



IBM i

e-business および Web サービス

DB2 for i OmniFind テキスト検索サーバー

7.1





IBM i

e-business および Web サービス

DB2 for i OmniFind テキスト検索サーバー

7.1

ご注意!

本書および本書で紹介する製品をご使用になる前に、117ページの『特記事項』に記載されている情報をお読みください。

本書は、OmniFind Text Search Server for DB2 for i (製品番号 5733-OMF) のバージョン 1、リリース 2、モディフィケーション 0 に適用されます。また、改訂版で断りがない限り、それ以降のすべてのリリースおよびモディフィケーションに適用されます。このバージョンは、すべての RISC モデルで稼働するとは限りません。また CISC モデルでは稼働しません。

お客様の環境によっては、資料中の円記号がバックスラッシュと表示されたり、バックスラッシュが円記号と表示されたりする場合があります。

原典： IBM i
e-business and Web serving
OmniFind Text Search Server for DB2 for i
7.1

発行： 日本アイ・ビー・エム株式会社

担当： トランスレーション・サービス・センター

第1刷 2010.4

© Copyright IBM Corporation 2002, 2010.

DB2 for i OmniFind テキスト検索サーバー

DB2® for i OmniFind® テキスト検索サーバーを使用すれば、DB2 データベースに格納されている文書に対して、使い慣れたテキスト検索照会の条件を満たす SQL ステートメントを実行できます。

注: コーディング例を使用すると、115 ページの『コードに関するライセンス情報および特記事項』の条件に同意したものとみなされます。

IBM i 7.1 の新機能

DB2 for i OmniFind テキスト検索サーバーのトピック・コレクションの新しい情報や大きく変更された情報についてお読みください。

主な新機能を以下にまとめます。

- 拡張 50 ページの『XML 検索』
 - 58 ページの『検索での名前空間の使用』
 - 53 ページの『数値の比較』
 - 54 ページの『日付と日時の比較』
- DB2 の XML 列を対象にしたテキスト検索索引: 63 ページの『XML 検索の例』
- 69 ページの『テキスト検索索引をデータなしで保管して復元する操作』
- 12 ページの『サーバーの別名』
- 2 つの GUI インターフェース:
 - 93 ページの『IBM Systems Director Navigator for i の使用』
 - 99 ページの『System i ナビゲーターの使用』
- 88 ページの『高可用性』
- 89 ページの『パフォーマンスの分析』
- 92 ページの『トランザクションに関する注意点』

新規情報または変更情報の見分け方

技術上の変更が加えられた場所を見分けるのに役立つように、Information Center では以下のイメージを使用しています。

-  イメージにより、新規または変更された情報の開始点を示します。
-  イメージにより、新規または変更された情報の終了点を示します。

PDF ファイルでは、左マージンに新規および変更情報のリビジョン・バー (I) があります。

このリリースでの新機能または変更点についてのその他の情報は、「プログラム資料説明書」を参照してください。

DB2 for i OmniFind テキスト検索サーバーの PDF ファイル

このページを使用して、この情報の PDF を表示/印刷します。

この文書の PDF ファイルを表示/ダウンロードするには、DB2 for i OmniFind テキスト検索サーバー を
選択します。

その他の情報

以下の PDF ファイルも表示/印刷できます。

• Preparing for and Tuning the SQL Query Engine on DB2 for i5/OS®  (英語)

• SQL Performance Diagnosis on IBM® DB2 Universal Database™ for iSeries®  (英語)

PDF ファイルの保存

表示または印刷のために PDF をワークステーションに保存するには、以下のようになります。

1. ブラウザーで PDF を右クリックします (前述のリンクを右クリックします)。
2. PDF をローカルに保存するオプションをクリックします。
3. PDF を保存するディレクトリーに移動します。
4. 「保存」をクリックします。

Adobe® Reader のダウンロード

PDF ファイルを表示/印刷するには、Adobe Reader がシステムにインストールされている必要があります。
その製品は、Adobeのサイト (<http://get.adobe.com/reader/>)  から無償でダウンロードできます。

DB2 for i OmniFind テキスト検索サーバーの概要

DB2 for i では、DB2 データベースに格納されている文書の索引作成/検索エンジンとして OmniFind(r) テキスト検索サーバーを使用します。

OMNIFINDは、IBM i のユーザーが DB2 for i の表の列に格納されている非構造化テキストを検索するためのテキスト検索製品です。列には、単純な文字テキスト、XML 文書、各種のリッチ・テキスト文書 (PDF ファイルや DOC ファイルなど) を格納することができます。この製品のユーザーは、非構造化データの構文を解析して SQL 表などの構造化形式に変換しなくても、非構造化データの索引を生成できます。

OmniFind テキスト検索サーバーは、コンテキスト・ベースの検索エンジンです。ファジー検索機能にも対応しています。例えば、「mice」 (「mouse」の複数形) を検索することによって、「mice」または「mouse」が含まれている文書を検出できます。この検索エンジンは、言語コンテキストにも対応しています。例えば、英語とスペイン語の両方で「mice」と「mouse」がファジー検索の等価項目であることを認識できます。

さまざまなアプリケーションでこの機能を活用できます。人事情報のデータベースは、その好例です。応募者の履歴書をそれぞれの送信形式のままデータベースに格納できます。その後の検索操作で OmniFind を使用すれば、重要なスキルを持った応募者を探し出すことができます。

DB2 for i OmniFind テキスト検索サーバーの概要

DB2 for i OmniFind テキスト検索サーバーには、一連の管理ストアード・プロシージャと 2 つの組み込み関数 (CONTAINS と SCORE) が用意されています。それらの関数では、DB2 表に格納されている文書

で作成したテキスト索引を検索します。管理ストアード・プロシージャーでは、テキスト検索を有効/無効にしたり、テキスト索引の作成、更新、削除の操作を実行したりします。

テキスト索引は、以下のデータ・タイプの列で作成できます。

- CHAR
- VARCHAR
- CLOB
- BLOB
- DBCLOB
- GRAPHIC
- VARGRAPHIC
- BINARY
- VARBINARY
- XML

データとしては、プレーン・テキスト、HTML、XML、さまざまなリッチ・テキスト文書タイプ (PDF ファイルなど) を格納できます。そのデータがテキスト列から読み取られ、Unicode (CCSID 1208) に変換されてから、索引が生成されます。

テキスト索引は、DB2 の標準的な索引ではありません。保守作業が自動的に行われるわけではなく、ジャーナル処理も実行できません。標準的なバックアップ/リストア方式でバックアップすることもできません。テキスト索引は、テキスト検索サーバーで作成され、格納されます。

デフォルトでは、テキスト検索サーバーは、DB2 データベースにデータが格納されているのと同じシステムで作成されます。ただし、IBM i、Linux[®]、UNIX[®]、AIX[®]、Windows[®] のいずれかを実行する別のサーバーでテキスト検索サーバーを作成することも可能です。

テキスト検索サーバーには、列の各行から抽出した重要語のコレクションを格納します。テキスト検索サーバーとの通信では、TCP/IP 接続を使用します。

CONTAINS 関数と SCORE 関数は、DB2 for i に統合されている組み込み関数です。

DB2 for i では、DB2 データベースに格納されている文書の索引作成/検索エンジンとして OmniFind(r) Text Search Server を使用します。

OMNIFINDでは、複数のコレクションがサポートされています。1 つのコレクションに、1 つのテキスト検索索引と、解析、索引生成、検索に関するその索引固有のオプションを組み込みます。

OMNIFINDには、サーバーとテキスト索引を管理するためのグラフィカル・ユーザー・インターフェースがあります。

このテキスト検索サーバーには、一般的なタスクを実行するために使用できる SQL ストアード・プロシージャーとコマンド行ツールも用意されています。そのような一般的なタスクとして、テキスト検索サーバーの構成と管理、コレクションの同義語ディクショナリーの作成、問題の診断などがあります。

関連概念

17 ページの『テキスト検索の管理ストアード・プロシージャ』
一連の管理 SQL ストアード・プロシージャを使用して、テキスト検索機能の開始と停止の操作や、テキスト検索索引の作成、削除、更新の操作を実行できます。これらのプロシージャは、どの SQL インターフェースからでも呼び出せます。IBM i のコマンド行から CL コマンドを使用してこれらのプロシージャを呼び出すことはできません。

関連資料

40 ページの『CONTAINS』
CONTAINS 関数では、検索指数として指定する基準を使用して、テキスト検索索引を検索できます。この関数は、一致項目が見つかったかどうかを示す結果を返します。

43 ページの『SCORE』
SCORE 関数では、検索指数として指定する基準を使用して、テキスト検索索引を検索できます。この関数は、文書が照会内容に一致する比率を測定した関連度スコアを返します。

DB2 for i OmniFind テキスト検索サーバーをインストールするためのシステム要件

DB2 for i OmniFind テキスト検索サーバーをインストールする前に、すべてのハードウェア要件、ソフトウェア要件、オペレーティング・システム要件をシステムが満たしていることを確認してください。

DB2 for i OmniFind テキスト検索サーバーをインストールすると、インストール・プログラムによって、IBM i に対応した 1 つのテキスト検索サーバーが作成されます。Linux または Windows を実行するリモート・サーバーにテキスト検索サーバーをインストールすることも可能です。それらのサーバーは、DB2 Accessories Suite for z/OS® (5655-R14) (<http://www-01.ibm.com/software/data/db2imstools/db2tools/accessories-suite/>)  の一部です。そのリンクには、その製品群のダウンロードに関する情報も含まれています。

