索はサポートされない。
- 外部ライブラリを使用して SQL Anywhere テーブルに **TEXT** インデックスを作成することはできない。実行した場合、エラーが発生する。

マルチプレックス・サーバでの外部ライブラリ

すべてのマルチプレックス・サーバが、外部のプレフィルタ・ライブラリと単語分割ライブラリにアクセスできる必要があります。

ユーザは、外部の各プレフィルタ・ライブラリと各単語分割ライブラリが、マルチプレックス・サーバをホストしているマシンにコピーされており、しかもサーバによってライブラリをロードできる場所に配置されていることを確認する必要があります。

外部のプレフィルタおよび単語分割の呼び出し時に、各マルチプレックス・サーバは、他のサーバとは独立して動作します。各プロセス領域では、外部ライブラリがロードされ、独自に実行できます。プレフィルタ関数および単語分割関数は、各サーバで同様に実装され、同じ結果が返されることが前提となります。

あるサーバのプロセス領域から外部ライブラリをアンロードしても、他のサーバのプロセス領域からライブラリがアンロードされるわけではありません。

起動時の外部ライブラリの有効化と無効化

Sybase IQ には、サードパーティ製の外部ライブラリのロードを有効または無効にする **-sf** 起動スイッチが用意されています。

このスイッチは、サーバ起動コマンド・ラインまたはサーバ設定ファイルで指定できます。

外部のサードパーティ・ライブラリのロードを有効にする場合、次の構文を使用します。

```
-sf -external_library_full_text
```

外部のサードパーティ・ライブラリのロードを無効にする場合、次の構文を使用します。

```
-sf external_library_full_text
```

現在サーバにロードされているライブラリを一覧表示するには、**sa_list_external_library** ストアド・プロシージャを使用します。

外部ライブラリのアンロード

外部ライブラリが不要になったときにライブラリをアンロードするには、システム・プロシージャ **dbo.sa_external_library_unload** を使用します。

dbo.sa_external_library_unload は、`LONG VARCHAR` 型のオプション・パラメータを 1 つ取ります。このパラメータでは、アンロードするライブラリの名前を指定します。パラメータを指定しない場合、使用されていないすべての外部ライブラリがアンロードされます。

外部ライブラリをアンロードするには、次の構文を使用します。

```
call sa_external_library_unload('library.dll')
```

非構造化データのクエリ

非構造化データと半構造化データを処理する全文検索機能を含む、ラージ・オブジェクト・データのクエリについて説明します。

全文検索

全文検索では、**TEXT** インデックスを使用して、テーブルのローをスキャンせずに、データベース内の単語 (ワード) のすべてのインスタンスをすばやく見つけます。

TEXT インデックスには、インデックス・カラム内の単語の位置情報が格納されます。**TEXT** インデックスを使用して、単語が含まれるローを検索する方法は、テーブル内の各ローをスキャンする方法よりも高速です。

全文検索では、**CONTAINS** 検索条件を使用します。これは、一致がパターンベースではなく単語ベースであるため、**LIKE**、**REGEXP**、**SIMILAR TO** などの述部を使用する検索とは異なります。

全文検索の文字列比較では、データベースのすべての標準照合設定が使用されます。たとえば、大文字と小文字を区別しないようにデータベースを設定すると、全文検索でも大文字と小文字は区別されません。

「CONTAINS 条件」と『リファレンス：ビルディング・ブロック、テーブル、およびプロシージャ』の「SQL 言語の要素」>「検索条件」>「CONTAINS 条件」を参照してください。

参照：

- CONTAINS 条件 (31 ページ)

全文検索のタイプ

全文検索を使用して、単語、プレフィクス、語句 (一連の単語) を検索できます。複数の単語、語句、またはプレフィクスをブール式に組み合わせたり、近接検索を使用して、式が互いに近接していることを求めたりできます。

WHERE 句または **SELECT** 文の **FROM** 句のいずれかで **CONTAINS** 句を使用して、全文検索を実行します。

注意： SQL Anywhere のマニュアルには、全文検索の例が記載されています。これらの例すべてが Sybase IQ に当てはまるわけではありません。たとえば、Sybase IQ では、IF 検索条件の一部となるテキスト検索はサポートされていません。

『SQL Anywhere サーバ - SQL の使用法』>「データのクエリと変更」>「全文検索」を参照してください。

注意： このリファレンスは SQL Anywhere マニュアルにリンクされています。

FROM 句

SELECT 文に必要なデータベース・テーブルまたはビューを指定します。

構文

```
... FROM table-expression [, …]
```

パラメータ

table-expression:{ *table-spec* | *table-expressionjoin-typetable-spec* [**ON***condition*] | (*table-expression* [, …]) }

table-spec:{ [*userid*.] *table-name* [[**AS**] *correlation-name*] | *select-statement* [**AS***correlation-name* (*column-name* [, …])] }

contains-expression:{*table-name* | *view-name* } **CONTAINS** (*column-name* [,...], *contains-query*) [[**AS**] *score-correlation-name*]

使用法

contains-expression – テーブル名の後に **CONTAINS** 句を使用してテーブルをフィルタし、*contains-query* で指定した全文クエリに一致するローのみを返します。

テーブルの一致するすべてのローが、*score-correlation-name* (指定されている場合) を使用して参照できる score カラムとともに返されます。*score-correlation-name* が指定されていない場合は、デフォルトの相関名 *contains* で score カラムを参照できます。

オプションの相関名の引数を例外として、**CONTAINS** 句は、**CONTAINS** 検索条件と同じ引数を取ります。**CONTAINS** 句でリストされているカラムには **TEXT** インデックスが設定されている必要があります。

「CONTAINS 条件」と『リファレンス：ビルディング・ブロック、テーブル、およびプロシージャ』の「SQL 言語の要素」>「検索条件」>「CONTAINS 条件」を参照してください。

FROM 句の完全な構文と説明については、『リファレンス：文とオプション』の「SQL 文」>「FROM 文」を参照してください。

参照：

- CONTAINS 条件 (31 ページ)

CONTAINS 条件

SELECT 文の **FROM** 句で **CONTAINS** 句を使用して、または **WHERE** 句で **CONTAINS** 検索条件 (述部) を使用して、全文クエリを実行します。

どちらの方法でも同じローが返されますが、**CONTAINS** 句では一致するローのスコアも返されます。

構文

```
CONTAINS ( column-name [,...], contains-query-string )

contains-query-string:
    simple-expression
    | or-expression

simple-expression:
    primary-expression
    | and-expression

or-expression:
    simple-expression { OR | | } contains-query-string

primary-expression:
    basic-expression
    | FUZZY " fuzzy-expression "
    | and-not-expression

and-expression:
    primary-expression [ AND | & ] simple-expression

and-not-expression:
    primary-expression [ AND | & ] { NOT | - } basic-expression

basic-expression:
    term
    | phrase
    | ( contains-query-string )
    | proximity-expression

fuzzy-expression:
    term
    | fuzzy-expression term

term:
    simple-term
    | prefix-term

prefix-term:
    simple-term*
```

```
phrase:
    " phrase-string "
```

```
proximity-expression:
    term  (BEFORE|NEAR)[minimum distance  ,|maximum distance ]
 term
    | term {  BEFORE|NEAR | ~ } term
```

```
phrase-string:
    term
    | phrase-string term
```

パラメータ
simple-term – スペースと特殊文字で区切られた文字列。これは、検索対象である、単一の、インデックスが付けられた単語 (ワード) を表します。

distance – 正の整数。

and-expression – *and-expression* を使用して、*primary-expression* と *simple-expression* の両方が、**TEXT** インデックスで見つかる必要があることを指定します。デフォルトでは、単語または式の間に演算子が指定されていない場合、and-expression と見なされます。たとえば、`'a b''` は `'a AND b'` と解釈されます。AND の代わりにアンパサンド (&) を使用して、両側に式または単語を隣接させることができます (`'a & b'` など)。

and-not-expression – *and-not-expression* を使用して、*primary-expression* は **TEXT** インデックスに存在する必要があるが、*basic-expression* は **TEXT** インデックスに存在しない必要があることを指定します。これは、符号反転とも呼ばれます。符号反転にハイフンを使用する場合は、ハイフンの前にスペースを挿入する必要があり、ハイフンがそれに続く単語に隣接している必要があります。たとえば、`'a -b'` は `'a AND NOT b'` と同じですが、`'a - b'` の場合は、ハイフンが無視され、文字列は `'a AND b'` と同じになります。`'a-b'` は語句 `'"a b"'` と同じです。

or-expression – *or-expression* を使用して、少なくとも *simple-expression* または *contains-query-string* のいずれかが **TEXT** インデックスに存在する必要があることを指定します。たとえば、`'a|b'` は `'a OR b'` と解釈されます。

fuzzy-expression – fuzzy-expression を使用して、指定した単語と似た単語を見つけます。ファジー一致は、**NGRAMTEXT** インデックスでのみサポートされています。「ファジー検索」を参照してください。

proximity-expression – *proximity-expression* を使用して、互いに近くにある単語を検索します。たとえば、`'b NEAR[2,5] c'` は、互いに単語 2 ～ 5 個分離れている b と c のインスタンスを検索します。単語の順序は重要ではありません。`'b NEAR c'` は `'c NEAR b'` と同じです。*distance* を指定せずに **NEAR** を指定した場合、デ

フォルトの単語 10 個分が適用されます。**NEAR** の代わりに波型記号 (~) を指定できます。これは、距離を指定せずに **NEAR** を指定して、デフォルトの単語 10 個分が適用されるのと同じです。`'a NEAR[1] b NEAR[1] c'` などのように **NEAR** 式をつなげることはできません。

BEFORE は、単語の順序が重要であることを除き、**NEAR** に似ています。`'b BEFORE c'` と `'c BEFORE b'` は同じではありません。前者の場合、単語 `'b'` が `'c'` より前にある必要があり、後者の場合、単語 `'b'` が `'c'` より後にある必要があります。**BEFORE** は **NEAR** と同様に最小距離と最大距離の両方を受け入れます。デフォルトの最小距離は 1 です。最小距離を指定する場合は、最大距離以下にする必要があります。そうしないと、エラーが返されます。

prefix-term – *prefix-term* を使用して、指定されたプレフィクスで始まる単語を検索します。たとえば、`'datab*'` では、*datab* で始まる単語が検索されます。これは、プレフィクス検索とも呼ばれます。プレフィクス検索では、アスタリスクの左側の部分にあたる単語の一部について照合が行われます。

使用法

CONTAINS 検索条件は、引数としてカラム・リストと *contains-query-string* を取ります。

CONTAINS 検索条件は、検索条件 (述部とも呼ばれる) を指定できる任意の場所で使用できます。この検索条件は、TRUE または FALSE を返します。*contains-query-string* は、定数文字列か、クエリ時に既知である値を持つ変数である必要があります。

複数のカラムを指定する場合は、それらすべてが単一のベース・テーブルを参照している必要があります。**TEXT** インデックスは、複数のベース・テーブルにまたがることはできません。ベース・テーブルは、**FROM** 句で直接参照するか、ビューまたは抽出テーブルで参照できます。ただし、これは、ビューまたは抽出テーブルで、**DISTINCT**、**GROUP BY**、**ORDER BY**、**UNION**、**INTERSECT**、**EXCEPT**、またはロー制限が使用されていない場合にかぎります。

ANSI ジョイン構文 (**FULL OUTER JOIN**、**RIGHT OUTER JOIN**、**LEFT OUTER JOIN**) を使用するクエリはサポートされますが、最良のパフォーマンスが得られない場合があります。**FROM** 句で **CONTAINS** に外部ジョインを使用するのは、各 **CONTAINS** 句の `score` カラムが必要な場合のみにしてください。それ以外の場合は、**CONTAINS** を **ON** 条件または **WHERE** 句に移動してください。

次のタイプのクエリはサポートされていません。

- リモート・テーブルにジョインされている、フル **TEXT** インデックスを持つ SQL Anywhere テーブルを使用したリモート・クエリ。

- 使用されるフル **TEXT** インデックスが SQL Anywhere テーブルにある場合の、Sybase IQ テーブルと SQL Anywhere テーブルを使用するクエリ。
- TSQL スタイルの外部ジョイン構文 (*=*、=*、*=) を使用するクエリ。

長さが 32KB 未満の SQL 変数を検索単語として使用し、変数のデータ型が LONG VARCHAR の場合は、**CAST** を使用して変数を VARCHAR データ型に変換します。次に例を示します。

```
SELECT * FROM tab1 WHERE CONTAINS(c1, cast(v1 AS VARCHAR(64))
```

クエリ文字列での英数字以外の文字の使用については、次の警告を考慮する必要があります。

- 単語の途中でアスタリスクを使用するとエラーが返される。
- 英数字以外の文字 (特殊文字を含む) はスペースとして処理され、単語の分割記号として機能するので、fuzzy-expression での使用は避ける。
- 可能な場合は、クエリ文字列での特殊文字ではない英数字以外の文字の使用は避ける。特殊文字ではない英数字以外の文字を使用すると、それらの文字の場所で単語が分割され、それらを含む単語が語句として処理される。たとえば、`'things we've done'` は `'things "we ve" done'` と解釈される。

語句内では、アスタリスクが、特殊文字としてそのまま解釈される唯一の特殊文字です。語句内のその他のすべての特殊文字は、スペースとして処理され、単語の分割記号として機能します。

contains-query-string の解釈は、2 つの主要な段階を経て実行されます。

- 手順 1：演算子と優先度の解釈。この段階では、キーワードが演算子として解釈され、優先度の規則が適用される。
- 手順 2：テキスト設定オブジェクトの設定の適用。この段階では、テキスト設定オブジェクトの設定が単語に適用される。単語の長さの設定を超えているか、またはストップ・リストに含まれているクエリ単語は削除される。

参照：

- ファジー検索 (39 ページ)