ソフトウェア要件

システムが満たさなければならない最小限のソフトウェア要件は、以下のとおりです。

- IBM Java™ Runtime 1.5 SR4
- DB2 Universal Java Driver (テキスト検索サーバーにインストールして構成します)
- IBM i の場合は、以下のプログラムをインストールする必要があります。
 - 5770SS1 オプション 30 Qshell
 - 5770SS1 オプション 33 IBM i ポータブル・アプリケーション・ソリューション環境 (IBM i PASE)
 - 5770SS1 オプション 39 International Components for Unicode
 - IBM DB2 for i の最新のグループ PTF をシステムに適用します。

重要な概念

テキスト検索機能に関する重要な概念を理解すると、DB2 for i OmniFind テキスト検索サーバーのさまざまなメリットを活用できるようになります。重要な概念としては、サポートされている文書タイプと言語があります。

テキスト検索索引の作成と更新

テキスト検索索引を作成するには、索引のプロパティを定義して宣言します。テキスト検索索引を更新するには、DB2 表から索引に新しいデータを追加します。テキスト検索索引の既存のデータを変更することによって索引を更新することも可能です。

それぞれのテキスト検索索引を作成すると、DB2 for i OmniFind テキスト検索サーバーで新しいコレクションが作成されます。初期作成後のテキスト検索索引にはデータが含まれていません。

テキスト検索索引にデータを追加するには、`SYSPROC.SYSTS_UPDATE` ストアド・プロシージャを呼び出します。最初の更新プロセスでは、テキスト列からテキスト検索索引にすべてのテキスト文書が追加されます。そのプロセスのことを初期更新といいます。その後の更新は、インクリメンタル更新になります。

テキスト検索索引を作成すると、以下のオブジェクトが作成/更新されます。

- ステージング表が `QSYS2` ライブラリーで作成されます。
- `INSERT` トリガー、`DELETE` トリガー、`UPDATE` トリガーが基本表に追加されます。
- テキスト検索索引と同じ名前の `SQL` ビューがテキスト検索索引のスキーマで作成されます。そのビューには、テキスト索引に関する情報が含まれています。例えば、そのビューを使用して、基本表の名前とステージング表の名前を取得できます。さらに、テキスト検索索引にまだ反映されていない基本表の保留変更の数もそのビューで確認できます。
- `QSYS2` ライブラリーにあるテキスト検索索引のカタログ (`SYSTEXTINDEXES` と `SYSTEXTCOLUMNS`) が更新され、新しいテキスト検索索引に対応した新しい項目が追加されます。

ステージング表に関する注意点:

- ステージング表では、ファイルの保管と復元や権限の変更以外の `DB` 操作を実行しないでください。
- 基本表に関する権限を変更する場合は、ステージング表に関する権限も変更してください。

基本表に関する注意点:

- テキスト検索索引の作成時に追加された `DELETE` トリガー、`UPDATE` トリガー、`INSERT` トリガーを削除しないでください。
- テキスト検索索引を削除すると、トリガーも削除されます。
- テキスト検索索引でキーとして使用していた `ROWID`、主キー、ユニーク列を変更/削除しないでください。
- 基本表のデータ列の変更でデータが切り捨てられると、テキスト検索索引で偽陽性の一致項目が検出される可能性があります。

関連資料

37 ページの『SYSPROC.SYSTS_UPDATE』
SYSPROC.SYSTS_UPDATE ストアド・プロシージャを呼び出すことによって、テキスト列の現在の内容に合わせてテキスト検索索引を更新できます。

107 ページの『QSYS2.SYSTEXTINDEXES 管理表』
QSYS2.SYSTEXTINDEXES 管理表では、各テキスト検索索引の情報を確認できます。それぞれのテキスト検索索引ごとに、名前、スキーマ名、テキスト検索サーバーにある関連コレクション名が記述されています。

109 ページの『QSYS2.SYSTEXTCOLUMNS 管理表』
QSYS2.SYSTEXTCOLUMNS 管理表では、テキスト検索索引に対応するテキスト列の情報を確認できます。それぞれのテキスト検索索引ごとに、索引 ID、テキスト列の名前、基本表のスキーマ名が記述されています。

非同期の索引作成とトリガー

DB2 for i OmniFind テキスト検索サーバーでは、テキスト検索索引を手動で更新することも、自動更新のスケジュールを設定することも可能です。

DB2 for i OmniFind テキスト検索サーバーで管理するテキスト検索索引は、DB2 表の更新と同時に更新されるわけではありません。テキスト検索索引の更新は、大規模な操作です。

DB2 表の列の変更は、トリガーによってローカル・ログ表に取り込まれます。そのログ表のことをステージング表といいます。トリガーは、文書の新規追加、変更、削除に関する情報を自動的にログ表に格納します。各ログ表は、それぞれ 1 つのテキスト検索索引に関連付けられています。ログ表の内容を対応するテキスト検索索引に適用する処理のことをインクリメンタル更新 といいます。

変更を照会に反映するために、テキスト検索索引を周期的に更新する必要があります。

テキスト検索索引を手動で更新する場合は、SYSPROC.SYSTS_UPDATE ストアド・プロシージャを呼び出します。

自動更新のスケジュールを設定する場合は、テキスト索引の作成時に SYSPROC.SYSTS_CREATE プロシージャの UPDATE FREQUENCY 文節を使用します。

関連資料

21 ページの『SYSPROC.SYSTS_CREATE』
SYSPROC.SYSTS_CREATE ストアド・プロシージャを呼び出すことによって、テキスト検索索引を作成できます。このストアド・プロシージャでは、テキスト列に基づいてテキスト検索索引を作成します。そのテキスト検索索引は、CONTAINS 関数または SCORE 関数を組み込んだ SQL 照会で使用できます。

37 ページの『SYSPROC.SYSTS_UPDATE』
SYSPROC.SYSTS_UPDATE ストアド・プロシージャを呼び出すことによって、テキスト列の現在の内容に合わせてテキスト検索索引を更新できます。

サポート対象の文書フォーマット

テキスト列のデータとしては、プレーン・テキスト、HTML 文書、XML 文書のほかに、検索エンジンが認識できるあらゆる文書が対象になります。

| DB2 for i OmniFind テキスト検索サーバーは、文書の構文を解析し、該当する部分を抽出して、検索の対
| 象にします。例えば、HTML 文書のタグやメタデータは、索引作成の対象になりません。

| 解析が可能な文書フォーマットは、以下のとおりです。

| • TEXT: フラット・テキスト

| • HTML: ハイパーテキスト・マークアップ言語

| • XML: Extensible Markup Language

| • INSO: DB2 for i OmniFind テキスト検索サーバーは、各種のフィルターを使用して、テキスト文書の形
| 式を検出します。サポート対象の INSO 文書フォーマットは、以下のとおりです。

| - XML

| - HTML

| - JustSystems 一太郎

| - Lotus[®] 123

| - Lotus Freelance

| - Lotus WordPro

| - Microsoft[®] Excel

| - Microsoft PowerPoint

| - Microsoft RTF (リッチ・テキスト・フォーマット)

| - Microsoft Visio

| - Microsoft Word

| - Microsoft Write

| - Portable Document Format (PDF)

| - Quattro Pro

| - リッチ・テキスト RTF

| - StarOffice Calc と OpenOffice Calc

| 索引作成の対象になる 1 つの列に存在するすべての文書は、同じ形式 (TEXT、HTML、XML、INSO) で
| なければなりません。

| XML データ

| DB2 for i OmniFind テキスト検索サーバーでは、XML パーサーで XML データの構文を解析してから、
| XML データの XML 構造の索引を作成します。その後、サポートされている XML 検索照会の構文を使
| 用して、検索結果を取得できます。

関連概念

50 ページの『XML 検索』

XML 文書の索引生成や検索が可能です。XML 検索の構文では、テキスト検索用の拡張機能を追加した W3 XPath 言語のサブセットを使用します。その拡張機能によって、XML の属性や要素に関連した数値、日付値、日時値の範囲検索を実行できます。構造要素を別個に使用することも、照会の中でフリー・テキストと組み合わせて使用することも可能です。

サポート対象のデータ・タイプ

索引を作成して検索を実行するテキスト列のデータは、バイナリー・データまたは文字データになります。

以下のデータ・タイプは、バイナリー・データです。

- BINARY
- VARBINARY
- BLOB

さらに、OMNIFINDは、以下のデータ・タイプをバイナリー・データの場合と同じように処理します。

- CHAR FOR BIT DATA
- VARCHAR FOR BIT DATA

以下のデータ・タイプは、文字データです。

- CHAR FOR SBCS DATA または FOR MIXED DATA
- VARCHAR FOR SBCS DATA または FOR MIXED DATA
- CLOB
- DBCLOB
- GRAPHIC
- VARGRAPHIC
- XML

バイナリー・データの場合は、テキスト検索索引の作成時に使用するコード化文字セット ID (CCSID) を指定できます。文字データの場合は、DB2 データベースがエンコード方式を認識できるので、CCSID を明示的に指定したとしても、その指定は無視されます。

テキスト・スコアと同義語サポート

同義語を使用すれば、照会の結果を改善できます。テキスト・スコアを使用すれば、照会と結果の一致の程度を確認できます。

テキスト・スコア

検索処理では、テキスト・スコアが計算されます。照会結果にそのスコアを組み込むことも可能です。テキスト・スコアは 0 から 1 の値であり、小数点以下の桁数は最大で 3 桁になります。例えば、0.000 から 1.000 の値になります。テキスト・スコアは、テキスト検索索引に含まれている他のすべての文書と比較した相対的なスコアであり、このスコアによって照会と結果の一致の程度を確認できます。