CONTAINS 検索条件での演算子の優先度

クエリの評価中、式は、優先順位を使用して評価されます。

クエリ式を評価する場合の優先順位は、次のとおりです。

1. **FUZZY**、**NEAR**
2. **AND NOT**
3. **AND**
4. **OR**

アスタリスク (*) の許可される構文

アスタリスクは、クエリでのプレフィクス検索に使用します。

アスタリスクは、クエリ文字列の最後に配置するか、その後ろにスペース、アンパサンド、縦線、終了カッコ、終了引用符を続けることができます。その他の方法でアスタリスクを使用するとエラーが返されます。

表「アスタリスクの解釈」は、アスタリスクの許可される使用法を示しています。

表 9 : アスタリスクの解釈

クエリ文字列	同等のクエリ文字列	解釈
`'th*&best'`	`'th* AND best'` および `'th* best'`	th で始まる単語、および単語 best が含まれるものを検索する。
`'th*|best'`	`'th* OR best'`	th で始まる単語、または単語 best が含まれるものを検索する。
`'very&(best|th*)'`	`'very AND (best OR th*)'`	単語 very が含まれており、かつ th で始まる単語または単語 best が含まれるものを検索する。
`'"fast auto*"'`		単語 fast の直後に auto で始まる単語が続く語句が含まれるものを検索する。
`'"auto* price"'`		auto で始まる単語の直後に単語 price が続く語句が含まれるものを検索する。

注意 : アスタリスクを含むクエリ文字列の解釈は、テキスト設定オブジェクトの設定によって異なります。

ハイフン (-) の許可される構文

ハイフンは、単語の一部または式の反転としてクエリ内で使用でき、NOT と同じ機能を提供します。

ハイフンが反転として解釈されるかどうかは、クエリ文字列のどこにあるかによって異なります。たとえば、ハイフンを単語または式の直前に配置した場合は、反転と解釈されます。ハイフンを単語内に配置した場合は、ハイフンとして解釈されます。

ハイフンを反転に使用する場合は、ハイフンの前にスペースを挿入し、ハイフンの直後に式を配置する必要があります。

ファジー式の語句内で使用した場合、ハイフンはスペースとして処理され、単語の分割記号として機能します。

表「ハイフンの解釈」は、ハイフンの許可される構文を示しています。

表 10 : ハイフンの解釈

クエリ文字列	同等のクエリ文字列	解釈
`'the -best'`	`'the AND NOT best', 'the AND - best', 'the & - best', 'the NOT best'`	単語 the を含み、単語 best を含まないものを検索する。
`'the -(very best)'`	`'the AND NOT (very AND best)'`	単語 the を含み、単語 very および best を含まないものを検索する。
`'the -"very best"'`	`'the AND NOT "very best"'`	単語 the を含み、語句 very best を含まないものを検索する。
`'alpha-numerics'`	`'"alpha numerics"'`	単語 alpha の直後に単語 numerics が続く語句が含まれるものを検索する。
`'wild - west'`	`'wild west'` と `'wild AND west'`	単語 wild と単語 west の両方が含まれるものを検索する。

特殊文字の許可される構文

表「特殊文字の解釈」は、アスタリスクとハイフンを除くすべての特殊文字の許可される構文を示しています。

アスタリスクとハイフンの文字は、語句内で使用した場合は特殊文字とは見なされず、削除されます。

注意 : クエリ文字列には、文字列リテラルを指定する場合の制限も適用されます。たとえば、アポストロフィは、エスケープ・シーケンス内にある必要があります。

表 11 : 特殊文字の解釈

文字または構文	使用例および備考
アンパサンド (&)	アンパサンドは **AND** と同じであり、次のように指定できる。 • `'a & b'` • `'a &b'` • `'a& b'` • `'a&b'`

文字または構文	使用例および備考
縦線 (|)	縦線は **OR** と同じであり、次のように指定できる。 • `'a| b'` • `'a |b'` • `'a | b` • `'a|b'`
二重引用符 (")	二重引用符を使用して、順序と相対距離が重要な一連の単語を囲む。たとえば、クエリ文字列 `'learn "full text search"'` では、"full text search" が語句である。この例では、learn は語句の前または後にあっても、別のカラムに存在していても (**TEXT** インデックスが複数のカラムに構築されている場合) 構わないが、このとおりの語句が単一のカラムに存在している必要がある。
カッコ ()	カッコを使用して、式の評価の順序を指定する (デフォルトの順序と異なる場合)。たとえば、`'a AND (b|c)'` は、a、かつ b または c と解釈される。
波型記号 (~)	波型記号は **NEAR** と同じであり、特別な構文規則はない。クエリ文字列 `'full~text'` は `'full NEAR text'` と同じであり、次のように解釈される。単語 text から 10 単語内の範囲にある単語 full。
角カッコ [ ]	キーワード **NEAR** と組み合わせて角カッコを使用して、*distance* を囲む。その他の方法で角カッコを使用するとエラーが返される。

削除した単語の影響

TEXT インデックスでは、特定の条件を満たす単語が除外されることがあります。

TEXT インデックスは、**TEXT** インデックスの作成に使用されるテキスト設定オブジェクトに定義されている設定に従って構築されます。**TEXT** インデックスは、次のいずれかの条件に当てはまる単語を除外します。

- 単語がストップ・リストに含まれている。
- 単語が単語の最小長よりも短い (**GENERIC** のみ)。
- 単語が単語の最大長よりも長い。

同じ規則がクエリ文字列に適用されます。削除される単語は、語句の先頭または最後の 0 個以上の単語が合致する場合があります。たとえば、単語 'the' がストップ・リストに含まれているとします。

- この単語が、**AND**、**OR**、または **NEAR** の左右のいずれかにある場合、演算子と単語の両方が削除されます。たとえば、`'the AND apple'`、`'the OR apple'`、または `'the NEAR apple'` を検索することは、`'apple'` を検索することと同じです。

- この単語が **AND NOT** の右側にある場合、**AND NOT** と単語の両方が削除されます。たとえば、`'apple AND NOT the'` を検索することは、`'apple'` を検索することと同じです。
- この単語が **AND NOT** の左側にある場合、式全体が削除されます。たとえば、`'the AND NOT apple'` を検索した場合、ローは返されません。別の例を示します。`'orange and the AND NOT apple'` は `'orange AND (the AND NOT apple)'` と同じで、**AND NOT** 式が削除されて `'orange'` を検索することと同じになります。これを検索式 `'(orange and the) and not apple'` と対比してみてください。この式は、`'orange and not apple'` を検索することと同じです。
- この単語が語句に含まれる場合、語句は、削除される単語の位置にどのような単語があっても一致します。たとえば、`'feed the dog'` の検索は、`'feed the dog'`、`'feed my dog'`、`'feed any dog'` などに一致します。

注意： 検索対象のすべての単語が削除される場合、Sybase IQ は、エラー `CONTAINS has NULL search term` を返します。SQL Anywhere では、エラーは報告されず、ローは返されません。

クエリ一致スコア

一致度を示すスコアを使用して、クエリ結果をソートできます。

クエリの **FROM** 句に **CONTAINS** 句を含めると、それぞれの一致にスコアが関連付けられます。スコアは、一致の近さを示し、スコア情報を使用してデータをソートできます。次の 2 つの主な条件によってスコアが決定されます。

- インデックス・ローにおける単語の出現回数。インデックス・ローにおける単語の出現回数が多いほど、そのスコアは高くなる。
- **TEXT** インデックスにおける単語の出現回数。**TEXT** インデックスにおける単語の出現回数が多いほど、そのスコアは低くなる。

全文検索の種類によっては、その他の条件がスコアに影響します。たとえば、近接検索では、検索単語の近接性がスコアに影響します。デフォルトでは、**CONTAINS** 句の結果セットは相関名 `contains` を持ち、これには、`score` という単一のカラムが含まれます。`"contains".score` は、**SELECT** リスト、**ORDER BY** 句、またはクエリの他の部分で参照できます。ただし、`contains` は SQL の予約語であるため、二重引用符で囲む必要があります。または、`CONTAINS ( expression ) AS ct` のように、別の相関名も指定できます。全文検索の例では、`score` カラムを `ct.score` と呼んでいます。

次の文は、`MarketingInformation.Description` で、"stretch" または "comfort" で始まる単語を検索します。

```
SELECT ID, ct.score, Description
FROM MarketingInformation
```

```
CONTAINS ( MarketingInformation.Description,
           'stretch* | comfort*' )
AS ct ORDER BY ct.score DESC;
```

NGRAM TEXT インデックス検索

TEXT インデックス上のファジー検索と非ファジー検索は、タイプ **NGRAM** の **TEXT** インデックス上で可能です。

ファジー検索

TEXT インデックス上のファジー検索は、**TEXT** インデックスがタイプ **NGRAM** の場合のみ可能です。**GENERICTEXT** インデックスでは、ファジー検索は処理できません。

ファジー検索は、単語の入力ミスまたは変形を含む検索に使用できます。これを行うには、**FUZZY** 演算子を使用し、その後に、近接一致を見つけるための文字列を二重引用符で囲んで指定します。

FUZZY 演算子を使用するということは、文字列を手動で長さ *n* の部分文字列に分割し、それぞれを **OR** 演算子で分離することと同じです。たとえば、テキスト・インデックスを **NGRAM** 単語分割と 3 の **MAXIMUM TERM LENGTH** を使用して設定している場合、`'FUZZY "500 main street" '` は `'500 OR mai OR ain OR str OR tre OR ree OR eet'` を指定することと同じです。

FUZZY 演算子は、スコアを返す全文検索で役立ちます。多くの近接一致が返される可能性がありますが、通常、最高のスコアを持つ一致文字列のみが意味を持ちます。

注意： ファジー検索では、プレフィクス検索とサフィックス検索はサポートされません。たとえば、検索句に "v*" または "*vis" は指定できません。

あいまい検索の詳細については、『SQL Anywhere サーバ - SQL の使用法』>「データのクエリと変更」>「全文検索」>「全文検索を実行する方法」>「あいまい検索」を参照してください。

注意： このリファレンスは SQL Anywhere マニュアルにリンクされています。

例 1： NGRAM TEXT インデックス上のあいまい検索

テーブルと **NGRAMTEXT** インデックスを作成します。

```
CREATE TEXT CONFIGURATION NGRAMTxtcfg
   FROM default_char;
ALTER TEXT CONFIGURATION NGRAMTxtcfg TERM BREAKER    NGRAM;
ALTER TEXT CONFIGURATION NGRAMTxtcfg maximum term    length 3;
```

```
CREATE TABLE t_iq(a int, b varchar(100));
CREATE TEXT INDEX TXT_IQ on t_iq(b) CONFIGURATION    NGRAMTxtcfg
```

次のデータをテーブルに挿入します。

```
INSERT INTO t_iq values (1,'hello this is hira ');
INSERT INTO t_iq values(2, ' book he ookw worm okwo
kwor');
INSERT INTO t_iq values(3,'Michael is a good person');
INSERT INTO t_iq values(4,'hello this is evaa');
INSERT INTO t_iq values(5,'he is a bookworm');
INSERT INTO t_iq values (6,'boo ook okw kwo wor orm');
```

データを挿入したら、次のクエリを実行して、**NGRAMTEXT** インデックス上でファジー検索を行います。

```
SELECT * FROM t_iq WHERE CONTAINS (b,'FUZZY "bookerm"');
```

このクエリの結果を次に示します。

```
a          b
2          book he ookw worm okwo kwor
5          he is a bookworm
6          boo ook okw kwo wor orm
```

例 2： あいまい検索句での余分な文字

次のクエリは、ファジー検索句に余分な文字を指定する例を示します。

```
SELECT * FROM t_iq WHERE CONTAINS (b,'FUZZY "hellow"');
```

このクエリの結果を次に示します。

```
a          b
1          hello this is hira
4          hello this is evaa
```

例 3： あいまい検索句から文字の削除

次のクエリでは、ファジー検索句に 1 文字を削除して指定します。

```
SELECT * FROM t_iq WHERE CONTAINS(b, 'FUZZY "hllo"');
```

このクエリの結果を次に示します。

```
a          b
1          hello this is hira
4          hello this is evaa
```

非ファジー検索

NGRAM 上の非ファジー検索は、単語を対応する n-gram に分割し、n-gram を **NGRAMTEXT** インデックス内で検索します。

クエリ `CONTAINS ( M.Description, 'ams' ) ct;` は 2GRAM インデックス上の非ファジー **NGRAM** 検索を示しており、セマンティック的に検索クエリ `CONTAINS( M.Description, '"am ms"' ) ct;` と同じです。

2GRAM インデックス上で 'v*' 単語を検索すると、v で始まり、その後に任意のアルファベット文字が続く文字列が、検索単語に一致する 2GRAM と見なされ、結果として出力されます。

クエリ `CONTAINS (M.Description, ‘white whale’) ct;` は 3GRAM インデックス上の非ファジー **NGRAM** 検索を示しており、セマンティック的に検索クエリ `CONTAINS (M.Description, ‘”whi hit ite wha hal ale”’);` と同じです。

NGRAM のファジー検索と非ファジー検索の違いは、ファジー検索は、個々の GRAM の分離であるということです。非ファジー検索は、個々の GRAM の結合です。**GENERIC** と **NGRAM** の **TEXT** のインデックスを同じカラムに作成すると、**GENERICTEXT** インデックスが非ファジー検索を伴うクエリに使用され、**NGRAMTEXT** インデックスがファジー検索に使用されます。

例 1：同じカラムに GENERICTEXT インデックスを作成した後の非ファジー検索

次のクエリは、同じカラムに **GENERICTEXT** インデックスを作成した後の非ファジー検索を示しています。

```
SELECT * FROM t_iq WHERE CONTAINS (b,'bookworm');
```