OMNIFIND は、検索語の一般的な重要度や出現箇所の隣接性など、さまざまな係数に基づいてテキスト・スコアを作成します。一般的な重要度は、すべての文書で検索語が出現する頻度を基準にして、各文書で検索語が出現する頻度の相対位置を示した係数です。

同義語サポート

DB2 for i OmniFind テキスト検索サーバーでは、同義語を使用して照会結果を変更できます。同義語を使用すれば、照会の基準に一致する文書の数が増え、照会結果の数が増える可能性があります。その一方で、同義語を使用することによって照会の精度が落ち、検索基準に完全に一致する少数の文書を検出するのが難しくなる可能性もあります。

照会で同義語を使用しない、というのがデフォルトの動作です。照会で同義語を使用するには、同義語ディクショナリーを作成し、同義語ツールを使用して同義語ディクショナリーをコレクションに追加する必要があります。

同義語の詳細については、81 ページの『同義語ディクショナリー』を参照してください。

関連資料

43 ページの『SCORE』

SCORE 関数では、検索回数として指定する基準を使用して、テキスト検索索引を検索できます。この関数は、文書が照会内容に一致する比率を測定した関連度スコアを返します。

言語処理

DB2 for i OmniFind テキスト検索サーバーには、英語以外の文書や照会の言語処理をサポートするためのディクショナリー・パックが用意されています。

DB2 for i OmniFind テキスト検索サーバーでは、中国語、日本語、韓国語などの言語に関して、ディクショナリー・ベースのワード・セグメンテーションの代わりに *n-gram* セグメンテーション という手法をサポートしています。*n-gram* セグメンテーション は、一定数の文字が重なり合っている文字列を 1 つの単語と見なす、という分析方式です。一方、Unicode ベースの空白文字セグメンテーション では、ブランク・スペースに基づいて単語を区切ります。

テキスト文書がいずれかのサポート対象言語で記述されている場合は、テキストを解析してトークンに変換するときに言語処理が実行されます。非サポート対象言語の場合は、エラー・コードが返されます。

テキスト検索索引を検索すると、照会語の言語処理上のバリエーションが含まれている一致項目が表示されます。単語のバリエーションは、照会の言語によって異なります。

サポート対象の言語

特定の言語を使用してテキスト文書进行处理することを指定できます。

SYSPROC.SYSTS_CREATE 管理ストアド・プロシージャでは、索引付きのテキスト・データの言語を指定できます。値を AUTO に設定すると、DB2 for i OmniFind テキスト検索サーバーが言語を判別しようとし、短い文書の場合は、自動検出の結果が不正確になる場合もあるので、そのような設定はお勧めできません。言語処理のデフォルト言語は、英語 (en_US) です。

サポート対象言語の 5 文字の言語コードを以下の表にまとめます。

表 1. サポート対象言語の 5 文字の言語コード

言語コード	言語
ar_AA	アラビア語
cs_CZ	チェコ語
da_DK	デンマーク語
de_CH	ドイツ語 (スイス)

表 1. サポート対象言語の 5 文字の言語コード (続き)

言語コード	言語
de_DE	ドイツ語 (ドイツ)
el_GR	ギリシャ語
en_AU	英語 (オーストラリア)
en_GB	英語 (英国)
en_US	英語 (米国)
es_ES	スペイン語 (スペイン)
fi_FI	フィンランド語
fr_CA	フランス語 (カナダ)
fr_FR	フランス語 (フランス)
it_IT	イタリア語
ja_JP	日本語
ko_KR	韓国語
nb_NO	ノルウェー語ブークモール
nl_NL	オランダ語
nn_NO	ノルウェー語ニーノシュク
pl_PL	ポーランド語
pt_BR	ブラジル・ポルトガル語
pt_PT	ポルトガル語 (ポルトガル)
ru_RU	ロシア語
sv_SE	スウェーデン語
zh_CN	中国語 (簡体字)
zh_TW	中国語 (繁体字)

関連資料

21 ページの『SYSPROC.SYSTS_CREATE』

SYSPROC.SYSTS_CREATE ストアード・プロシージャーを呼び出すことによって、テキスト検索索引を作成できます。このストアード・プロシージャーでは、テキスト列に基づいてテキスト検索索引を作成します。そのテキスト検索索引は、CONTAINS 関数または SCORE 関数を組み込んだ SQL 照会で使用できます。

中国語、日本語、韓国語の文書の言語処理

中国語、日本語、韓国語の文書を処理するときには、ディクショナリー・ベース・セグメンテーションまたは n-gram セグメンテーションのいずれかを使用できます。

検索エンジンの側からすれば、高品質の検索結果が得られるかどうかは、テキスト処理の手法にかなりの程度依存しています。文書からテキストを抽出した後に、テキスト処理の最初のステップになるのは、テキストに含まれている個々の単語を識別する、ということです。テキストに含まれている個々の単語を識別することをセグメンテーションといいます。多くの言語では、空白文字 (ブランク、行末、ある種の句読法) に基づいてワード境界を認識できます。ところが、中国語、日本語、韓国語では、文字間の空白文字によって単語を区分するわけではないので、他の手法を使用しなければなりません。

DB2 for i OmniFind テキスト検索サーバーでは、中国語、日本語、韓国語の言語処理をサポートするために以下の 2 つの方式が用意されています。

- ・ ディクショナリー・ベース・ワード・セグメンテーション (形態学的分析ともいいます)
- ・ n-gram セグメンテーション

ディクショナリー・ベース・ワード・セグメンテーション

ディクショナリー・ベース・ワード・セグメンテーションでは、言語固有のディクショナリーを使用して、文書内の文字列の中からそれぞれの単語を識別します。ディクショナリーを使用してワード境界を識別するこの手法では、確かに正確な検索結果が得られます。ただし、ディクショナリー・ベース・ワード・セグメンテーションでは、特定の突き合わせ結果が失われてしまう可能性もあります。

n-gram セグメンテーション

n-gram セグメンテーションでは、ワード境界の識別に伴う問題を回避し、文字が重なり合っている文字列のペアの索引を生成します。DB2 for i OmniFind テキスト検索サーバーでは、2つの文字を使用するの

で、この手法のことを bi-gram セグメンテーションともいいます。

n-gram セグメンテーションの手法では、検索語が含まれているすべての一致文書が常に返されますが、照会の条件に一致しない文書が返されてしまう可能性もあります。

DB2 for i OmniFind テキスト検索サーバーでは、中国語、日本語、韓国語で n-gram セグメンテーションを使用した事前構成索引がデフォルトで用意されています。

両方のタイプの言語処理の動作を理解するために、「神奈川県知事選挙」というテキストが含まれている文書を例として取り上げましょう。日本語では、このテキストに 8 つの文字が含まれています。この例では、その 8 つの文字を「A B C D E F G H」と記述することにします。ユーザーが入力できるサンプル照会としては、「知事選挙」が考えられます。これは、「E F G H」の 4 文字に相当します (つまり、文書テキストとサンプル照会は、同じ文字を共有しています)。

n-gram セグメンテーションの処理を使用する場合:

検索エンジンは、文書の索引の生成後に、「神奈川県知事選挙」というテキストを「AB BC CD DE EF FG GH」という文字のセット (セグメント) に区分します。

「知事選挙」というサンプル照会は、「EF FG GH」という文字のセット (セグメント) に区分されます。「知事選挙」というサンプル照会で検索を実行すれば、その文書が検出されます。文書テキストとサンプル照会の両方のトークンが同じ順序で存在するからです。

n-gram セグメンテーションを有効にすると、より多くの結果が得られる可能性もありますが、その一方で精度が落ちる危険もあります。例えば、日本語では、「京都」という照会で検索を実行したときに、「東京都」というテキストが含まれている文書が索引に入っていると、その文書が検出されてしまいます。「東京都」と「京都」は、2つの同じ日本語文字を共有しているからです。

n-gram セグメンテーションの処理を使用しない場合:

検索エンジンは、文書の索引の生成後に、「神奈川県知事選挙」というテキストを「ABC DEF GH」という文字のセット (セグメント) に区分します。

「知事選挙」というサンプル照会は、「EF GH」という文字のセット (セグメント) に区分されます。「EF」という文字列は、文書テキストのトークンには存在しません (文書に含まれているのは、「EF」ではなく「DEF」です)。

文書テキストには「DEF」が含まれていますが、照会には「EF」だけが含まれています。したがって、サンプル照会でその文書が検出される可能性は低くなります。

n-gram セグメンテーションを有効にしなければ、より正確な結果が得られる可能性がありますが、結果の数は少なくなると考えられます。

サーバーの別名

サーバーの別名を使用して、意味のある名前をサーバーに割り当てることができます。

それぞれのテキスト検索サーバーは、QSYS2.SYSTEXTSERVERS カタログの SERVERID 列によって一意的に識別できます。SERVERID 列には、データベースによって増分的に生成される整数が入っています。

QSYS2.SYSTEXTSERVERS には、各サーバーに意味のある別名を割り当てるための ALIASNAME 列があります。SYSTS_START、SYSTS_STOP、SYSTS_CREATE の各プロシージャでは、サーバーの別名でサーバーを参照できるようになっています。

注: サーバーの別名は、QSYS2.SYSTEXTSERVERS カタログ表で ALIASNAME の値を更新することによって直接変更できます。

サーバー番号 1 のサーバーに「PRIMARY_LOCAL_SERVER」という別名を割り当てます。

```
UPDATE QSYS2.SYSTEXTSERVERS
SET ALIASNAME = 'PRIMARY_LOCAL_SERVER'
WHERE SERVERID = 1
```