このクエリの結果を次に示します。

```
            a
            b
5         he is a bookworm
```

例 2：同じカラムに NGRAM と GENERIC の両方の TEXT インデックスがある場合のファジー検索

次のクエリは、同じカラムに **NGRAM** と **GENERIC** の両方の **TEXT** インデックスがある場合のファジー検索を示しています。

```
SELECT * FROM t_iq
WHERE CONTAINS (b,’FUZZY “bookwerm”’);
```

このクエリの結果を次に示します。

```
a          b
2          book he ookw worm okwo kwor
5          he is a bookworm
6          boo ook okw kwo wor orm
```

例 3：非ファジー検索句でのファジー検索語句

次のクエリは、非ファジー検索句でのファジー検索語句の動作を示しています。

```
SELECT * FROM t_iq WHERE CONTAINS (b,'bookwerm');
```

このクエリでは、結果は何も返されません。

LONG BINARY カラムに対するクエリ

SELECT 文の **WHERE** 句では、LONG BINARY カラムは、**BYTE_LENGTH64**、**BYTE_SUBSTR64**、**BYTE_SUBSTR**、**BIT_LENGTH**、**OCTET_LENGTH**、**CHARINDEX**、**LOCATE** の各関数、IS NULL 式と IS NOT NULL 式のみで使用できます。

LONG BINARY カラムは、**SELECT** 文の **ORDER BY** 句、**GROUP BY** 句、**HAVING** 句では使用できません。 **DISTINCT** キーワードと一緒に使用することもできません。

Sybase IQ は、**LONG BINARY** (BLOB) のカラムまたは変数に対する **LIKE** 述部はサポートしていません。**LIKE** 述部を使用して LONG BINARY カラムのパターン検索を行うと、エラー `Invalid data type comparison in predicate` が返ります。

参照：

- 関数のサポート (81 ページ)

LONG VARCHAR カラムに対するクエリ

SELECT 文の **WHERE** 句では、LONG VARCHAR カラムは、**BIT_LENGTH**、**CHAR_LENGTH**、**CHAR_LENGTH64**、**CHARINDEX**、**LOCATE**、**OCTET_LENGTH**、**PATINDEX**、**SUBSTRING64**、**SUBSTRING** の各関数、IS NULL 式と IS NOT NULL 式のみで使用できます。

LIKE 述部を使用して、LONG VARCHAR カラムに対してパターン検索を実行できます。126 文字以下のパターンは、すべてサポートされています。254 文字よりも長いパターンは、サポートされていません。127 ～ 254 文字の長さのパターンは、パターンの内容によってはサポートされることがあります。

LIKE 述部は、任意のデータ・サイズの **LONG VARCHAR** (CLOB) 変数をサポートします。現在、SQL 変数で保持できる最大の長さは 2GB - 1 です。

LONG VARCHAR カラムは、**SELECT** 文の **ORDER BY** 句、**GROUP BY** 句、**HAVING** 句では使用できません。**DISTINCT** キーワード (**SELECT DISTINCT** と **COUNT DISTINCT**) と一緒に使用することもできません。

参照：

- 関数のサポート (81 ページ)

CONTAINS 述部のサポート

WORD (**WD**) インデックスを LONG VARCHAR (CLOB) カラムに作成し、**CONTAINS** 述部を使用して、カラムで最大長が 255 文字の文字列定数を検索できます。

CONTAINS 述部は、**WD** インデックスを使用している **LONG BINARY** (BLOB) カラムではサポートされません。**WD** インデックスを使用している LONG BINARY カラムに対して、**CONTAINS** 述部を使用して文字列検索を実行すると、エラーが返ります。外部ライブラリを使用する **TEXT** インデックスは、バイナリ・データでの **CONTAINS** をサポートしています。

『リファレンス：ビルディング・ブロック、テーブル、およびプロシージャ』の「SQL 言語の要素」>「検索条件」>「CONTAINS 条件」を参照してください。

LONG BINARY カラムと LONG VARCHAR カラムのパフォーマンスのモニタリング

Sybase IQ パフォーマンス・モニタは、LONG BINARY カラムと LONG VARCHAR カラムのパフォーマンス・データを表示します。

ストアド・プロシージャのサポート

LONG BINARY (BLOB) データ型と LONG VARCHAR (CLOB) データ型のカラムと全文検索に対するストアド・プロシージャのサポートについて説明します。

TEXT インデックス内の単語の管理

ストアド・プロシージャを使用して、文字列の単語への分割、**TEXT** インデックス内の単語の個数とそれぞれの位置の情報の取得、**TEXT** インデックスについての統計情報の表示を実行できます。

sa_char_terms システム・プロシージャ

CHAR 文字列を単語に分割し、各単語をローとして、その位置とともに返します。

構文

```
sa_char_terms( 'char-string' [, 'text-config-name'
[, 'owner' ] ] ]     )
```

パラメータ

char-string – 解析する CHAR 文字列。

text-config-name – 文字列の処理時に適用するテキスト設定オブジェクト。デフォルト値は 'default_char' です。

owner – 指定されたテキスト設定オブジェクトの所有者。デフォルト値は DBA です。

説明

sa_char_terms を使用して、テキスト設定オブジェクトの設定が適用されるときに、文字列がどのように解釈されるのかを確認できます。これは、インデックスの作成中またはクエリ文字列からどの単語が削除されるのかを知りたい場合に便利です。

パーミッション

なし。

例

CHAR 文字列 'the quick brown fox jumped over the fence' 内の単語が返されます。

```
CALL sa_char_terms
( 'the quick brown fox jumped over the fence' );
```

表 12 : CHAR 文字列の解釈

用語	位置
the	1
quick	2
brown	3
fox	4
jumped	5
over	6
the	7
fence	8

sa_nchar_terms システム・プロシージャ

NCHAR 文字列を単語に分割し、各単語をローとして、その位置とともに返します。

構文

```
sa_nchar_terms( 'char-string' [ , 'text-config-
name' [, 'owner' ] ] ]     )
```

パラメータ

char-string – 解析する NCHAR 文字列。

text-config-name – 文字列の処理時に適用するテキスト設定オブジェクト。デフォルト値は 'default_nchar' です。

owner – 指定されたテキスト設定オブジェクトの所有者。デフォルト値は DBA です。

説明

sa_nchar_terms を使用して、テキスト設定オブジェクトの設定が適用されるときに、文字列がどのように解釈されるのかを確認できます。これは、インデックスの作成中またはクエリ文字列からどの単語が削除されるのかを知りたい場合に便利です。

sa_nchar_terms の構文は、**sa_char_terms** システム・プロシージャの構文と似ています。

注意： NCHAR データ型は、**IN SYSTEM** テーブルについてのみサポートされています。

パーミッション
なし。

sa_text_index_stats システム・プロシージャ

データベース内の **TEXT** インデックスに関する統計情報を返します。

構文

```
sa_text_index_stats( )
```

説明
sa_text_index_stats を使用して、データベース内の各 **TEXT** インデックスの統計情報を表示します。

表 13 : sa_text_index_stats によって返される TEXT インデックスの統計情報

カラム名	データ型	説明
owner_id	UNSIGNED INT	テーブルの所有者の ID
table_id	UNSIGNED INT	テーブルの ID
index_id	UNSIGNED INT	**TEXT** インデックスの ID
text_config_id	UNSIGNED BIGINT	**TEXT** インデックスによって参照されるテキスト設定の ID
owner_name	CHAR(128)	所有者の名前
table_name	CHAR(128)	テーブルの名前
index_name	CHAR(128)	**TEXT** インデックスの名前
text_config_name	CHAR(128)	テキスト設定オブジェクトの名前
doc_count	UNSIGNED BIGINT	**TEXT** インデックス内のインデックス・カラム値の総数
doc_length	UNSIGNED BIGINT	**TEXT** インデックス内のデータの長さの合計
pending_length	UNSIGNED BIGINT	保留中の変更の長さの合計
deleted_length	UNSIGNED BIGINT	保留中の削除の長さの合計
last_refresh	TIMESTAMP	最終更新日時

`pending_length`、`deleted_length`、`last_refresh` の各値は、**IMMEDIATE REFRESHTEXT** インデックスの場合は NULL です。

パーミッション
DBA 権限が必要です。

例
データベース内の各 **TEXT** インデックスに関する統計情報を返します。

```
CALL sa_text_index_stats( );
```

sa_text_index_vocab システム・プロシージャ

TEXT インデックスに含まれるすべての単語と、各単語が含まれるインデックス値の合計数のリストを返します。

構文

```
sa_text_index_vocab(
    'text-index-name',
    'table-name',
    'table-owner'
    )
```

パラメータ
text-index-name – この CHAR(128) パラメータを使用して、**TEXT** インデックスの名前を指定します。

table-name – この CHAR(128) パラメータを使用して、**TEXT** インデックスが構築されているテーブルの名前を指定します。

table-owner – この CHAR(128) パラメータを使用して、テーブルの所有者を指定します。

説明
sa_text_index_vocab は、**TEXT** インデックスに含まれるすべての単語と、各単語が含まれるインデックス値の合計数を返します (これは、単語が一部のインデックス値で複数回出現する場合、合計出現回数未満となります)。

パラメータ値はホスト変数またはホスト式にできません。引数の *text-index-name*、*table-name*、*table-owner* は、制約または変数である必要があります。

パーミッション
DBA 権限、またはインデックス・テーブルに対する SELECT パーミッションが必要です。

例

sa_text_index_vocab を実行して、GROUPO によって所有されるテーブル Customers の **TEXT** インデックス MyTextIndex に含まれるすべての単語を返します。

```
sa_text_index_vocab
('MyTextIndex','Customers','GROUPO');
```

表 14 : インデックス内の単語

用語	出現回数
a	1
Able	1
Acres	1
Active	5
Advertising	1
Again	1
...	...

外部ライブラリの確認

sa_list_external_library ストアド・プロシージャは、現在サーバにロードされているライブラリを示します。確認後に、ライブラリをサーバからアンロードするには、**sa_external_library_unload** を使用します。

sa_external_library_unload システム・プロシージャ

外部ライブラリをアンロードします。

構文

```
sa_external_library_unload ( [ 'external-library'  ] )
```

パラメータ

external-library– 必要に応じて、この LONG VARCHAR パラメータを使用して、アンロードするライブラリの名前を指定します。ライブラリが指定されていない場合、使用されていないすべての外部ライブラリがアンロードされます。

説明
指定した外部ライブラリが使用中であるか、ロードされていない場合は、エラーが返されます。パラメータを指定しない場合で、ロードされている外部ライブラリが見つからないときは、エラーが返されます。

パーミッション
DBA 権限が必要です。

例
`myextlib.dll` という名前の外部ライブラリをアンロードします。

```
CALL sa_external_library_unload( 'myextlib.dll' );
```

現在使用されていないすべてのライブラリをアンロードします。

```
CALL sa_external_library_unload();
```

sa_list_external_library プロシージャ

現在サーバにロードされている外部ライブラリをリストします。

構文

```
sa_list_external_library( )
```

説明
エンジンにロードされている外部ライブラリのリストを、参照カウントとともに返します。

参照カウントは、エンジン内のライブラリのインスタンス数です。**sa_external_library_unload** プロシージャを実行して外部ライブラリをアンロードできるのは、その参照カウントが 0 の場合のみです。

パーミッション
DBA 権限が必要です。

例
外部ライブラリとその参照カウントを一覧表示します。

```
CALL sa_list_external_library()
```

ラージ・オブジェクト・データの圧縮

ラージ・オブジェクト・カラムの圧縮の制御は、**sp_iqsetcompression** ストアド・プロシージャで行います。

sp_iqsetcompression は、データベース・バッファをディスクに書き込むときに実行される、LONG BINARY データ型と LONG VARCHAR データ型のカラムの圧縮を制御します。**sp_iqsetcompression** を使用して圧縮を無効にもできます。LONG BINARY カラムまたは LONG VARCHAR カラムに格納されている特定のデータ・フォーマット (JPG ファイルなど) は既に圧縮されており、圧縮を重ねても意味がないので、この機能を使用することで CPU サイクルを節約できます。

ラージ・オブジェクト・カラムの圧縮設定の表示は、**sp_iqshowcompression** ストアド・プロシージャで行います。

sp_iqsetcompression プロシージャ

LONG BINARY (BLOB) データ型と LONG VARCHAR (CLOB) データ型のカラムのデータの圧縮を設定します。

構文

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

パーミッション

DBA 権限が必要です。

説明

sp_iqsetcompression は、LONG BINARY (BLOB) データ型と LONG VARCHAR (CLOB) データ型のカラムの圧縮を制御します。圧縮設定は、Sybase IQ ベース・テーブルにのみ適用されます。

sp_iqsetcompression には、圧縮設定を変更した後に **COMMIT** が実行されるという二次的な影響があります。

表 15 : sp_iqsetcompression パラメータ

名前	説明
owner	圧縮を設定するテーブルの所有者
table	圧縮を設定するテーブル
column	圧縮を設定するカラム

名前	説明
on_off_flag	圧縮設定：圧縮を有効にする場合は ON、無効にする場合は OFF

例
次のテーブル定義を想定します。

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

LOB カラム picJPG で圧縮を無効にするには、**sp_iqsetcompression** を呼び出します (この作業を行うには DBA パーミッションが必要です)。

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

このコマンドはローを返しません。

sp_iqshowcompression プロシージャ

LONG BINARY (BLOB) データ型と LONG VARCHAR (CLOB) データ型のカラムの圧縮設定を表示します。

構文

```
sp_iqshowcompression ( owner, table, column )
```

パーミッション
DBA 権限が必要です。

説明
カラム名と圧縮設定を返します。圧縮設定値は 'ON' (圧縮は有効) および 'OFF' (圧縮は無効) です。

表 16 : sp_iqshowcompression パラメータ

名前	説明
owner	圧縮を設定するテーブルの所有者
table	圧縮を設定するテーブル
column	圧縮を設定するカラム

例
次のテーブル定義を想定します。

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

pixTable テーブルのカラムの圧縮ステータスをチェックするには、**sp_iqshowcompression** を呼び出します (この作業を行うには DBA パーミッションが必要です)。

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

このコマンドは 1 つのローを返します。

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

ラージ・オブジェクト・カラムについての情報

ストアド・プロシージャ **sp_iqindexsize** では、LONG BINARY と LONG VARCHAR の個々のカラムのサイズを表示できます。

LONG BINARY カラムのサイズ

次の **sp_iqindexsize** 出力は、約 42GB のデータが格納されている LONG BINARY カラムを示しています。

ページ・サイズは 128KB です。largelob Info 型は最後のローにあります。

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

この例では、圧縮率は 42953952/(623009*128) = 53.9% です。

LONG VARCHAR カラムのサイズ

次の **sp_iqindexsize** 出力は、約 42GB のデータが格納されている LONG VARCHAR カラムを示しています。

ページ・サイズは 128KB です。largelob Info 型は最後のローにあります。

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
```

この例では、圧縮率は 42953952/(623009*128) = 53.9% です。

ラージ・オブジェクト・データのロードとアンロード

Sybase IQ でラージ・オブジェクト・データのエクスポートとロードを行う方法について説明します。

ラージ・オブジェクト・データのエクスポート

Sybase IQ データ抽出機能には、個別の LONG BINARY セルと LONG VARCHAR セルをサーバ上の個別のオペレーティング・システム・ファイルに抽出できる **BFILE** 関数が用意されています。

BFILE は、データ抽出機能と一緒に使用できるほか、単独でも使用できます。

BFILE 関数

個別の LONG BINARY セルと LONG VARCHAR セルをサーバ上の個別のオペレーティング・システム・ファイルに抽出します。

構文

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

パラメータ

file-name-expression – LONG BINARY または LONG VARCHAR のデータが書き込まれる出力ファイルの名前。このファイル名の最大長は (32K -1) バイトですが、ファイル・システムによってサポートされている有効なパス名とします。

large-object-column – LONG BINARY カラムまたは LONG VARCHAR カラムの名前。

使用法

BFILE は次の値を返します。

- 1 (ファイルの書き込みが成功した場合)
- 0 (ファイルが開かれていない、または書き込みが失敗した場合)
- NULL (LONG BINARY セルまたは LONG VARCHAR セルの値が NULL の場合)

LONG BINARY セルまたは LONG VARCHAR セルの値が NULL の場合、ファイルは開かれず、データも書き込まれません。

ファイル・パスは、サーバが開始されたロケーションから見た相対指定です。ファイルを開いて書き込むには、サーバ・プロセスのパーミッションが必要です。**BFILE** 出力ファイルについては、テープ・デバイスはサポートされていません。

BFILE 関数以外によって (つまり、クライアント／サーバ・データベースに接続した後で) 取得された LONG BINARY セルと LONG VARCHAR セルの最大長は、2GB です。**SELECT** (**SELECT**、**OPEN CURSOR**) を使用して 2GB を超える LONG BINARY セルを取得するには、**SUBSTRING64** または **BYTE_SUBSTR64** を使用します。**SELECT** (**SELECT**、**OPEN CURSOR**) を使用して 2GB を超える LONG VARCHAR セルを取得するには、**SUBSTRING64** を使用します。なお、ODBC、JDBC™、Open Client™ などの一部の接続ドライバの場合、2GB を超える値は単一の **SELECT** では返されません。

BFILE は、データ抽出機能と一緒に使用できるほか、単独でも使用できます。

BFILE 関数の例

BFILE を使用して、LOB データの抽出と再ロードを行います。

テーブル LobA を作成します。

```
create table LobA
  (rowid  int primary key,
    col1   clob null,
    col2   blob null)
```

LobA には、2 つのローのデータがあると想定します。

非 LOB データとファイルへのパスを、LOB データの抽出先に抽出します。

```
BEGIN
  SET TEMPORARY OPTION
    Temp_Extract_Name1 = LobA_data.txt';
  SELECT rowid,
    'row' + string(rowid) + '.' + 'col1',
    'row' + string(rowid) + '.' + 'col2'
  FROM LobA;
END
```

ファイル LobA_data.txt が作成され、この非 LOB データとこれらのファイル名が格納されます。

```
  1,row1.col1,row1.col2,
  2,row2.col1,row2.col2,
```

LOB データの抽出を実行します。

```
SELECT
  BFILE('row' + string(rowid) + '.' + 'col1',col1),
  BFILE('row' + string(rowid) + '.' + 'col2',col2)
FROM LobA;
```

抽出が終了すると、抽出された LOB データのセルごとにファイルが作成されます。たとえば、テーブル LobA に、rowid 値として 1 と 2 を持つ 2 つのローのデータが含まれている場合、次のファイルが作成されます。

- row1.col1
- row1.col2
- row2.col1
- row2.col2

抽出されたデータを再ロードします。

```
LOAD TABLE LobA
(rowid,
  col1 ASCII FILE (',') NULL('NULL'),
  col2 BINARY FILE (',') NULL('NULL'))
FROM LobA_data.txt'
DELIMITED BY ','
ROW DELIMITED BY '¥n'
ESCAPES OFF;
```

ラージ・オブジェクト・データのロード

LOAD TABLE 文の拡張構文を使用して、LONG BINARY データと LONG VARCHAR データをロードします。

オペレーティング・システムによって制限されていないかぎり、無制限のサイズのラージ・オブジェクト・データを、ASCII フォーマットまたは BCP フォーマットでプライマリ・ファイルからロードできます。プライマリ・ファイルからラージ・オブジェクト・カラムにロードされる固定幅データの最大長は 32K - 1 です。

ここで、プライマリ・ロード・ファイルにセカンダリ・ロード・ファイルを指定することもできます。個々のセカンダリ・データ・ファイルには、LONG BINARY セルまたは LONG VARCHAR セルの値が 1 つだけあります。

拡張 LOAD TABLE 構文

LOAD TABLE には、ラージ・オブジェクト・データをロードするための拡張構文が用意されています。

```
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' )
```

キーワード **BINARY FILE** (LONG BINARY 用) または **ASCII FILE** (LONG VARCHAR 用) は、カラムのプライマリ入力ファイルには、LONG BINARY データ自体または LONG VARCHAR データ自体ではなく、LONG BINARY セルまたは LONG VARCHAR セルの値を含むセカンダリ・ファイルのパスが含まれていることをロード処理に指示します。セカンダリ・ファイルのパス名は、完全修飾されたパスまたは相対パスのいずれかを使用できます。セカンダリ・ファイルのパス名が完全修飾されたパスでない場合は、パスはサーバが起動されたディレクトリからの相対パスです。セカンダリ・ファイルについては、テープ・デバイスはサポートされていません。

Sybase IQ では、プライマリ・ロード・ファイルの無制限の長さ (オペレーティング・システムの制限に従います) の LONG BINARY 値と LONG VARCHAR 値をロードできます。16 進数形式のバイナリ・データをプライマリ・ファイルから LONG BINARY カラムにロードする場合、Sybase IQ では、16 進数字の合計数は偶数である必要があります。セルの値に奇数個の 16 進数字が含まれている場合、"Odd length of binary data value detected on column" というエラーが報告されます。LONG BINARY をロードする入力ファイルには、必ず偶数個の 16 進数字が含まれている必要があります。

Sybase IQ では、**LOAD TABLE...FORMAT BINARY** 句を使用してプライマリ・ファイルからラージ・オブジェクト・カラムをロードすることはサポートされていません。セカンダリ・ファイルからは、バイナリ・フォーマットでラージ・オブジェクト・データをロードできます。

バイナリ形式を使用したデータのロードの詳細については、『システム管理ガイド：第 1 巻』の「データのインポートとエクスポート」>「BINARY ロード形式」を参照してください。

LOAD TABLE FORMAT BCP では、ロード仕様にカラム名、**NULL**、**ENCRYPTED** のみを含めることができます。つまり、**LOAD TABLE FORMAT BCP** オプションを使用して LONG BINARY カラムと LONG VARCHAR カラムをロードする場合、セカンダリ・ファイルは使用できません。

『リファレンス：文とオプション』の「SQL 文」>「LOAD TABLE 文」を参照してください。

ラージ・オブジェクト・データのロード例

LONG BINARY データを使用するテーブルを作成し、ロードします。

```
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;
```

プライマリ・ファイル abc.inp には次のデータがあります。

```
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,
```

LONG BINARY データがテーブル tab にロードされると、1 番目と 2 番目のローの lobcol カラムには、それぞれ boston.jpg ファイルと concord.bmp ファイルの内容が入ります。3 番目のローには長さ 0 の値が、4 番目のローには NULL がそれぞれ入ります。

ロード・エラーの制御

データベース・オプション SECONDARY_FILE_ERROR では、セカンダリ・ファイル (**BINARY FILE** または **ASCII FILE**) を開けなかった場合、またはその読み取りに失敗した場合に実行する、ロードのアクションを指定できます。

SECONDARY_FILE_ERROR を ON にすると、セカンダリ・ファイル (**BINARY FILE** または **ASCII FILE**) を開けなかった場合、またはその読み取りに失敗した場合、ロードはロールバックします。

SECONDARY_FILE_ERROR を OFF (デフォルト) にすると、セカンダリ・ファイル (**BINARY FILE** または **ASCII FILE**) を開けなかった場合、またはその読み取りに失敗した場合でも、ロードは続行されます。LONG BINARY セルまたは LONG VARCHAR セルは次のいずれかの値となります。

- NULL (カラムが NULL を許可する場合)
- 長さ 0 の値 (カラムが NULL を許可しない場合)

SECONDARY_FILE_ERROR は任意のユーザが PUBLIC グループに対して設定したり、一時的に設定したりできます。また、その設定はただちに有効になります。

整合性制約違反をロード・エラー **ROW LOG** ファイルにロギングすると、LONG BINARY カラムまたは LONG VARCHAR カラムには次の情報が記録されます。

- プライマリ・データ・ファイルから実際に読み取ったテキスト (1 回目のロード操作でロギングが発生した場合)
- 長さ 0 の値 (2 回目のロード操作でロギングが発生した場合)

後続ブランクを含むラージ・オブジェクト・データのロード

LOAD TABLE...STRIP オプションは LONG VARCHAR データには何の影響も及ぼしません。

STRIP オプションが ON でも、LONG VARCHAR データの後続ブランクは削除されません。

引用符を含むラージ・オブジェクト・データのロード

LOAD TABLE...QUOTES オプションは、その設定に関係なく、セカンダリ・ファイルからの LONG BINARY (BLOB) データまたは LONG VARCHAR (CLOB) データのロードに適用されません。

開始引用符または終了引用符は、CLOB データの一部としてロードされます。引用符で囲まれている 2 つの連続した引用符は、**QUOTESON** オプションを使用すると 2 つの連続した引用符としてロードされます。

部分的なマルチバイト文字データのトランケート

部分的なマルチバイト LONG VARCHAR データは、TRIM_PARTIAL_MBC データベース・オプションの値に従ってロード時にトランケートされます。

- TRIM_PARTIAL_MBC を ON にすると、プライマリ・データと **ASCII FILE** オプションを指定して **LOAD** した値の両方で、部分的なマルチバイト文字はトランケートされます。
- TRIM_PARTIAL_MBC を OFF にすると、**ASCII FILE** オプションを指定した **LOAD** は、部分的なマルチバイト文字を SECONDARY_FILE_ERROR データベース・オプションの値に従って処理します。

表「LONG VARCHAR ロード時 (ASCII FILE オプション指定) のマルチバイト文字の一部」では、TRIM_PARTIAL_MBC と SECONDARY_FILE_ERROR の値に応じて後続マルチバイト文字がどのようにロードされるかを説明します。

表 17 : LONG VARCHAR ロード時 (ASCII FILE オプション指定) のマルチバイト文字の一部

TRIM_PARTIAL_MBC	SECONDARY_FILE_ERROR	一部の後続マルチバイト文字の処理
ON	ON/OFF	後続マルチバイト文字の一部をトランケート

TRIM_PARTIAL_MBC	SECONDARY_FILE_ERROR	一部の後続マルチバイト文字の処理
OFF	ON	セル - NULL (NULL が許容されている場合) LOAD エラー - ロールバック (NULL が許容されていない場合)
OFF	OFF	セル - NULL (NULL が許容されている場合) セル - 長さ 0 の値 (NULL が許容されていない場合)

ラージ・オブジェクト変数のロード・サポート

LOAD TABLE、**INSERT…VALUES**、**INSERT…SELECT**、**INSERT…LOCATION**、**SELECT…INTO**、**UPDATE** の各 SQL 文でのラージ・オブジェクト変数のサポート情報については、「ラージ・オブジェクト・データ型」を参照してください。

参照：

- ラージ・オブジェクト・データ型 (63 ページ)
- ラージ・オブジェクト変数 (67 ページ)

ラージ・オブジェクト・データ型

ラージ・オブジェクト LONG BINARY データ型カラムとラージ・オブジェクト LONG VARCHAR データ型カラムの特性、ラージ・オブジェクト・データのインデックスのサポートについて説明します。

ラージ・オブジェクト・データ型、LONG BINARY と BLOB

Sybase IQ のバイナリ・ラージ・オブジェクト (BLOB) データは、LONG BINARY または BLOB のデータ型のカラムに格納されます。

個々の LONG BINARY データ型の取り得る長さは、IQ ページ・サイズが 128KB の場合は 0 ～ 512TB (テラバイト)、IQ ページ・サイズが 512KB の場合は 0 ～ 2PB (ペタバイト) です (最大長は、4GB にデータベース・ページ・サイズを掛けた値になります)。LONG BINARY データが格納されているテーブルをサポートするには、IQ ページ・サイズを最低 128KB (131072 バイト) に設定して、IQ データベースを作成します。

テーブルまたはデータベースは、それぞれ、テーブルあたりのサポートされる最大カラム数、データベースあたりのサポートされる最大カラム数を上限として、LONG BINARY カラムをいくつでも持つことができます。

LONG BINARY カラムでは、NULL または NOT NULL が許容され、長さ 0 の値を格納できます。ドメイン BLOB は、NULL を許容する LONG BINARY データ型です。

非 FP インデックスまたはジョイン・インデックスは、LONG BINARY カラムには作成できません。

結果セットに BLOB カラムが含まれている場合、プリフェッチは無効です。

LONG BINARY カラムを、**UPDATE**、**INSERT**、**LOAD TABLE**、**DELETE**、**TRUNCATE**、**SELECT...INTO**、**INSERT...LOCATION** の各 SQL 文を使用して変更します。位置付け更新と位置付け削除は、LONG BINARY カラムではサポートされていません。

Adaptive Server Enterprise IMAGE カラムを LONG BINARY カラムに **INSERT...LOCATION** コマンドを使用して挿入できます。挿入されたすべての IMAGE データは、2147483648 バイト (2GB) を超えた分が暗黙的に右トランケートされます。

LONG BINARY データ型の変換

LONG BINARY データ型との間と非 LONG BINARY データ型との間で、限定された暗黙的なデータ型変換が行われます。

LONG BINARY データ型から他の非 LONG BINARY データ型 (**INSERT** と **UPDATE** における BINARY データ型と VARBINARY データ型を除きます) への暗黙的なデータ型変換は行われません。TINYINT、SMALLINT、INTEGER、UNSIGNED INTEGER、BIGINT、UNSIGNED BIGINT、CHAR、VARCHAR のデータ型から LONG BINARY データ型へは、暗黙的に変換が行われます。BIT、REAL、DOUBLE、または NUMERIC のデータ型から LONG BINARY データ型への暗黙的なデータ変換は行われません。暗黙的な変換は、 CONVERSION_MODE データベース・オプションを使用して制御できます。

LONG BINARY データ型に対して現在サポートされているバイト部分文字列関数は、**INSERT** 文と **UPDATE** 文の暗黙的な変換の入力として受け入れられます。「関数のサポート」を参照してください。

LONG BINARY データ型は、明示的に BINARY または VARBINARY に変換できます。LONG BINARY データ型が変換先または変換元となるその他の明示的なデータ型変換 (たとえば、**CAST** 関数または **CONVERT** 関数によるデータ型変換) は行われません。

LONG BINARY から BINARY または VARBINARY への変換中の LONG BINARY データのトランケーションは、BINARY データと VARBINARY データのトランケーションが処理されるのと同じ方法で処理されます。STRING_RTRUNCATION オプションが ON の場合は、バイナリ・カラムの **INSERT** または **UPDATE** で (スペース以外の文字だけでなくすべての値の) 右トランケートを実行すると、トランケーションエラーが発生し、トランザクションがロールバックされます。

参照：

- 関数のサポート (81 ページ)

ラージ・オブジェクト・データ型、LONG VARCHAR と CLOB

Sybase IQ のキャラクタ・ラージ・オブジェクト (CLOB) データは、LONG VARCHAR または CLOB のデータ型のカラムに格納されます。

個々の LONG VARCHAR データ型の取り得る長さは、IQ ページ・サイズが 128KB の場合は 0 ～ 512TB (テラバイト)、IQ ページ・サイズが 512KB の場合は 0 ～ 2PB

(ペタバイト) です (最大長は、4GB にデータベース・ページ・サイズを掛けた値になります)。LONG VARCHAR データが格納されているテーブルをサポートするには、IQ ページ・サイズを最低 64KB (65536 バイト) に設定して、IQ データベースを作成します。

テーブルまたはデータベースは、それぞれ、テーブルあたりのサポートされる最大カラム数、データベースあたりのサポートされる最大カラム数を上限として、LONG VARCHAR カラムをいくつでも持つことができます。

Sybase IQ は、シングルバイトとマルチバイトの両方の LONG VARCHAR データをサポートします。

LONG VARCHAR カラムでは、NULL または NOT NULL が許容され、長さ 0 の値を格納できます。ドメイン CLOB は、NULL を許容する LONG VARCHAR データ型です。非 NULL LONG VARCHAR カラムを作成するには、NOT NULL をカラム定義に明示的に指定します。

テーブルを作成するとき、または既存のテーブルにカラムを追加するときは、ドメイン CLOB を使用して LONG VARCHAR カラムを作成できます。次に例を示します。

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