サーバーから別名を削除するには、その列をヌルに設定します。

サーバー番号 1 のサーバーから別名を削除します。

```
UPDATE QSYS2.SYSTEXTSERVERS
SET ALIASNAME = NULL
WHERE SERVERID = 1
```

関連資料

110 ページの『QSYS2.SYSTEXTSERVERS 管理表』

QSYS2.SYSTEXTSERVERS 管理表では、テキスト検索サーバーのインストール先に関する情報を確認できます。

17 ページの『SYSPROC.SYSTS_START』

SYSPROC.SYSTS_START ストアード・プロシージャを呼び出すことによって、DB2 のテキスト検索機能を有効にできます。

19 ページの『SYSPROC.SYSTS_STOP』

SYSPROC.SYSTS_STOP ストアード・プロシージャを呼び出すことによって、DB2 のテキスト検索機能を停止できます。このストアード・プロシージャを実行すると、QSYS2.SYSTEXTSERVERS カタログの SERVERSTATUS 値が 1 (停止) に設定されます。

21 ページの『SYSPROC.SYSTS_CREATE』

SYSPROC.SYSTS_CREATE ストアード・プロシージャを呼び出すことによって、テキスト検索索引を作成できます。このストアード・プロシージャでは、テキスト列に基づいてテキスト検索索引を作成します。そのテキスト検索索引は、CONTAINS 関数または SCORE 関数を組み込んだ SQL 照会で使用できます。

テキスト検索機能のインストールと構成

DB2 for i OmniFind テキスト検索サーバー をインストールして構成できます。テキスト検索索引を作成/更新して、表の列に対するテキスト検索機能の使用を開始することも可能です。

DB2 for i OmniFind テキスト検索サーバーのインストール

ライセンス・プログラムの標準的なインストール手順で、IBM i に DB2 for i OmniFind テキスト検索サーバーをインストールします。デフォルトのテキスト・サーバーが作成され、デフォルトのサーバー情報が QSYS2.SYSTEXTSERVERS 表に取り込まれます。オプションとして、インストール後にさらにテキスト検索サーバーを追加することもできます。

QSYS2.SYSTEXTSERVERS 表へのデータの取り込み

DB2 for i OmniFind テキスト検索サーバーは、IBM のライセンス・プログラム 5733-OMF です。ライセンス・プログラムをインストールする方法の詳細については、『追加ライセンス・プログラムのインストール』を参照してください。この製品を検索する場合は、コマンド行で GO LICPGM と入力し、オプション 10 (導入済みライセンス・プログラムの表示) を選択します。ライセンス・プログラムのリストにこの製品が表示されます。

QSYS2.SYSTEXTSERVERS 表には、インストール済みの DB2 for i OmniFind テキスト検索サーバーに関する情報を組み込みます。その情報は DB2 for i で使用できます。DB2 for i OmniFind テキスト検索サーバー製品を初めてインストールすると、デフォルトのテキスト・サーバーが IBM i システムに作成されます。さらに、QSYS2.SYSTEXTSERVERS 表にデフォルトのサーバー情報が取り込まれます。

追加のテキスト検索サーバーの作成

リモートの IBM システムでテキスト検索サーバーを使用する場合や、IBM 以外のサーバー (Windows や Linux のサーバーなど) を使用する場合は、SQL INSERT ステートメントを使用してその表に明示的にデータを取り込む必要があります。

QSYS2.SYSTEXTSERVERS 表に追加のサーバーを取り込むには、SQL INSERT を使用して以下の手順を実行します。

1. SQL INSERT ステートメントを実行して、QSYS2.SYSTEXTSERVERS 表の SERVERPORT 列と SERVERNAME 列で各テキスト検索サーバーのポート番号とサーバー名を指定します。
2. SQL INSERT ステートメントを使用して、QSYS2.SYSTEXTSERVERS 表の SERVERAUTHTOKEN 列で各テキスト検索サーバーの認証トークンを指定します。

DB2 データベースがテキスト検索サーバーと通信するときには、認証トークンが必要になります。そのトークンは、インストール時にテキスト検索サーバーで生成されます。

3. SQL INSERT ステートメントを使用して、QSYS2.SYSTEXTSERVERS 表の SERVERMASTERKEY 列で各テキスト検索サーバーのサーバー・キーを指定します。
4. オプション: ALIASNAME 列でサーバーの別名を指定します。その後の操作では、その別名でサーバーを参照できるようになります。

例

テキスト検索サーバーの必要な情報を QSYS2.SYSTEXTSERVERS 表の各列にコピーする SQL INSERT ステートメントの例を以下に示します。

```
INSERT INTO QSYS2.SYSTEXTSERVERS(SERVERNAME,  
                                SERVERADRIDFNO,  
                                SERVERPORT,  
                                SERVERTYPE,  
                                SERVERAUTHTOKEN,  
                                SERVERMASTERKEY,  
                                SERVERPATH,  
                                ALIASNAME)
```

```
|      VALUES('127.0.0.1',  
|              VARBINARY(X'0000'),  
|              49200,  
|              0,  
|              'AH2X4w==',  
|              'b1YhcR90858ArwxLJeIY/Q==',  
|              '/QOpenSys/QIBM/ProdData/TextSearch/server1/bin/',  
|              'LOCALSERVER2');
```

| 関連概念

| 12 ページの『サーバーの別名』
| サーバーの別名を使用して、意味のある名前をサーバーに割り当てることができます。

| 関連資料

| 110 ページの『QSYS2.SYSTEXTSERVERS 管理表』
| QSYS2.SYSTEXTSERVERS 管理表では、テキスト検索サーバーのインストール先に関する情報を確認できます。

| テキスト検索機能の開始

| テキスト検索関数の使用を開始する前に、SYSPROC.SYSTS_START ストアド・プロシージャーを呼び出します。このプロシージャーを呼び出すことによって、システムのローカル・サーバーとして定義されているすべての実動サーバーを開始できます。

| テキスト検索のサポートには、CONTAINS 関数、SCORE 関数、以下の管理ストアド・プロシージャーを使用した SQL ステートメントが含まれます。

- | • SYSPROC.SYSTS_CREATE
- | • SYSPROC.SYSTS_UPDATE
- | • SYSPROC.SYSTS_DROP

関連資料

17 ページの『SYSPROC.SYSTS_START』

SYSPROC.SYSTS_START ストアド・プロシージャを呼び出すことによって、DB2 のテキスト検索機能を有効にできます。

21 ページの『SYSPROC.SYSTS_CREATE』

SYSPROC.SYSTS_CREATE ストアド・プロシージャを呼び出すことによって、テキスト検索索引を作成できます。このストアド・プロシージャでは、テキスト列に基づいてテキスト検索索引を作成します。そのテキスト検索索引は、CONTAINS 関数または SCORE 関数を組み込んだ SQL 照会で使用できます。

37 ページの『SYSPROC.SYSTS_UPDATE』

SYSPROC.SYSTS_UPDATE ストアド・プロシージャを呼び出すことによって、テキスト列の現在の内容に合わせてテキスト検索索引を更新できます。

35 ページの『SYSPROC.SYSTS_DROP』

SYSPROC.SYSTS_DROP ストアド・プロシージャを呼び出すことによって、SYSPROC.SYSTS_CREATE ストアド・プロシージャで定義したテキスト検索索引を削除できます。

40 ページの『CONTAINS』

CONTAINS 関数では、検索指数として指定する基準を使用して、テキスト検索索引を検索できます。この関数は、一致項目が見つかったかどうかを示す結果を返します。

43 ページの『SCORE』

SCORE 関数では、検索指数として指定する基準を使用して、テキスト検索索引を検索できます。この関数は、文書が照会内容に一致する比率を測定した関連度スコアを返します。

テキスト検索索引の作成

SYSPROC.SYSTS_CREATE ストアド・プロシージャを呼び出して、テキスト検索索引を作成できます。

DB2 の基本表に、ROWID 列、ユニーク・キー、主キーのいずれかを組み込む必要があります。

テキストが入った列が含まれている既存の DB2 表でテキスト検索索引を作成するには、以下のようになります。

SYSPROC.SYSTS_CREATE ストアド・プロシージャを呼び出します。

テキスト検索索引を最初に更新するまで、その索引は空です。

関連資料

21 ページの『SYSPROC.SYSTS_CREATE』

SYSPROC.SYSTS_CREATE ストアド・プロシージャを呼び出すことによって、テキスト検索索引を作成できます。このストアド・プロシージャでは、テキスト列に基づいてテキスト検索索引を作成します。そのテキスト検索索引は、CONTAINS 関数または SCORE 関数を組み込んだ SQL 照会で使用できます。

テキスト検索索引の更新

SYSPROC.SYSTS_UPDATE ストアド・プロシージャを呼び出して、テキスト検索索引を更新できます。

まず、SYSTS_UPDATE 呼び出しを使用して、テキスト検索索引にデータを設定します。さらに、DB2 表の内容に変更があったときに、その変更に合わせてテキスト検索索引を同期化する場合にも、その呼び出しを使用できます。

テキスト検索索引の更新後に、そのテキスト検索索引に対して検索照会を実行できます。更新後に、基本表のテキスト検索列が変更されている可能性もあります。その場合は、テキスト検索索引の更新を次に実行するまで、その変更内容が検索照会の結果に反映されることはありません。

関連資料

37 ページの『SYSPROC.SYSTS_UPDATE』

SYSPROC.SYSTS_UPDATE ストアド・プロシージャを呼び出すことによって、テキスト列の現在の内容に合わせてテキスト検索索引を更新できます。

テキスト検索索引の検索

CONTAINS 関数または SCORE 関数を組み込んだ SQL ステートメントを使用して、テキスト検索索引を検索できます。その関数で検索指数の基準を指定します。

DB2 表に対するテキスト照会を実行するユーザーは、あらゆる形式の照会で必要になる標準的な特権セットを持っていないければなりません (DB2 SQL 解説書を参照してください)。

関連資料

40 ページの『CONTAINS』

CONTAINS 関数では、検索指数として指定する基準を使用して、テキスト検索索引を検索できます。この関数は、一致項目が見つかったかどうかを示す結果を返します。

43 ページの『SCORE』

SCORE 関数では、検索指数として指定する基準を使用して、テキスト検索索引を検索できます。この関数は、文書が照会内容に一致する比率を測定した関連度スコアを返します。

文書の切り捨て

DB2 for i OmniFind テキスト検索サーバーでは、テキスト文書ごとに索引生成の対象にできる文字の数が制限されています。この文字の長さ制限によって、テキスト検索索引で大きなテキスト文書が切り捨てられることもあります。

1000 万個を超える Unicode 文字が含まれている文書は、テキスト検索サーバーによって切り捨てられる可能性があります。リッチ・テキスト文書の場合は、プレーン・テキストへの変換後にこの制限が適用されません。

解析段階でテキスト文書が切り捨てられると、完全に処理されていない文書があるという趣旨の警告が生成されます。その警告は、ジョブ・ログに書き込まれます。その文書では、一部の索引だけが生成されます。制限値に達した後に文書に残っているテキストについては、索引が生成されず、そのテキストは、検索の対象になりません。

検索処理時に想定外の動作を回避するために、切り捨てが発生した文書をテキスト検索索引から削除できます。その文書を削除するには、対応するレコードを DB2 表から削除するか、文書の値を空またはヌルに変更します。

テキスト検索の管理ストアード・プロシージャ

一連の管理 SQL ストアード・プロシージャを使用して、テキスト検索機能の開始と停止の操作や、テキスト検索索引の作成、削除、更新の操作を実行できます。これらのプロシージャは、どの SQL インターフェースからでも呼び出せます。IBM i のコマンド行から CL コマンドを使用してこれらのプロシージャを呼び出すことはできません。

STRSQL を使用してシステム・カタログを表示するときに、ジョブを 65535 以外の CCSID に設定すれば、各列の内容を確認できます。

SYSPROC.SYSTS_START

SYSPROC.SYSTS_START ストアード・プロシージャを呼び出すことによって、DB2 のテキスト検索機能を有効にできます。

テキスト検索機能には、CONTAINS 関数、SCORE 関数、管理ストアード・プロシージャ (テキスト検索索引の保守作業のためのストアード・プロシージャ) を使用した SQL 照会のサポートが含まれています。

SYSPROC.SYSTS_START ストアード・プロシージャは、QSYS2.SYSTEXTSERVERS 表にサーバーを追加したり、その表に含まれているサーバーを変更したりするたびに実行する必要があります。

テキスト検索機能を開始していなければ、CONTAINS 関数と SCORE 関数に対してデータベースから SQLCODE -20424 (理由コード 4) が返されます。さらに、サーバーを開始していなければ、管理プロシージャの SYSPROC.SYSTS_CREATE と SYSPROC.SYSTS_UPDATE も SQLCODE -20424 で失敗します。

QSYS2.SYSTEXTSERVERS 表に含まれているテキスト検索サーバーについては、TCP/IP 名が解決されます。SYSPROC.SYSTS_START ストアード・プロシージャに対する複数の呼び出しもエラーとは見なされません。このプロセスによって、QSYS2.SYSTEXTSERVERS 表でのアドレス解決を検証できます。

ローカル・サーバーの場合は、サーバーがまだ始動していなければ、このストアード・プロシージャの呼び出しによってサーバーが始動します。リモート・サーバーの場合は、このプロシージャ呼び出しによって、サーバーがアクティブであるかどうかを検証されますが、実際にサーバーが始動することはありません。

前提条件

SYSPROC.SYSTS_START ストアード・プロシージャを呼び出す前に、QSYS2.SYSTEXTSERVERS 表に少なくとも 1 つの項目が含まれていることを確認します。

権限

このストアード・プロシージャを呼び出すときに使用するユーザー ID には、以下の特権が必要です。

- このプロシージャに対する *EXECUTE 権限
- SYSTXTSRVRS 表に対する SELECT 特権と UPDATE 特権
- SYSTXTSRVRS ファイルの QSYS2 ライブラリーに対する *EXECUTE 権限
- *JOBCTL 権限または QIBM_DB_SQLADM セキュリティー特殊関数使用権限

SQL 特権に対応するシステム権限については、『GRANT (表またはビュー特権)』を参照してください。

構文

```
CALL SYSPROC.SYSTS_START(  
    [serverid]  
    [aliasname]  
)
```

スキーマ修飾子は SYSPROC です。

パラメーター

serverid または *aliasname*

開始するサーバーの ID を指定します。*serverid* またはサーバーの *aliasname* は文字列です。ID を指定しない場合は、デフォルトですべてのサーバーを開始することになります。ID の文字列は、QSYS2.SYSTEXTSERVERS 表の SERVERID 列に存在する有効な *serverid*、または ALIASNAME 列に存在する有効なサーバーの *aliasname* でなければなりません。ID を整数値に変換できる場合は、*serverid* と解釈されます。ID を整数値に変換できない場合は、サーバーの *aliasname* と解釈されます。

このパラメーターのデータ・タイプは VARCHAR(128) です。

注: *serverid* または *aliasname* の値を指定しない場合は、実動サーバーと見なされるサーバーだけが始動します。実動サーバーと見なされるサーバーは、QSYS2.SYSTEXTSERVERS 表で **SERVERCLASS** パラメーターが 0 になっているサーバーです。テスト・サーバーを開始する場合は、そのテスト・サーバーに関連した *serverid* または *aliasname* を指定する必要があります。

すべての実動サーバーを開始するには、以下のようにします。

```
CALL SYSPROC.SYSTS_START()
```

ID 1 のサーバーを開始するには、以下のようにします。

```
CALL SYSPROC.SYSTS_START(1)
```

別名「LOCAL_SERVER」のサーバーを開始するには、以下のようにします。

```
CALL SYSPROC.SYSTS_START('LOCAL_SERVER')
```

別名「local_server」のサーバーを開始するには、以下のようにします。

```
CALL SYSPROC.SYSTS_START('local_server')
```

| 関連概念

- | 12 ページの『サーバーの別名』
- | サーバーの別名を使用して、意味のある名前をサーバーに割り当てることができます。

| 関連タスク

- | 67 ページの『DB2 for i OmniFind テキスト検索サーバーの開始』
- | SYSPROC.SYSTS_START を呼び出すことによって、DB2 for i OmniFind テキスト検索サーバーを開始できます。

| 関連資料

- | 110 ページの『QSYS2.SYSTEXTSERVERS 管理表』
- | QSYS2.SYSTEXTSERVERS 管理表では、テキスト検索サーバーのインストール先に関する情報を確認できます。
- | 40 ページの『CONTAINS』
- | CONTAINS 関数では、検索引数として指定する基準を使用して、テキスト検索索引を検索できます。この関数は、一致項目が見つかったかどうかを示す結果を返します。
- | 43 ページの『SCORE』
- | SCORE 関数では、検索引数として指定する基準を使用して、テキスト検索索引を検索できます。この関数は、文書が照会内容に一致する比率を測定した関連度スコアを返します。

| SYSPROC.SYSTS_STOP

- | SYSPROC.SYSTS_STOP ストアード・プロシージャを呼び出すことによって、DB2 のテキスト検索機能を停止できます。このストアード・プロシージャを実行すると、QSYS2.SYSTEXTSERVERS カタログの SERVERSTATUS 値が 1 (停止) に設定されます。
- | このストアード・プロシージャの完了後に、CONTAINS 関数または SCORE 関数を使用した SQL 照会や、索引の保守作業のための管理ストアード・プロシージャを実行すると、テキスト検索サーバーへのアクセスも試みられずに障害が返されます。
- | 注: SYSTS_STOP の呼び出し時に、管理プロシージャや、CONTAINS または SCORE 組み込み関数を使用した SQL 照会が実行中の状態になっていた場合は、その管理プロシージャや SQL 照会が最後まで処理を完了します。
- | サーバーを停止しても、索引の基本表に対する変更は、引き続きログに記録されます。ただし、索引のスケジュール更新は、SYSPROC.SYSTS_START を呼び出すまで発生しません。

| 権限

- | このストアード・プロシージャを呼び出すときに使用するユーザー ID には、以下の特権が必要です。
- | • このプロシージャに対する *EXECUTE 権限
- | • SYSTEXTSERVERS 表に対する SELECT 特権と UPDATE 特権
- | • SYSTEXTSERVERS ファイルの QSYS2 ライブラリーに対する *EXECUTE 権限
- | • *JOBCTL 権限または QIBM_DB_SQLADM セキュリティー特殊関数使用権限
- | SQL 特権に対応するシステム権限については、『GRANT (表またはビュー特権)』を参照してください。

| 構文

|

| ►► SYSTS_STOP (

<i>serverid</i>
<i>aliasname</i>

)

| スキーマ修飾子は SYSPROC です。

| パラメーター

| *serverid* または *aliasname*

| 停止するサーバーの ID を指定します。*serverid* またはサーバーの *aliasname* は文字列です。ID を指定しない場合は、デフォルトですべてのサーバーを停止することになります。ID の文字列は、QSYS2.SYSTEXTSERVERS 表の SERVERID 列に存在する有効な *serverid*、または ALIASNAME 列に存在する有効なサーバーの *aliasname* でなければなりません。ID を整数値に変換できる場合は、*serverid* と解釈されます。ID を整数値に変換できない場合は、サーバーの *aliasname* と解釈されます。

| このパラメーターのデータ・タイプは VARCHAR(128) です。

| すべての実動サーバーを停止するには、以下のようにします。

| CALL SYSPROC.SYSTS_STOP()

| ID 1 のサーバーを停止するには、以下のようにします。

| CALL SYSPROC.SYSTS_STOP(1)

| 別名「LOCAL_SERVER」のサーバーを停止するには、以下のようにします。

| CALL SYSPROC.SYSTS_STOP('LOCAL_SERVER')

| 別名「local_server」のサーバーを停止するには、以下のようにします。

| CALL SYSPROC.SYSTS_STOP('local_server')