```
ALTER TABLE lvtab ADD c4 CLOB;
```

LONG VARCHAR カラムに **WORD** (**WD**) インデックスを作成できます。その他の非 **FP** インデックスとジョイン・インデックスは、LONG VARCHAR カラムには作成できません。

LONG VARCHAR カラムを、**UPDATE**、**INSERT...VALUES**、**INSERT...SELECT**、**LOAD TABLE**、**DELETE**、**TRUNCATE**、**SELECT...INTO**、**INSERT...LOCATION** の各 SQL 文を使用して変更できます。位置付け更新と位置付け削除は、LONG VARCHAR カラムではサポートされていません。

Adaptive Server Enterprise TEXT カラムを LONG VARCHAR カラムに **INSERT...LOCATION** コマンドを使用して挿入できます。挿入されたすべての TEXT データは、2147483648 バイト (2GB) を超えた分が暗黙的に右トランケートされます。

LONG VARCHAR データ型の変換

LONG VARCHAR データ型との間と非 LONG VARCHAR データ型との間で、限定された暗黙的なデータ型変換が行われます。

LONG VARCHAR データ型から他の非 LONG VARCHAR データ型 (LONG BINARY と、**INSERT** と **UPDATE** のみにおける CHAR と VARCHAR を除く) への暗黙的なデータ変換は行われません。CHAR データ型と VARCHAR データ型から LONG VARCHAR データ型への暗黙的な変換が行われます。BIT、REAL、DOUBLE、NUMERIC、TINYINT、SMALLINT、INT、UNSIGNED INT、BIGINT、UNSIGNED BIGINT、BINARY、VARBINARY、または LONG BINARY のデータ型から LONG VARCHAR データ型への暗黙的な変換は行われません。暗黙的な変換は、CONVERSION_MODE データベース・オプションを使用して制御できます。

LONG VARCHAR データ型に対して現在サポートされている文字列関数は、**INSERT** 文と **UPDATE** 文の暗黙的な変換の入力として受け入れられます。「関数のサポート」を参照してください。

LONG VARCHAR データ型は、明示的に CHAR と VARCHAR に変換できます。LONG VARCHAR データ型が変換先または変換元となるその他の明示的なデータ型変換 (たとえば、**CAST** 関数または **CONVERT** 関数によるデータ型変換) は行われません。

LONG VARCHAR から CHAR への変換中の LONG VARCHAR データのトランケーションは、CHAR データのトランケーションが処理されるのと同じ方法で処理されます。STRING_RTRUNCATION オプションが ON の場合は、スペース以外の文字の文字列の右トランケートを実行すると、トランケーション・エラーが報告され、トランザクションがロールバックされます。末尾にある部分的なマルチバイト文字は、変換時にスペースに置き換えられます。

LONG VARCHAR から VARCHAR への変換中の LONG VARCHAR データのトランケーションは、VARCHAR データのトランケーションが処理されるのと同じ方法で処理されます。STRING_RTRUNCATION オプションが ON の場合は、スペース以外の文字の文字列の右トランケートを実行すると、トランケーション エラーが報告され、トランザクションがロールバックされます。末尾にある部分的なマルチバイト文字は、変換時にトランケートされます。

参照：

- 関数のサポート (81 ページ)

ラージ・オブジェクト変数

Sybase IQ はラージ・オブジェクト変数をサポートします。

インバウンドの LONG BINARY 変数と LONG VARCHAR 変数 (IQ が使用するホスト変数または SQL 変数) に最大長はありません。

アウトバウンドの LONG BINARY 変数と LONG VARCHAR 変数 (IQ が設定する変数) の最大長は 2GB - 1 です。

LOAD TABLE、**INSERT…VALUES**、**INSERT…SELECT**、**INSERT…LOCATION**、**SELECT…INTO**、**UPDATE** の各 SQL 文は、任意のデータ・サイズの LONG BINARY 変数と LONG VARCHAR 変数を受け入れます。現在、SQL 変数で保持できる最大の長さは 2GB - 1 です。

BIT_LENGTH、**BYTE_LENGTH**、**BYTE_LENGTH64**、**BYTE_SUBSTR**、**BYTE_SUBSTR64**、**CHARINDEX**、**LOCATE**、 **OCTET_LENGTH**、**SUBSTRING64** の各関数は、SQL 変数が保持できる任意のデータ・サイズの LONG BINARY 変数と LONG VARCHAR 変数をサポートします。さらに、**CHAR_LENGTH**、**CHAR_LENGTH64**、**PATINDEX**、**SUBSTR**、**SUBSTRING** の各関数は、SQL 変数が保持できる任意のデータ・サイズの LONG VARCHAR 変数をサポートします。

ラージ・オブジェクト変数のデータ型変換

データベース オプション ENABLE_LOB_VARIABLES は、ラージ・オブジェクト変数のデータ型変換を制御します。

ENABLE_LOB_VARIABLES オプション

ラージ・オブジェクト変数のデータ型変換を制御します。

指定できる値
ON、OFF

デフォルト値
OFF

スコープ
このオプションを設定するために、DBA パーミッションは必要ありません。個々の接続に temporary レベルで設定できます。また、PUBLIC グループに設定できます。すぐに有効になります。

説明
ENABLE_LOB_VARIABLES は、ラージ・オブジェクト変数のデータ型変換を制御します。

ENABLE_LOB_VARIABLES が OFF の場合、32K 未満のラージ・オブジェクト変数は暗黙的に変換されます。ラージ・オブジェクト変数が 32K 以上の場合はエラーが報告されます。LONG VARCHAR 変数は VARCHAR データ型に暗黙的に変換され、32K を超えた部分がトランケートされます。LONG BINARY 変数は VARBINARY データ型に暗黙的に変換され、32K を超えた部分がトランケートされます。

ENABLE_LOB_VARIABLES が ON の場合、任意のサイズのラージ・オブジェクト変数で元のデータ型とサイズが保持されます。

例
32K を超えるラージ・オブジェクト変数のデータ型とサイズを保持するには、次のように入力します。

```
SET TEMPORARY OPTION ENABLE_LOB_VARIABLES = ON
```

ラージ・オブジェクト・カラムのインデックスのサポート

Sybase IQ は LONG BINARY カラムと LONG VARCHAR カラムの **TEXT** インデックス、LONG VARCHAR カラムの **WORD** (**WD**) インデックスをサポートします。

ラージ・オブジェクト・カラムの TEXT インデックスのサポート

Sybase IQ **TEXT** インデックスは、LONG BINARY カラムと LONG VARCHAR カラムをサポートします。

「SQL 文のサポート」と「TEXT インデックスとテキスト設定オブジェクト」を参照してください。

参照：
- SQL 文のサポート (71 ページ)
- TEXT インデックスとテキスト設定オブジェクト (3 ページ)

LONG VARCHAR (CLOB) カラムの WD インデックスのサポート

Sybase IQ は、LONG VARCHAR (CLOB) カラムの **WORD** (**WD**) インデックスを限定的にサポートします。

- CHAR、VARCHAR、LONG VARCHAR のデータ型のカラムに **WD** インデックスを Sybase Central で作成できる。
- **WD** インデックスによってサポートされる最大幅のカラムは、LOB カラムの最大幅 (最大長は、4GB にデータベース・ページ・サイズを掛けた値になります)。

  Sybase IQ によってサポートされる最大ワード長は 255 バイト。
- CHAR カラムと VARCHAR カラムでの **WD** インデックスの **sp_iqcheckdb** オプションもすべて、LONG VARCHAR (CLOB) カラムで、それも割り付け、チェック、検証の各モードでサポートされる。
- **sp_iqrebuildindex** を使用して、LONG VARCHAR (CLOB) カラムの **WD** インデックスを再構築できる。

バイナリ・フォーマットで書かれた中国語のテキストや文書では、ETL の前処理を実施し、単語を見つけ出して **WD** インデックスが解析できる形に変換する必要があります。

SQL 文のサポート

TEXT インデックスとテキスト設定の操作をサポートする SQL 文および構文について説明します。

ALTER TEXT CONFIGURATION 文

テキスト設定オブジェクトを変更します。

構文

```
ALTER TEXT CONFIGURATION [ owner.]config-name
   STOPLIST
            stoplist
   | DROP STOPLIST
   | { MINIMUM | MAXIMUM } TERM LENGTH
            integer
   |  TERM BREAKER
      { GENERIC
        [ EXTERNAL NAME library-and-entry-point-name-string ]
       | NGRAM }
   |  PREFILTER EXTERNAL NAME library-and-entry-point-name-string
```

```
stoplist: string-expression
```

```
library-and-entry-point-name-string: [operating-system:]function-
name@library
```

例

- **例 1 –** テキスト設定オブジェクト `maxTerm16` を作成し、単語の最大長を 16 に変更します。

  ```
  CREATE TEXT CONFIGURATION maxTerm16 FROM default_char;
  ```

  ```
  ALTER TEXT CONFIGURATION maxTerm16 MAXIMUM TERM LENGTH 16;
  ```

- **例 2 –** ストップリスト単語を `maxTerm16` 設定オブジェクトに追加します。

  ```
  ALTER TEXT CONFIGURATION maxTerm16
  ```

  ```
  STOPLIST 'because about therefore only';
  ```

- **例 3 –** 外部ライブラリ `mytermbreaker.dll` のエントリ・ポイント `my_term_breaker` を使用してテキストを分割するように、テキスト設定オブジェクト `my_text_config` を更新します。

```
CREATE TEXT CONFIGURATION my_text_config FROM default_char;

ALTER TEXT CONFIGURATION my_text_config

TERM BREAKER GENERIC EXTERNAL NAME
'platform:my_term_breaker@mytermbreaker';
```

- **例 4 –** 外部ライブラリ `myprefilter.dll` のエントリ・ポイント `my_prefilter` を使用してドキュメントのプレフィルタを実行するように、テキスト設定オブジェクト `my_text_config` を更新します。

```
ALTER TEXT CONFIGURATION my_text_config
PREFILTER EXTERNAL NAME 'platform:my_prefilter@myprefilter';
```

使用法

ALTER TEXT CONFIGURATION を使用して、テキスト設定オブジェクトを変更します。

TEXT インデックスは、テキスト設定オブジェクトに依存します。Sybase IQ の **TEXT** インデックスでは、即時更新が使用され、トランケートすることはできません。テキスト設定オブジェクトを変更する前に、インデックスを削除する必要があります。

テキスト設定オブジェクトの設定を表示するには、**SYSTEXTCONFIG** システム・ビューに対してクエリを実行します。

STOPLIST 句 – この句を使用して、**TEXT** インデックスの構築時に無視する単語のリストを作成したり、置き換えたりします。このリストで指定されている単語は、クエリでも無視されます。ストップリスト単語はスペースで区切ります。

ストップリスト単語にスペースを含めることはできません。ストップリスト単語には、英数字以外の文字は含めないでください。英数字以外の文字はスペースとして解釈され、単語が複数の単語に分割されます。たとえば、"and/or" は、2 つの単語 "and" と "or" として解釈されます。ストップリスト単語の最大数は 7999 です。

DROP STOPLIST 句 – この句を使用して、テキスト設定オブジェクトのストップリストを削除します。

MINIMUM TERM LENGTH 句 – **TEXT** インデックスに含める単語の最小長 (文字数) を指定します。**NGRAMTEXT** インデックスを使用する場合は、**MINIMUM TERM LENGTH** 句に指定されている値は無視されます。

この設定よりも短い単語は、**TEXT** インデックスの構築または更新時に無視されます。このオプションの値は 0 よりも大きい値である必要があります。このオプションに **MAXIMUM TERM LENGTH** よりも大きい値を設定した場合、**MAXIMUM TERM LENGTH** の値は、新しい **MINIMUM TERM LENGTH** の値と同じになるように自動的に調整されます。

MAXIMUM TERM LENGTH 句 – **GENERICTEXT** インデックスの場合、**TEXT** インデックスに含める単語の最大長 (文字数) です。この設定よりも長い単語は、**TEXT** インデックスの構築または更新時に無視されます。

MAXIMUM TERM LENGTH の値は 60 以下である必要があります。このオプションに **MINIMUM TERM LENGTH** よりも小さい値を設定した場合、**MINIMUM TERM LENGTH** の値は、新しい **MAXIMUM TERM LENGTH** の値と同じになるように自動的に調整されます。

TERM BREAKER 句 – カラム値を単語に分割するために使用するアルゴリズムの名前を指定します。**IN SYSTEM** テーブルでの選択肢は、**GENERIC** (デフォルト) または **NGRAM** です。**GENERIC** アルゴリズムでは、英数字以外の文字で区切られた 1 つ以上の英数字の文字列は、1 つの単語として扱われます。

NGRAM アルゴリズムでは、文字列が n-gram に分割されます。n-gram は、大きな文字列の n 文字の部分文字列です。**NGRAM** 単語分割が必要となるのは、ファジー (近接) 一致の場合、または、スペースもしくは英数字以外の文字を使用せずに単語を区切るドキュメントの場合です。**NGRAM** は **IN SYSTEM** テーブルでのみサポートされています。

NGRAM 単語分割は **TEXT** インデックスに構築されます。そのため、**NGRAM** インデックスまたは **GENERICTEXT** インデックスを使用するかどうかを定義するには、テキスト設定オブジェクトの設定を使用します。

TERM BREAKER には、**EXTERNAL NAME** とライブラリのエントリ・ポイントを使用して、外部の単語分割ライブラリの指定を含めることができます。

DROP PREFILTER 句 – 外部のプレフィルタを削除し、**ISYSTEXTCONFIG** テーブルのプレフィルタ・カラムに NULL を設定します。

PREFILTER EXTERNAL NAME 句 – 外部ベンダによって提供される外部のプレフィルタ・ライブラリのエントリ・ポイントとライブラリ名を指定します。

関連する動作：

- オートコミット。

パーミッション

テキスト設定オブジェクトを変更して、外部プレフィルタまたは単語分割のための外部ライブラリおよび関数を指定するには、ユーザに DBA 権限がある必要があります。

テキスト設定に対するその他すべての変更は、設定オブジェクトの所有者か、DBA 権限を持つユーザが実行できます。

ALTER TEXT INDEX 文

TEXT・インデックスの定義を変更します。

構文

```
ALTER TEXT INDEX [ owner. ]text-index-name
             ON [ owner.
             ]table-name
  alter-clause

alter-clause :
   rename-object | move-object

rename-object :
   RENAME { AS | TO } new-name

move-object:
   MOVE TO  dbspace-name
```

例

- **– TEXT** インデックスの MyTextIndex を作成し、**IMMEDIATE REFRESH** として定義します。**TEXT** インデックスの名前を Text_index_daily に変更し、この **TEXT** インデックスを tispace という名前の DB 領域に移動します。

```
CREATE TEXT INDEX MyTextIndex ON Customers ( CompanyName )
IMMEDIATE REFRESH;
```

```
ALTER TEXT INDEX MyTextIndex ON Customers RENAME AS
Text_index_daily;
```

```
ALTER TEXT INDEX Text_Index_Daily ON Customers MOVE TO tispace;
```

使用法

TEXT インデックスの名前を変更したり移動したりするには、**ALTER TEXT INDEX** を使用します。

RENAME 句 – **TEXT** インデックスの名前を変更します。

MOVE 句 – **TEXT** インデックスを、指定の DB 領域に移動します。

関連する動作：

- オートコミット。

パーミッション

インデックスの名前を変更するには、基になるテーブルの所有者であるか、DBA 権限または REFERENCES パーミッションを持っている必要があります。

TEXT インデックスを移動するには、DBA 権限または SPACE ADMIN 権限を持っているか、テーブル所有者であり、かつ DB 領域に対する CREATE パーミッションを持っている必要があります。

CREATE TEXT CONFIGURATION 文

テキスト設定オブジェクトを作成します。

構文

```
CREATE TEXT CONFIGURATION [ owner.]new-config-name
  FROM [ owner.]existing-config-name
```

例

- – default_char テキスト設定オブジェクトを使用して、テキスト設定オブジェクト max_term_sixteen を作成します。その後で **ALTER TEXT CONFIGURATION** を使用して、max_term_sixteen の単語の最大長を 16 に変更します。

```
CREATE TEXT CONFIGURATION max_term_sixteen FROM default_char;
```

```
ALTER TEXT CONFIGURATION max_term_sixteen MAXIMUM TERM LENGTH 16;
```

使用法

テキスト設定オブジェクトを作成するには、**CREATE TEXT CONFIGURATION** を使用します。

別のテキスト設定オブジェクトをテンプレートとして使用してテキスト設定オブジェクトを作成し、必要に応じて **ALTER TEXT CONFIGURATION** 文を使用してオプションを変更します。

データベース内のすべてのテキスト設定オブジェクトとその設定のリストを表示するには、**SYSTEXTCONFIG** システム・ビューに対してクエリを実行します。

FROM 句 – 新しいテキスト設定オブジェクトを作成するためのテンプレートとして使用するテキスト設定オブジェクトの名前を指定します。デフォルトのテキスト設定オブジェクトの名前は、default_char と default_nchar です。default_char は Sybase IQ テーブルでのみサポートされており、default_nchar は SQL Anywhere テーブルでのみサポートされています。

関連する動作：

- オートコミット。

パーミッション

DBA 権限または RESOURCE 権限が必要です。

すべてのテキスト設定オブジェクトに PUBLIC アクセスが設定されています。**TEXT** インデックスを作成するパーミッションを持つユーザは、任意のテキスト設定オブジェクトを使用できます。

CREATE TEXT INDEX 文

TEXT インデックスを作成します。

構文

```
CREATE TEXT INDEX text-index-name
 ON [ owner.]table-name( column-name, ...)
   [ IN dbspace-name ]
   [ CONFIGURATION [ owner.]text-configuration-name]
   [  IMMEDIATE REFRESH ]
```

例

- – `max_term_sixteen` テキスト設定オブジェクトを使用して、`iqdemo` データベース内の `Customers` テーブルの `CompanyName` カラムに、**TEXT** インデックス `myTxtIdx` を作成します。

```
CREATE TEXT INDEX myTxtIdx ON Customers (CompanyName );
```

```
CONFIGURATION max_term_sixteen;
```

使用法

TEXT インデックスを作成して、使用するテキスト設定オブジェクトを指定するには、**CREATE TEXT INDEX** を使用します。

ビューまたはテンポラリ・テーブルには **TEXT** インデックスを作成できません。IN SYSTEM マテリアライズド・ビューにも **TEXT** インデックスを作成できません。

TEXT インデックスは、ジョイン・インデックス・テーブルには複写されません。**TEXT** インデックスは、ジョイン・インデックスの一部となっているテーブルのカラムに作成できます。

BEGIN PARALLEL IQ…END PARALLEL IQ 文では、**CREATE TEXT INDEX** はサポートされません。

ON 句 – **TEXT** インデックスを構築するテーブルとカラムを指定します。

IN 句 – **TEXT** インデックスを格納する DB 領域を指定します。この句を指定しない場合、**TEXT** インデックスは、基になるテーブルと同じ DB 領域に作成されます。

CONFIGURATION 句 – **TEXT** インデックスの作成時に使用するテキスト設定オブジェクトを指定します。この句を指定しない場合、`default_char` テキスト設定オブジェクトが使用されます。

REFRESH 句 – **IMMEDIATE REFRESH** がデフォルト値として使用され、Sybase IQ のテーブルではこの値のみが許可されます。**IMMEDIATE REFRESH** を指定すると、基になるテーブルでの変更により **TEXT** インデックスのデータが影響を受けるたびに **TEXT** インデックスが更新されます。

IMMEDIATE REFRESH TEXT インデックスは、作成時に値が設定され、元になるカラムのデータが変更されるたびに更新されます。**TEXT** インデックスが作成された後は、**IMMEDIATE REFRESH** に変更することも、**IMMEDIATE REFRESH** から変更することもできません。

関連する動作：

- オートコミット。

パーミッション

基になるテーブルの所有者であるか、DBA 権限または REFERENCES パーミッションを持っている必要があります。

DB 領域に対する CREATE パーミッションを持っている必要があります。

DROP TEXT CONFIGURATION 文

テキスト設定オブジェクトを削除します。

構文

```
DROP TEXT CONFIGURATION   [ owner.]text-config-name
```

例

- – `mytextconfig` テキスト設定オブジェクトを作成して、削除します。

```
CREATE TEXT CONFIGURATION mytextconfig FROM default_char;
DROP TEXT CONFIGURATION mytextconfig;
```

使用法

テキスト設定オブジェクトを削除するには、**DROP TEXT CONFIGURATION** を使用します。

依存する **TEXT** インデックスがあるテキスト設定オブジェクトを削除しようとすると、エラーが発生します。テキスト設定オブジェクトを削除する前に、依存する **TEXT** インデックスを削除する必要があります。

テキスト設定オブジェクトは、**ISYSTEXTCONFIG** システム・テーブルに格納されます。

関連する動作：

- オートコミット。

パーミッション

テキスト設定オブジェクトの所有者であるか、DBA 権限を持っている必要があります。

DROP TEXT INDEX 文

データベースから **TEXT** インデックスを削除します。

構文

```
DROP TEXT INDEX text-index-name
   ON [ owner ] table-name
```

例

- – TextIdx **TEXT** インデックスを作成して、削除します。

```
CREATE TEXT INDEX TextIdx ON Customers ( Street );
DROP TEXT INDEX TextIdx ON Customers;
```

使用法

データベースから **TEXT** インデックスを削除するには、**DROP TEXT INDEX** を使用します。

ON 句 – **TEXT** インデックスが構築されているテーブルを指定します。

テキスト設定オブジェクトを削除する前に、依存する **TEXT** インデックスを削除する必要があります。

関連する動作：

- オートコミット。

パーミッション

基になるテーブルの所有者であるか、DBA 権限または REFERENCES パーミッションを持っている必要があります。

関数のサポート

LONG BINARY データ型と LONG VARCHAR データ型をサポートする Sybase IQ 関数について説明します。

ラージ・オブジェクト・データをサポートする関数の概要

ラージ・オブジェクト・データ型と変数に対する関数のサポートの概要

表「LOB データ型と変数に対する関数のサポート」は、LONG BINARY (BLOB) データ型、LONG VARCHAR (CLOB) データ型、LONG BINARY 変数、LONG VARCHAR 変数に対する関数のサポートをまとめたものです。

この表に記載されている関数の他に、**BFILE** 関数を使用して LOB データを抽出できます。「ラージ・オブジェクト・データのエクスポート」を参照してください。

ユーザ定義のスカラ関数および集合関数は、入力パラメータとしてラージ・オブジェクト・データ型をサポートしています。「ラージ・オブジェクト・カラムのユーザ定義関数のサポート」を参照してください。

表 18 : LOB データ型と変数に対する関数のサポート

関数	BLOB データに対するサポート	BLOB 変数に対するサポート	CLOB データに対するサポート	CLOB 変数に対するサポート
BIT_LENGTH()	あり	あり	あり	あり
BYTE_LENGTH()	あり*	あり*	あり*	あり*
BYTE_LENGTH64()	あり	あり	あり	あり
BYTE_SUBSTR()	あり	あり	あり	あり
BYTE_SUBSTR64()	あり	あり	あり	あり
CHAR_LENGTH()	なし	なし	あり	あり
CHAR_LENGTH64()	なし	なし	あり	あり
CHARINDEX()	あり	あり	あり	あり
LOCATE()	あり	あり	あり	あり
OCTET_LENGTH()	あり	あり	あり	あり
PATINDEX()	なし	なし	あり	あり

関数	BLOB データに対するサポート	BLOB 変数に対するサポート	CLOB データに対するサポート	CLOB 変数に対するサポート
SUBSTR() / SUBSTRING()	なし	なし	あり	あり
SUBSTRING64()	あり	あり	あり	あり

*BYTE_LENGTH 関数が LONG BINARY と LONG VARCHAR のカラムと変数の両方をサポートするのは、クエリの戻り値が 2GB 未満の場合のみです。返された LONG BINARY データまたは LONG VARCHAR データのバイト長が 2GB を超える場合は、BYTE_LENGTH によって、BYTE_LENGTH64 関数を使用する必要があることを示すエラーが返されます。

これらの関数の詳細と使用例については、『リファレンス：ビルディング・ブロック、テーブル、およびプロシージャ』の「SQL 関数」を参照してください。

参照：

- ラージ・オブジェクト・カラムのユーザ定義関数のサポート (91 ページ)
- ラージ・オブジェクト・データのエクスポート (55 ページ)

BIT_LENGTH 関数

BIT_LENGTH 関数は、ラージ・オブジェクト・カラムまたは変数のパラメータのビット長を表す符号なし 64 ビット値を返します。引数が NULL の場合、BIT_LENGTH は NULL を返します。

構文