関連概念

- 12 ページの『サーバーの別名』
- サーバーの別名を使用して、意味のある名前をサーバーに割り当てることができます。

関連資料

- 110 ページの『QSYS2.SYSTEXTSERVERS 管理表』
- QSYS2.SYSTEXTSERVERS 管理表では、テキスト検索サーバーのインストール先に関する情報を確認できます。
- 17 ページの『SYSPROC.SYSTS_START』
- SYSPROC.SYSTS_START ストアド・プロシージャを呼び出すことによって、DB2 のテキスト検索機能を有効にできます。
- 40 ページの『CONTAINS』
- CONTAINS 関数では、検索指数として指定する基準を使用して、テキスト検索索引を検索できます。この関数は、一致項目が見つかったかどうかを示す結果を返します。
- 43 ページの『SCORE』
- SCORE 関数では、検索指数として指定する基準を使用して、テキスト検索索引を検索できます。この関数は、文書が照会内容に一致する比率を測定した関連度スコアを返します。

SYSPROC.SYSTS_CREATE

- SYSPROC.SYSTS_CREATE ストアド・プロシージャを呼び出すことによって、テキスト検索索引を作成できます。このストアド・プロシージャでは、テキスト列に基づいてテキスト検索索引を作成します。そのテキスト検索索引は、CONTAINS 関数または SCORE 関数を組み込んだ SQL 照会で使用できます。

- テキスト検索索引は、QSYS2.SYSTEXTSERVERS 表で列挙されているいずれかのテキスト検索サーバーで作成します。テキスト検索索引は、DB2 表の更新と同時に更新されるわけではありません。DB2 表の列の変更は、トリガーによって取り込まれ、ステージング表にログとして書き込まれます。

- 注:** このストアド・プロシージャでは、テキスト検索索引を定義する操作だけを実行します。新しいテキスト検索索引に対して SYSPROC.SYSTS_UPDATE ストアド・プロシージャを最初に呼び出すときまで、そのテキスト検索索引にはデータが含まれていません。順序としては、最初に表にデータを取り込んでから、テキスト検索索引を作成します。表にデータを取り込んでからテキスト検索索引を作成すれば、索引の初期更新の前に変更トリガーが起動する、という動作を回避できます。

前提条件

- SYSPROC.SYSTS_CREATE ストアド・プロシージャを呼び出す前に、以下の前提条件を確認します。
- SYSPROC.SYSTS_START ストアド・プロシージャを呼び出すことによって、DB2 のテキスト検索機能を開始し、少なくとも 1 つのテキスト検索サーバーを実行中の状態にしておく必要があります。
- 主キー、ユニーク索引、ROWID のいずれかとして定義されている列が表に含まれている必要があります。
- QSYS2.SYSTEXTSERVERS 表に少なくとも 1 つの項目が含まれている必要があります。

権限

- ステートメントの許可 ID が保持する特権には、以下のうち最低 1 つの特権が含まれている必要があります。

| • このスキーマでの作成特権。詳細については、『権限、特権、およびオブジェクト所有権』を参照してください。

| • 管理権限

| ステートメントの許可 ID が保持する特権には、以下のうち最低 1 つの特権が含まれている必要があります。

| • 以下のシステム権限:

| - 論理ファイル作成 (CRTLF) コマンドに対する *USE

| - データ・ディクショナリーに対する *CHANGE (テキスト索引の作成先のライブラリーがデータ・ディクショナリー付きの SQL スキーマの場合)

| • 管理権限

| また、ステートメントの許可 ID が保持する特権には、以下のうちの最低 1 つの特権も含まれている必要があります。

| • 参照先の表に対する権限:

| - その表に対する INDEX 特権

| - その表が含まれているライブラリーに対する *EXECUTE システム権限

| • 管理権限

| • テキスト索引の作成先のライブラリーと同じ名前のユーザー・プロファイルが存在し、その名前がステートメントの許可 ID とは異なっている、という状況で SQL 名を指定する場合は、ステートメントの許可 ID が所有する特権に少なくとも以下のいずれかの特権が含まれている必要があります。

| - その名前のユーザー・プロファイルに対する *ADD システム権限

| - 管理権限

| 特殊タイプを参照する場合は、ステートメントの許可 ID が所有する特権に少なくとも以下のいずれかの特権が含まれている必要があります。

| • ステートメントで指定する特殊タイプごとに必要な権限:

| - その特殊タイプに対する USAGE 特権、および

| - その特殊タイプが含まれているライブラリーに対する *EXECUTE システム権限

| • 管理権限

| SQL 特権に対応するシステム権限については、『GRANT (表またはビュー特権)』を参照してください。

| 構文

| ►►SYSYS_CREATE—(—indexSchema—,—indexName—,—textSource—,—options—)——►►

|

| スキーマ修飾子は SYSPROC です。

| パラメーター

| *indexSchema*

| テキスト検索索引のスキーマを識別します。このパラメーターがヌルの場合、呼び出し側の CURRENT SCHEMA 特殊レジスターの値が使用されます。この値は、有効な SQL 名でなければなりません。

注: 名前が、使用可能な SQL キーワードまたは OmniFind キーワードと競合する場合、名前を二重引用符で囲みます。

このパラメーターのデータ・タイプは VARCHAR(128) です。

indexName

テキスト検索索引の名前を識別します。索引スキーマと一緒にテキスト検索索引の名前を記述すると、DB2 サブシステムでテキスト検索索引を一意的に識別できます。このパラメーターには、ヌル以外の値を指定する必要があります。この値は、有効な SQL 名でなければなりません。

注: 名前が、使用可能な SQL キーワードまたは OmniFind キーワードと競合する場合、名前を二重引用符で囲みます。

このパラメーターのデータ・タイプは VARCHAR(128) です。

textSource

文書テキスト・ソースの表と列の指定内容を記述します。このパラメーターには、ユーザー定義関数を組み込むこともできます。このパラメーターには、ヌル以外の値を指定する必要があります。

このパラメーターのデータ・タイプは VARCHAR(1024) です。

textSource:

`tableSchema.tableName-(text-column-name | function-name-(text-column-name)-)`

Diagram description: The text shows a syntax diagram for the textSource parameter. It starts with a horizontal line containing `tableSchema.tableName-`. From the end of this line, a vertical line goes down to a horizontal line containing `text-column-name`. From the end of this line, a vertical line goes down to a horizontal line containing `function-name-(text-column-name)-`. From the end of this line, a vertical line goes down to a horizontal line containing `function-schema` and `.` (representing a schema separator).

tableSchema

テキスト検索索引を作成する表のスキーマを指定します。

注: 名前が、使用可能な SQL キーワードまたは OmniFind キーワードと競合する場合、名前を二重引用符で囲みます。

tableName

外部のテキスト検索索引を作成する列が含まれているテキスト表の名前を指定します。

注:

- ビューと論理ファイルはサポートされていません。
- 別名では、表を参照するか、物理ファイルの 1 つのメンバーを参照する必要があります。
- SQL キーワードまたは OmniFind キーワードと競合する名前は、二重引用符で囲みます。

text-column-name

テキスト検索索引を作成するためのテキストが含まれている列の名前を指定します。その列は、CHAR、CHAR FOR BIT DATA、BINARY、VARCHAR、VARCHAR FOR BIT DATA、VARBINARY、CLOB、DBCLOB、BLOB、XML、GRAPHIC、VARGRAPHIC のいずれかのタイプでなければなりません。そのいずれかのデータ・タイプでない場合は、サポート対象のデータ・タイプを返す外部関数を指定できます。

注:

- 1 つの列で作成できるテキスト検索索引は 1 つに限られます。その列のテキスト検索索引が既に存在する場合は、SQLCODE-20427 が返されます。

- SQL キーワードまたは OmniFind キーワードと競合する名前は、二重引用符で囲みます。

function-schema. function-name

組み込み関数またはユーザー定義関数のスキーマと名前を指定します。その関数を使用して、列に格納されているテキスト文書を変更できます。さらに、その関数を使用して、サポート対象のデータ・タイプでない列のテキスト文書にアクセスすることも可能です。あるいは、他の場所に格納されている文書にアクセスするために、その関数を使用することもできます。その関数には、テキスト列のデータ・タイプに関する 1 つの入力パラメーターがあります。例えば、別の表にある文書の内容に対する外部キーとしての役割を果たす整数を指定できます。その関数からは、DB2 for i OmniFind テキスト検索サーバーのいずれかのサポート対象のデータ・タイプの値が返されます。その関数によって、テキスト列の内容が、索引を生成する文書の内容に変換されます。