```
BIT_LENGTH( large-object-column )
```

パラメータ

large-object-column – LONG VARCHAR カラムまたは変数、または LONG BINARY カラムまたは変数の名前です。

使用法

BIT_LENGTH は、すべての Sybase IQ データ型、任意のデータ・サイズの LONG BINARY 変数、任意のデータ・サイズの LONG VARCHAR 変数をサポートします。現在、SQL 変数で保持できる最大の長さは 2GB - 1 です。

BYTE_LENGTH 関数

BYTE_LENGTH 関数は、文字列のバイト数を返します。

使用法
BYTE_LENGTH 関数が LONG BINARY と LONG VARCHAR のカラムと変数の両方をサポートするのは、クエリの戻り値が 2GB 未満の場合のみです。返された LONG BINARY データまたは LONG VARCHAR データのバイト長が 2GB 以上の場合は、**BYTE_LENGTH** によって、**BYTE_LENGTH64** 関数を使用する必要があることを示すエラーが返されます。

BYTE_LENGTH 関数の構文と使用法については、『リファレンス：ビルディング・ブロック、テーブル、およびプロシージャ』の「SQL 関数」>「アルファベット順の関数リスト」>「BYTE_LENGTH 関数 [文字列]」を参照してください。

BYTE_LENGTH64 関数

BYTE_LENGTH64 関数は、ラージ・オブジェクト・カラムまたは変数のパラメータのバイト長を表す符号なし 64 ビット値を返します。

構文
```
BYTE_LENGTH64(  large-object-column )
```

パラメータ
large-object-column – LONG VARCHAR カラムまたは変数、または LONG BINARY カラムまたは変数の名前です。

使用法
BYTE_LENGTH64 関数は、LONG BINARY カラム、LONG VARCHAR カラム、任意のデータ・サイズの LONG BINARY 変数、任意のデータ・サイズの LONG VARCHAR 変数をサポートしています。 現在、SQL 変数で保持できる最大の長さは 2GB - 1 です。

BYTE_SUBSTR64 関数と BYTE_SUBSTR 関数

BYTE_SUBSTR64 関数と **BYTE_SUBSTR** 関数は、ラージ・オブジェクト・カラムまたは変数のパラメータのバイト部分文字列を返します。

構文

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

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

パラメータ

large-object-column – LONG VARCHAR カラムまたは変数、または LONG BINARY カラムまたは変数の名前です。

start – 部分文字列の始まりを表す整数式です。正の整数は部分文字列が文字列の先頭から始まり、最初のバイトがポジション 1 であることを示します。負の整数は部分文字列が文字列の末尾から始まり、最後のバイトがポジション -1 であることを示します。

length – 部分文字列の長さを表す整数式です。正の値は、*start* ポジションを始点として返される部分文字列のバイト数を示します。負の値は、*start* ポジションを終点として返される部分文字列のバイト数を示します。

使用法

- **BYTE_LENGTH64** 関数、**BYTE_SUBSTR64** 関数、**BYTE_SUBSTR** 関数のネストされた演算では、ラージ・オブジェクト・カラムまたは変数はサポートされません。
- **BYTE_SUBSTR64** 関数と **BYTE_SUBSTR** 関数は、LONG BINARY カラム、LONG VARCHAR カラム、任意のデータ・サイズの LONG BINARY 変数、任意のデータ・サイズの LONG VARCHAR 変数をサポートしています。現在、SQL 変数で保持できる最大の長さは 2GB - 1 です。

CHAR_LENGTH 関数

CHAR_LENGTH 関数は、LONG VARCHAR カラムまたは変数のパラメータの文字長 (後続ブランクを含む) を表す符号付き 32 ビット値を返します。

構文

```
CHAR_LENGTH( long-varchar-object )
```

パラメータ
long-varchar-object – LONG VARCHAR カラムまたは LONG VARCHAR 変数の名前です。

使用法

- **CHAR_LENGTH** は、LONG VARCHAR カラムと、任意のデータ・サイズの LONG VARCHAR 変数をサポートします。現在、SQL 変数で保持できる最大の長さは 2GB - 1 です。
- 引数が NULL の場合、**CHAR_LENGTH** は NULL を返します。
- 文字長が 2GB - 1 (2147483647) を超える場合はエラーが返されます。

CHAR_LENGTH64 関数

CHAR_LENGTH64 関数は、LONG VARCHAR カラムまたは変数のパラメータの文字長 (後続ブランクを含む) を表す符号なし 64 ビット値を返します。

構文
```
CHAR_LENGTH64( long-varchar-object )
```

パラメータ
long-varchar-object – テーブル内の LONG VARCHAR カラムまたは LONG VARCHAR 変数の名前です。

使用法

- **CHAR_LENGTH64** は、LONG VARCHAR カラムと、任意のデータ・サイズの LONG VARCHAR 変数をサポートします。現在、SQL 変数で保持できる最大の長さは 2GB - 1 です。
- 引数が NULL の場合、**CHAR_LENGTH64** は NULL を返します。

CHARINDEX 関数

CHARINDEX 関数は、指定された文字列がラージ・オブジェクト・カラムまたは変数のパラメータで最初に出現する位置を格納した 64 ビット符号付き整数を返します。**CHARINDEX** は、CHAR カラムと VARCHAR カラムの場合、32 ビット符号付き整数で位置を返します。

構文
```
CHARINDEX( string-expression, large -object-column )
```

パラメータ
string-expression – 検索対象の最大 255 バイトの文字列です。

large-object-column – LONG VARCHAR カラムまたは変数、または LONG BINARY カラムまたは変数の名前です。

使用法

- **CHARINDEX** 関数で返されるか指定される位置またはオフセットはすべて、常に文字オフセットであり、マルチバイト・データの場合はバイト・オフセットとは異なることがあります。
- 検索されるラージ・オブジェクト・セルに、*string-expression* のインスタンスが 2 つ以上含まれる場合、**CHARINDEX** は最初のインスタンスの位置だけを返します。
- カラムに文字列が含まれない場合、**CHARINDEX** 関数はゼロ (0) を返します。
- 長さが 255 バイトを超える文字列を検索すると、NULL が返されます。
- 長さが 0 の文字列を検索すると、1 が返されます。
- 引数のどれか 1 つでも NULL の場合、結果は NULL になります。
- **CHARINDEX** は、LONG VARCHAR カラムと LONG BINARY カラム、および任意のデータ・サイズの LONG VARCHAR 変数と LONG BINARY 変数の検索をサポートします。現在、SQL 変数で保持できる最大の長さは 2GB - 1 です。

『リファレンス：ビルディング・ブロック、テーブル、およびプロシージャ』の「SQL 関数」>「アルファベット順の関数リスト」>「CHARINDEX 関数 [文字列]」を参照してください。

LOCATE 関数

LOCATE 関数は、ラージ・オブジェクト・カラムまたは変数のパラメータでの指定された文字列の位置を格納した 64 ビット符号付き整数を返します。**LOCATE** は、CHAR カラムと VARCHAR カラムの場合、32 ビット符号付き整数で位置を返します。

構文

```
LOCATE( large-object-column, string-expression
[, numeric-expression ] )
```

パラメータ
large-object-column – 検索対象の LONG VARCHAR カラムまたは変数、または LONG BINARY カラムまたは変数の名前です。

string-expression – 検索対象の最大 255 バイトの文字列です。

numeric-expression – 文字列内で検索を開始する文字位置またはオフセットです。*numeric-expression* は LONG VARCHAR カラムと LONG BINARY カラムの場合は 64 ビット符号付き整数であり、CHAR カラム、VARCHAR カラム、BINARY カラムの場合は 32 ビット符号付き整数です。最初の文字の位置は 1 です。開始オフセットが負の場合、**LOCATE** は、最初ではなく最後にマッチする文字列のオフセットを返します。負のオフセットは、文字列の末尾から何文字を検索から除外するかを示します。除外する文字数は、(-1 * オフセット) - 1 で計算します。

使用法

- **LOCATE** 関数で返されるか指定される位置またはオフセットはすべて、常に文字オフセットであり、マルチバイト・データの場合はバイト・オフセットとは異なることがあります。
- 検索されるラージ・オブジェクト・セルに、文字列のインスタンスが 2 つ以上含まれる場合、次のように処理されます。
  - *numeric-expression* を指定した場合、**LOCATE** は文字列内のそのオフセット位置から検索を開始します。
  - *numeric-expression* を指定しなかった場合、**LOCATE** は、指定した文字列の最初のインスタンスの位置だけを返します。
- カラムに文字列が含まれない場合、**LOCATE** はゼロ (0) を返します。
- 長さが 255 バイトを超える文字列を検索すると、NULL が返されます。
- 長さが 0 の文字列を検索すると、1 が返されます。
- 引数のどれか 1 つでも NULL の場合、結果は NULL になります。
- **LOCATE** は、LONG VARCHAR カラムと LONG BINARY カラム、および任意のデータ・サイズの LONG VARCHAR 変数と LONG BINARY 変数の検索をサポートします。現在、SQL 変数で保持できる最大の長さは 2GB - 1 です。

『リファレンス：ビルディング・ブロック、テーブル、およびプロシージャ』の「SQL 関数」>「アルファベット順の関数リスト」>「LOCATE 関数 [文字列]」を参照してください。

OCTET_LENGTH 関数

OCTET_LENGTH 関数は、ラージ・オブジェクト・カラムまたは変数のパラメータのバイト長を表す符号なし 64 ビット値を返します。

構文

```
OCTET_LENGTH( column-name )
```

パラメータ
large-object-column – LONG VARCHAR カラムまたは変数、または LONG BINARY カラムまたは変数の名前です。

使用法

- 引数が NULL の場合、**OCTET_LENGTH** は NULL を返します。
- **OCTET_LENGTH** は、すべての Sybase IQ データ型、任意のデータ・サイズの LONG VARCHAR 変数、任意のデータ・サイズの LONG BINARY 変数をサポートします。現在、SQL 変数で保持できる最大の長さは 2GB - 1 です。

PATINDEX 関数

PATINDEX 関数は、指定されたパターンが LONG VARCHAR カラムまたは変数で最初に出現する位置を格納した 64 ビット符号なし整数を返します。**PATINDEX** は、CHAR カラムと VARCHAR カラムの場合、32 ビット符号なし整数で位置を返します。

構文
```
PATINDEX( '%pattern%', long-varchar-column )
```

パラメータ
pattern – 検索するパターンです。パターンは、ワイルドカードを使用して 126 バイトまでの文字列を指定してください。先頭の % ワイルドカードを省略すると、**PATINDEX** はパターンがカラム値の最初に出現する場合は 1 を、そうでない場合は 0 を返します。同様に、末尾の % ワイルドカードを省略した場合、パターンがカラム値の最後に出現する必要があります。パターンに使用するワイルドカードは、**LIKE** での比較の場合と同じです。

パターンにワイルドカード (パーセント (%) またはアンダースコア (_)) を使用しない場合は、255 バイトの長さまで指定できます。

long-varchar-column – LONG VARCHAR カラムまたは変数の名前です。

使用法

- **PATINDEX** 関数で返されるか指定される位置またはオフセットはすべて、常に文字オフセットであり、マルチバイト・データの場合はバイト・オフセットとは異なることがあります。
- 文字列パターンが検索対象の LONG VARCHAR セルに 2 つ以上含まれる場合、**PATINDEX** は最初の文字列の位置だけを返します。

- カラムに文字パターンが含まれない場合、**PATINDEX** はゼロ (0) を返します。
- 長さが 126 バイトを超えるパターンを検索すると、NULL が返されます。
- 長さが 0 のパターンを検索すると、1 が返されます。
- 引数のどれか 1 つでも NULL の場合、結果はゼロ (0) となります。
- **PATINDEX** は、任意のデータ・サイズの LONG VARCHAR 変数をサポートします。現時点では、SQL 変数で保持できる最大長は 2GB - 1 です。**PATINDEX** は、LONG BINARY 変数、または LONG BINARY カラムの検索をサポートしていません。

『リファレンス：ビルディング・ブロック、テーブル、およびプロシージャ』の「SQL 関数」>「アルファベット順の関数リスト」>「PATINDEX 関数 [文字列]」と『リファレンス：ビルディング・ブロック、テーブル、およびプロシージャ』の「SQL 言語の要素」>「検索条件」>「LIKE 条件」を参照してください。

SUBSTRING 関数

SUBSTRING 関数は、LONG VARCHAR カラムまたは変数パラメータの可変長文字列を返します。引数のいずれか 1 つでも NULL の場合、**SUBSTRING** は NULL を返します。

構文

```
{ SUBSTRING | SUBSTR } ( long-varchar-column, start [, length ] )
```

パラメータ

long-varchar-column – LONG VARCHAR カラムまたは変数の名前です。

start – 部分文字列の始まりを表す整数式です。正の整数は部分文字列が文字列の先頭から始まり、最初の文字がポジション 1 であることを示します。負の整数は部分文字列が文字列の末尾から始まり、最後の文字がポジション -1 であることを示します。

length – 部分文字列の文字長を表す整数式です。正の値は、*start* ポジションを始点として返される文字数を示します。負の値は、*start* ポジションを終点として返される文字数を示します。

使用法

SUBSTRING は、任意のデータ・サイズの LONG VARCHAR 変数をサポートします。現時点では、SQL 変数で保持できる最大長は 2GB - 1 です。**SUBSTRING** は、LONG BINARY 変数、または LONG BINARY カラムの検索をサポートしていません。

SUBSTRING64 関数

SUBSTRING64 関数は、ラージ・オブジェクト・カラムまたは変数パラメータの可変長文字列を返します。

構文

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

パラメータ

large-object-column – LONG VARCHAR カラムまたは変数、または LONG BINARY カラムまたは変数の名前です。

start – 部分文字列の開始を表す 8 バイト整数です。**SUBSTRING64** では、*start* オフセット値が負またはゼロの場合、文字列の左側に非文字が埋め込まれているものと解釈します。最初の文字はポジション 1 から始まります。

length – 部分文字列の長さを表す 8 バイト整数式です。*length* の値が負の場合、エラーが返されます。

例

col1 という名前のカラムに文字列 ("ABCDEFG") が格納されている場合、**SUBSTRING64** は次の値を返します。

SUBSTRING64(col1, 2, 4) は文字列 "BCDE" を返します。

SUBSTRING64(col1, 1, 3) は文字列 "ABC" を返します。

SUBSTRING64(col1, 0, 3) は文字列 "AB" を返します。

SUBSTRING64(col1, -1, 3) は文字列 "A" を返します。

使用法

- 引数のいずれか 1 つでも NULL の場合、**SUBSTRING64** は NULL を返します。
- **SUBSTRING64** 関数、**SUBSTRING** 関数、**SUBSTR** 関数、**BYTE_SUBSTR** 関数、**BYTE_SUBSTR64** 関数のネストされた演算では、ラージ・オブジェクト・カラムまたは変数はサポートされません。
- **SUBSTRING64** は、LONG VARCHAR カラムと LONG BINARY カラム、および任意のデータ・サイズの LONG VARCHAR 変数と LONG BINARY 変数の検索をサポートします。現在、SQL 変数で保持できる最大の長さは 2GB - 1 です。

ラージ・オブジェクト・カラムの集合関数のサポート

LONG BINARY カラムと LONG VARCHAR カラムでサポートされている唯一の集合関数は、**COUNT (*)** です。

COUNT DISTINCT パラメータはサポートされていません。**MIN** 集合関数、**MAX** 集合関数、**AVG** 集合関数、または **SUM** 集合関数に LONG BINARY カラムまたは LONG VARCHAR カラムを指定すると、エラーが返されます。

ラージ・オブジェクト・カラムのユーザ定義関数のサポート

ユーザ定義のスカラ関数と集合関数は、入力パラメータとして 4GB (ギガバイト) までのラージ・オブジェクト (LOB) データ型の LONG VARCHAR (CLOB) と LONG BINARY (BLOB) をサポートしています。LOB データ型は、出力パラメータとしてはサポートされていません。

ユーザ定義関数のサポートには、別途ライセンスが必要な Sybase IQ オプションが必要です。『ユーザ定義関数ガイド』を参照してください。

エラー・メッセージと警告メッセージ

LONG BINARY カラムと LONG VARCHAR カラムなどの非構造化データの操作で返される可能性のあるエラー・メッセージと警告メッセージの参照情報を示します。

エラー 1000195

ロード指定 '%2' はデータ型 '%3' を含むカラムに対してのみ有効です。 %1

項目	値
SQLCode	-1000195L
定数	EMSG_BINARYFILE
SQLState	QDB95
ODBC 2 State	ERROR
ODBC 3 State	ERROR
Sybase エラー・コード	20855
重大度コード	14
Parameter 1	例外の発生場所
Parameter 2	ロード仕様の種類
Parameter 3	カラムのデータ型

考えられる原因
LOAD TABLE 文のロード指定が有効なのは、ここで示されるデータ型のカラムだけです。

エラー 1000198

データ型 %2 のカラムがあるテーブルを含むジョイン・テーブルを作成できません。 %1

項目	値
SQLCode	-1000198L
定数	EMSG_CANNOT_CREATE_JOIN_INDEX

項目	値
SQLState	QDB98
ODBC 2 State	ERROR
ODBC 3 State	ERROR
Sybase エラー・コード	20858
重大度コード	14
Parameter 1	例外の発生場所
Parameter 2	カラムのデータ型

考えられる原因
このエラーは、1 つまたは複数の LONG VARCHAR データ型カラムまたは LONG BINARY データ型カラムを持つテーブルにジョイン・インデックスを作成しようとすると報告されます。

JOIN INDEX 関数はほとんどのデータ型に対応しています。しかし、一部には、この機能が対応していないデータ型 (LONG BINARY、LONG VARCHAR など) もあります。

エラー 1000332

"Odd length of binary data value detected on column %2 %1"

項目	値
SQLCode	-1000332L
定数	EMSG_ODDNUMBER_NIBBLES
SQLState	QDD20
ODBC 2 State	ERROR
ODBC 3 State	ERROR
Sybase エラー・コード	21206
重大度コード	14
Parameter 1	例外の発生場所
Parameter 2	column name

考えられる原因
16 進数形式のバイナリ・データをプライマリ・ロード・ファイルから LONG BINARY カラムにロードする場合、Sybase IQ では、16 進数の合計数は偶数である必要があります。

セルの値に奇数個の 16 進数字が含まれている場合に、このエラーが報告されます。LONG BINARY をロードする入力ファイルには、必ず偶数個の 16 進数字が含まれている必要があります。

エラー 1001013

無効なデータ型比較 %2、%1

項目	値
SQLCode	-1001013L
定数	EMSG_TYPECOMPAREERROR
SQLState	QFA13
ODBC 2 State	ERROR
ODBC 3 State	ERROR
Sybase エラー・コード	20522
重大度コード	14
Parameter 1	例外の発生場所

考えられる原因
LIKE 述部を使用して、LONG BINARY カラムでパターンを検索しようとすると、このエラーが報告されます。

LIKE 述部は、**LONG BINARY** (BLOB) カラムではサポートされていません。

エラー 1001051

クエリは 2GB を超える %2 データを返します。 %3 を使用してください %1

項目	値
SQLCode	-1001051L
定数	EMSG_LOB_OVER_2G_W_ARG

項目	値
SQLState	QFA47
ODBC 2 State	ERROR
ODBC 3 State	ERROR
Sybase エラー・コード	21097
重大度コード	14
Parameter 1	SA 解析ソース・コード行
Parameter 2	推奨される関数
Parameter 3	long binary または long varchar データ型

考えられる原因
このエラーは、クエリが 2 ギガバイトを超える `LONG BINARY` 値または `LONG VARCHAR` 値を返そうとすると報告されます。

エラー 1001052

パラメータ %2 は long binary／varchar 型である必要があります。 %3 %1

項目	値
SQLCode	-1001052L
定数	EMSG_ONLY_SUPPORT_LOB_W_ARG
SQLState	QFA48
ODBC 2 State	ERROR
ODBC 3 State	ERROR
Sybase エラー・コード	21098
重大度コード	14
Parameter 1	SA 解析ソース・コード行
Parameter 2	LOB 引数名
Parameter 3	推奨される関数の名前

考えられる原因
このエラーがレポートされるのは、ラージ・オブジェクト (LOB) 関数のパラメータとして無効なデータ型が使用された場合です。

エラー 1001053

関数 %2 に対するパラメータの数が間違っています %1

項目	値
SQLCode	-1001053L
定数	EMSG_WRONG_NUM_PARAMS_W_ARG
SQLState	QFA49
ODBC 2 State	ERROR
ODBC 3 State	ERROR
Sybase エラー・コード	21099
重大度コード	14
Parameter 1	SA 解析ソース・コード行
Parameter 2	関数名

考えられる原因
このエラーがレポートされるのは、ラージ・オブジェクト (LOB) 関数に渡された引数の個数が不正な場合です。

エラー 1001054

long binary／varchar カラムは ORDER/GROUP by 句または集合関数内で指定できません。 %1

項目	値
SQLCode	-1001054L
定数	EMSG_LOB_NOT_ALLOWED_GROUP
SQLState	QFA50
ODBC 2 State	ERROR
ODBC 3 State	ERROR
Sybase エラー・コード	21100
重大度コード	14

項目	値
Parameter 1	例外の発生場所

考えられる原因
このエラーは、**ORDER BY** 句、**GROUP BY** 句、または集約関連の句に LONG BINARY カラムが指定されていた場合に報告されます。

警告 1001055

%1 カラム、%2、%3、ロー ID %4 のロード中にエラーが発生しました。

項目	値
SQLCode	1001055L
定数	EMSG_LOB_LOAD_ERROR_WARN
SQLState	QFA51
ODBC 2 State	OK
ODBC 3 State	OK
Sybase エラー・コード	21101
重大度コード	10
Parameter 1	long binary または long varchar データ型
Parameter 2	FP インデックス名
Parameter 3	セカンダリ・ファイル名
Parameter 4	ロー ID

考えられる原因
この警告メッセージは、ロード操作中に LONG BINARY 型または LONG VARCHAR 型のセカンダリ・ファイルを開こう (または読み込もう) としてエラーが発生すると返されます。

この警告メッセージは、SECONDARY_FILE_ERROR オプションがオフの状態でエラーが発生すると、サーバ・ログと IQ メッセージ・ファイルに返されます。

警告 1001056

%3 の %1 カラム %2 の抽出中にエラーが発生しました。

項目	値
SQLCode	1001056L
定数	EMSG_LOB_EXTRACT_ERROR_WARN
SQLState	QFA52
ODBC 2 State	OK
ODBC 3 State	OK
Sybase エラー・コード	21102
重大度コード	10
Parameter 1	long binary または long varchar データ型
Parameter 2	FP インデックス名
Parameter 3	セカンダリ・ファイル名

考えられる原因
この警告メッセージは、`LONG BINARY` カラムまたは `LONG VARCHAR` カラムを抽出しようとしてエラーが発生すると返されます。

この警告メッセージは、`SECONDARY_FILE_ERROR` オプションがオフの状態でエラーが発生すると、サーバ・ログと IQ メッセージ・ファイルに返されます。

エラー 1001057

%2 カラムを抽出するには、BFILE() を使用する必要があります。 %1

項目	値
SQLCode	-1001057L
定数	EMSG_LOB_EXTRACT_USE_BFILE
SQLState	QFA53
ODBC 2 State	ERROR
ODBC 3 State	ERROR

項目	値
Sybase エラー・コード	21103
重大度コード	14
Parameter 1	例外の発生場所
Parameter 2	long binary または long varchar データ型

考えられる原因
このエラーは、LONG BINARY カラムまたは LONG VARCHAR カラムを対象とするクエリで、データベース・オプション TEMP_EXTRACT_NAME1 が ON に設定されているにもかかわらず **BFILE** 関数が指定されていなかった場合に報告されます。

エラー 1001058

セカンダリ・ファイル名 %2 が長すぎます。 %1

項目	値
SQLCode	-1001058L
定数	EMSG_LOB_SECONDARY_FILE_TOOLONG
SQLState	QFA54
ODBC 2 State	OK
ODBC 3 State	OK
Sybase エラー・コード	21104
重大度コード	14
Parameter 1	例外の発生場所
Parameter 2	セカンダリ・ファイル名

考えられる原因
このエラーは、**LOAD TABLE** セカンダリ・ファイルのパス名がオペレーティング・システムのパス名の長さ制限を超えている場合に報告されます。

このエラーの発生時に実行されるアクションは、SECONDARY_FILE_ERROR データベース・オプションの値によって異なります。

エラー 1009189

"Text document exceeds maximum number of terms. Support up to 4294967295 terms per document. %1"

項目	値
SQLCode	-1009189L
定数	EMSG_MAXTERM_ERROR
SQLState	QSB84
ODBC 2 State	ERROR
ODBC 3 State	ERROR
Sybase エラー・コード	21210
重大度コード	14
Parameter 1	例外の発生場所

考えられる原因
外部のプレフィルタ・ライブラリまたは単語分割ライブラリによるエラー。

エラー 1012030

long binary/varchar カラム '%2' において、データベース・ページ・サイズ (%3) は %4 よりも大きい必要がありま

項目	値
SQLCode	-1012030
定数	EMSG_CAT_PAGESIZETOOSMALL
SQLState	QUA30
ODBC 2 State	ERROR
ODBC 3 State	ERROR
Sybase エラー・コード	20953
重大度コード	14
Parameter 1	例外の発生場所
Parameter 2	カラム番号

項目	値
Parameter 3	要求ページ・サイズ
Parameter 4	最小許容ページ・サイズ

考えられる原因
データベースのページ・サイズが小さすぎるため、LONG BINARY カラムまたは LONG VARCHAR カラムを作成できません。

LONG BINARY カラムまたは LONG VARCHAR カラムを作成するのに必要なページ・サイズは 128K 以上です。

索引

A

B

C

D

E

F

I

L

M

N

O

P

S

T

W

あ

い

え

お

き

く

す

て

は

ふ

ま

め

ら

り

ろ