注:

- cast 関数や複数の引数を組み込んだ関数は使用できません。
- 名前が、使用可能な SQL キーワードまたは OmniFind キーワードと競合する場合、名前を二重引用符で囲みます。

options

このストアード・プロシージャで使用できる各種のオプションを指定する文字ストリング。

このパラメーターのデータ・タイプは VARCHAR(32000) です。

options:

`text-default-information` `update-characteristics` `index-configuration-options`

text-default-information:

`CCSID—ccsid` `LANGUAGE— language` `FORMAT— format`

text-default-information

バイナリー・テキスト文書の索引を生成するときに使用するコード化文字セット ID を指定します。文書を処理するときに使用する言語と、列に格納されているテキスト文書の形式も指定します。

CCSID *ccsid*

バイナリー・データ・タイプの列のテキスト検索索引で使用するコード化文字セット ID を指定します。デフォルト値は 1208 (UTF-8) であり、QSYS2.SYSTEXTDEFAULTS 表から取り込まれます。このパラメーターでは、IBM i の変換サービスによる UTF-8 への変換でサポートされているすべての CCSID を指定できます。

バイナリー・データ・タイプ以外の列のテキスト検索索引でこのパラメーターを指定しても、その値は無視されます。テキスト列は、表の指定内容から CCSID を継承します。*format* 値を INSO に設定すると、*ccsid* 値は無視されます。

LANGUAGE *language*

DB2 for i OmniFind テキスト検索サーバーがテキスト文書の言語処理で使用する言語を指定します。デフォルト値は en_US (英語) です。このパラメーターの値として AUTO を指定すると、DB2 for i OmniFind テキスト検索サーバーが言語を判別しようとします。

重要: 文書の言語が英語でない場合は、デフォルト値 en_US を使用しないでください。この値を文書の言語に変更しないと、言語処理が正しく動作しません。

FORMAT *format*

列に存在するテキスト文書の形式 (HTML など) を指定します。DB2 for i OmniFind テキスト検索サーバーでは、索引生成と検索の対象にするテキスト文書の形式またはコンテンツ・タイプを判別する必要があります。*format* パラメーターを使用しない場合は、デフォルト値が QSYS2.SYSTEXTDEFAULTS 表の FORMAT 列から取り込まれます。サポートされている *format* 値は、TEXT、HTML、XML、INSO です。

format 値として INSO を使用すると、DB2 for i OmniFind テキスト検索サーバーが形式を判別します。その場合、*ccsid* 値は無視されます。DB2 for i OmniFind テキスト検索サーバーが文書フォーマットを判別できない場合は、SYSPROC.SYSTS_UPDATE ストアド・プロシージャによる処理中に文書エラーがジョブ・ログに記録されます。

注: XML データ・タイプの列で索引を作成するときに *format* パラメーターを指定しない場合のデフォルト値は XML です。*format* パラメーターで TEXT または INSO を指定すると、その索引では XML 検索機能を使用できなくなります。さらに、警告メッセージがジョブ・ログに書き込まれます。

update-characteristics:



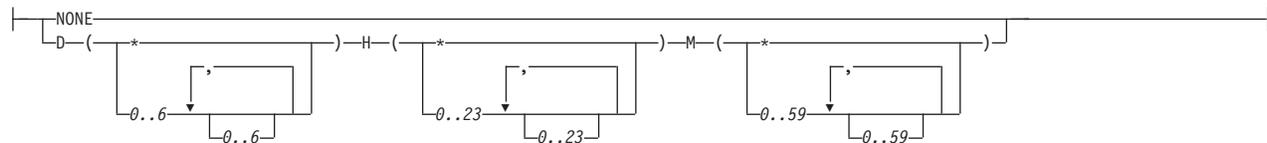
update-characteristics

テキスト検索索引の自動更新の頻度を指定します。さらに、指定のタイミングでテキスト検索索引のインクリメンタル更新を実行する前に、テキスト文書で発生していなければならない変更の最小数も指定します。

UPDATE FREQUENCY *update-frequency*

テキスト検索索引の自動更新を実行する頻度を指定します。デフォルト値は NONE です。このオプションは、今後変更が発生しないテキスト列で使用できます。*update-frequency* オプションでは、2 種類の形式を使用できます。

update-frequency (形式 1):



NONE NONE を指定すると、索引は更新されません。更新する場合は、手動で操作を開始する必要があります。このオプションは、今後変更する計画がないテキスト列で使用できます。

D 索引を更新する曜日 (複数可) を指定します。アスタリスク (*) を指定すると、すべての曜日が対象になります。0 を指定すると、日曜日が対象になります。

H 索引を更新する時刻 (複数可) を指定します。アスタリスク (*) を指定すると、毎時が対象になります。

M 索引を更新する分 (複数可) を指定します。アスタリスク (*) を指定することはできません。最小更新頻度は 5 分です。

例: 索引の更新を 30 分ごとに実行するように指定する例を以下に示します。

```
UPDATE FREQUENCY D(*) H(*) M(0,30)
```

update-frequency (形式 2、日時順):

|< minute>|< hour>|< dayOfMonth>|< monthOfYear>|< dayOfWeek>|

update-frequency (日時順) オプションは、5 つの値をブランク・スペースで区切ったリストの形式になります。その 5 つの値では、分、時、日、月、曜日 (起点は日曜日) をそれぞれ指定します。

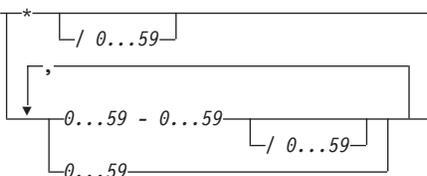
値の間隔またはアスタリスク (*) を指定する場合は、定義する間隔の末尾でスラッシュ (/) を使用してステップ値を指定することもできます。

例: 毎月、月曜日から金曜日に (* * 1-5)、午前 8 時から午後 6 時 45 分までの間の偶数の時刻 (8-18/2 は 8,10,12,14,16,18 と等価) を対象として、1 時間の 4 分の 1 ごとに (0,15,30,45) 索引の更新を実行するように指定する例を以下に示します。

```
0,15,30,45 8-18/2 * * 1-5
```

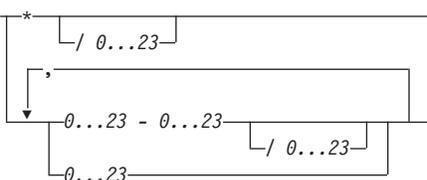
minute テキスト検索索引を更新する分を指定します。5 分ごとの間隔を意味するアスタリスク (*) を指定するか、0 から 59 の整数を指定できます。値を繰り返すことはできません。最小更新頻度は 5 分です。「1,4,8」という値は無効です。

update-frequency (minute):



hour テキスト検索索引を更新する時刻を指定します。毎時を意味するアスタリスク (*) を指定するか、0 から 23 の整数を指定できます。値を繰り返すことはできません。

update-frequency (hour):



dayOfMonth

テキスト検索索引を更新する日を指定します。毎日を意味するアスタリスク (*) を指定するか、1 から 31 の整数を指定できます。値を繰り返すことはできません。

update-frequency (dayOfMonth):



monthOfYear

テキスト検索索引を更新する月を指定します。毎月を意味するアスタリスク (*) を指定するか、1 から 12 の整数を指定できます。値を繰り返すことはできません。

update-frequency (monthOfYear):



dayOfWeek

テキスト検索索引を更新する曜日を指定します。毎日意味するアスタリスク (*) を指定するか、0 から 7 の整数を指定できます。0 と 7 はいずれも日曜日を意味する有効な値です。値を繰り返すことはできません。

update-frequency (dayOfWeek):



UPDATE MINIMUM *minchanges*

update-frequency オプションで指定するタイミングでテキスト検索索引のインクリメンタル更新を実行する前に、基礎表で発生していなければならないレコード変更の最小数を指定します。値は、1 から 2147483647 の整数でなければなりません。デフォルト値は、QSYS2.SYSTEXTDEFAULTS 表の UPDATEMINIMUM 列から取り込まれます。

テキスト検索索引を更新するときに、SYSPROC.SYSTS_UPDATE ストアド・プロシージャで USING UPDATE MINIMUM オプションを指定しなければ、このオプションは無視されます。

index-configuration-options:



index-configuration-options

追加の索引固有の値をオプション値の組として指定します。ストリング値は、単一引用符で囲む必要があります。ストリング値の中の単一引用符は、2 つの連続した単一引用符として記述しなければなりません。

CJKSEGMENTATION

CJK 言語 (中国語、日本語、韓国語) の文書の索引を生成するときに使用するセグメンテーション方式を指定します。サポートされている値は、MORPHOLOGICAL と NGRAM です。

CJKSEGMENTATION 値を指定しない場合は、デフォルト値が使用されます。デフォルト値は、QSYS2.SYSTEXTDEFAULTS 表の DEFAULTNAME 値として指定しておきます。

COMMENT

DB2 for i OmniFind テキスト検索サーバーコレクションに関する説明として QSYS2.SYSTEXTINDEXES 管理表の REMARKS 列に格納するコメントを指定します。

このオプションの値は、512 バイト以下のストリング値です。

IGNOREEMPTYDOCS

テキスト検索索引に空の文書を組み込むかどうかを指定します。空の文書とは、空ストリングまたはヌル値の文書です。

このオプションのサポート対象の値は、0 (ゼロ) と 1 です。デフォルト値は 1 です。

このオプションを 1 に設定すると、テキスト検索索引に空の文書は組み込まれません。このオプションを使用する場合に文書の内容を空に変更すると、次のインクリメンタル更新で、空に変更した文書がテキスト検索索引から削除されます。

KEYCOLUMN

テキスト索引のキー列として使用するユニーク列の名前を指定します。キー列を使用するのは、テキスト索引のデータと基本表の文書または行を関連付けるためです。主キー制約またはユニーク索引が設定されている列を指定する必要があります。KEYCOLUMN を指定しない場合は、表に ROWID 列が存在すればその列が使用されます。そうでなければ、表で定義されている主キーが使用されます。

SERVER

テキスト検索索引を格納するために使用するサーバーの ID または別名を指定します。ID を使用する場合は、QSYS2.SYSTEXTSERVERS カタログの SERVERID 列に存在する整数を値として指定する必要があります。別名を使用する場合は、QSYS2.SYSTEXTSERVERS カタログの ALIASNAME 列に存在するストリングを値として指定する必要があります。SERVER を指定しない場合は、デフォルトの動作として、QSYS2.SYSTEXTSERVERS 表で SERVERSTATUS パラメーターが 0 (ゼロ) に設定されているサーバー (つまり、使用可能なサーバー) のうち、テキスト検索索引の数が最も少ないサーバーが選択されます。

UPDATEAUTOCOMMIT

索引の更新中に文書を取り出すときに実行するコミット操作の頻度を指定します。値として 0 (ゼロ) を指定すると、処理の最後の時点でのみコミット操作が発生します。

値は、0 (ゼロ) と 2147483647 の間の整数でなければなりません。デフォルト値は 100 です。

パフォーマンスに関するヒント: UPDATEAUTOCOMMIT の値は、索引の更新処理のパフォーマンスに大きな影響を及ぼす可能性があります。指定の間隔でコミット操作を実行すれば、中断が発生した場合に索引の更新処理を再開するためのチェックポイントを一貫して確保できます。一方、コミットが発生すれば、更新処理が一時的に中断することにもなります。

UPDATEAUTOCOMMIT 値を大きくすると (あるいは 0 に設定すると)、特に初期更新の場合

に更新のパフォーマンスが高くなります。索引の更新処理の頻度に基づいて、パフォーマンスに関する要件とリカバリーに関する要件のバランスを考えながら、値を指定する必要があります。

options パラメーターのデフォルト値

DB2 for i OmniFind テキスト検索サーバーをインストールすると、QSYS2.SYSTEXTDEFAULTS 表が作成され、SYSPROC.SYSTS_CREATE ストアド・プロシージャの options パラメーターのデフォルト値が設定されます。

各オプションのデフォルト値と説明を以下の表にまとめます。

表 2. options パラメーターのデフォルト値

オプション	デフォルト値	説明
CCSID	1208	バイナリー・テキスト文書の索引を生成するときに使用するコード化文字セット ID を指定します。
CJKSEGMENTATION	NGRAM	CJK 言語 (中国語、日本語、韓国語) の文書の索引を生成するときに使用するセグメンテーション方式を指定します。
LANGUAGE	en_US	テキスト文書を処理するときに使用する言語を指定します。
FORMAT	TEXT	列に存在するテキスト文書の形式を指定します。データ・タイプが XML でない限り、デフォルトの形式はプレーン・テキストです。
UPDATEFREQUENCY	NONE	自動更新のスケジュールを設定しないことを指定します。
UPDATEMINIMUM	1	索引の最後の更新時以降、少なくとも 1 つの文書が変更されたら、SYSPROC.SYSTS_UPDATE ストアド・プロシージャの処理を開始します。
IGNOREEMPTYDOCS	1	空の文書 (空ストリングまたはヌル値の文書) をテキスト検索索引に組み込まないことを指定します。その種の文書のメタデータ・フィールドは、検索の対象になりません。
UPDATEAUTOCOMMIT	100	索引の更新中に文書を取り出すときに実行するコミット操作の頻度を指定します。
MINIMUMUPDATEINTERVAL	5	UPDATEFREQUENCY オプションの間隔を指定します。5 分より短い間隔を指定することはできません。
USEREXITTHREADS	0	予約済み

関連概念

6 ページの『サポート対象の文書フォーマット』
テキスト列のデータとしては、プレーン・テキスト、HTML 文書、XML 文書のほか、検索エンジンが認識できるあらゆる文書が対象になります。

8 ページの『サポート対象のデータ・タイプ』
索引を作成して検索を実行するテキスト列のデータは、バイナリー・データまたは文字データになります。

関連資料

110 ページの『QSYS2.SYSTEXTSERVERS 管理表』
QSYS2.SYSTEXTSERVERS 管理表では、テキスト検索サーバーのインストール先に関する情報を確認できます。

107 ページの『QSYS2.SYSTEXTINDEXES 管理表』
QSYS2.SYSTEXTINDEXES 管理表では、各テキスト検索索引の情報を確認できます。それぞれのテキスト検索索引ごとに、名前、スキーマ名、テキスト検索サーバーにある関連コレクション名が記述されています。

106 ページの『QSYS2.SYSTEXTDEFAULTS 管理表』
QSYS2.SYSTEXTDEFAULTS 管理表では、デフォルト・パラメーターとその値を確認できます。この表は、DB2 for i OmniFind テキスト検索サーバー のインストール時に作成されます。

37 ページの『SYSPROC.SYSTS_UPDATE』
SYSPROC.SYSTS_UPDATE ストアド・プロシージャを呼び出すことによって、テキスト列の現在の内容に合わせてテキスト検索索引を更新できます。

17 ページの『SYSPROC.SYSTS_START』
SYSPROC.SYSTS_START ストアド・プロシージャを呼び出すことによって、DB2 のテキスト検索機能を有効にできます。

9 ページの『サポート対象の言語』
特定の言語を使用してテキスト文書进行处理することを指定できます。

SYSPROC.SYSTS_ALTER

SYSPROC.SYSTS_ALTER ストアド・プロシージャを呼び出すことによって、SYSPROC.SYSTS_CREATE で作成した索引の属性を変更できます。変更できるのは、このプロシージャで明示的に指定した属性に限られます。索引の他の属性はすべて変更されません。

索引の作成後に索引の属性 (更新頻度など) を変更しなければならなくなった場合に、このプロシージャを使用できます。

前提条件

SYSPROC.SYSTS_ALTER ストアド・プロシージャを呼び出す前に、以下の前提条件を確認します。

- テキスト検索索引を (SYSPROC.SYSTS_CREATE ストアド・プロシージャの呼び出しによって) 作成しておく必要があります。

権限

ステートメントの許可 ID が保持する特権には、以下のうち最低 1 つの特権が含まれている必要があります。

- このスキーマでの作成特権。詳細については、『権限、特権、およびオブジェクト所有権』を参照してください。

- 管理権限

| ステートメントの許可 ID が保持する特権には、以下のうち最低 1 つの特権が含まれている必要があります。

| • 以下のシステム権限:

| - 論理ファイル作成 (CRTLF) コマンドに対する *USE

| - データ・ディクショナリーに対する *CHANGE (テキスト索引の作成先のライブラリーがデータ・ディクショナリー付きの SQL スキーマの場合)

| • 管理権限

| また、ステートメントの許可 ID が保持する特権には、以下のうちの最低 1 つの特権も含まれている必要があります。

| • 参照先の表に対する権限:

| - その表に対する INDEX 特権

| - その表が含まれているライブラリーに対する *EXECUTE システム権限

| • 管理権限

| • テキスト索引の作成先のライブラリーと同じ名前のユーザー・プロファイルが存在し、その名前がステートメントの許可 ID とは異なっている、という状況で SQL 名を指定する場合は、ステートメントの許可 ID が所有する特権に少なくとも以下のいずれかの権限が含まれている必要があります。

| - その名前のユーザー・プロファイルに対する *ADD システム権限

| - 管理権限

| 特殊タイプを参照する場合は、ステートメントの許可 ID が所有する特権に少なくとも以下のいずれかの特権が含まれている必要があります。

| • ステートメントで指定する特殊タイプごとに必要な権限:

| - その特殊タイプに対する USAGE 特権、および

| - その特殊タイプが含まれているライブラリーに対する *EXECUTE システム権限

| • 管理権限

| SQL 特権に対応するシステム権限については、『GRANT (表またはビュー特権)』を参照してください。

| 構文

| ►►SYSYS_ALTER(—*indexSchema*—,—*indexName*—,—*options*—)◄◄

| スキーマ修飾子は SYSPROC です。

| パラメーター

| *indexSchema*

| テキスト検索索引のスキーマを識別します。このパラメーターがヌルの場合、呼び出し側の CURRENT SCHEMA 特殊レジスターの値が使用されます。

| このパラメーターのデータ・タイプは VARCHAR(128) です。

| *indexName*

| テキスト検索索引の名前を識別します。索引スキーマと一緒にテキスト検索索引の名前を記述すると、DB2 サブシステムでテキスト検索索引を一意的に識別できます。このパラメーターには、ヌル以外の値を指定する必要があります。

| このパラメーターのデータ・タイプは VARCHAR(128) です。

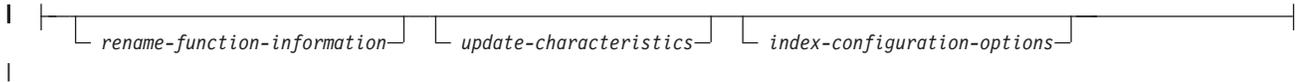
| *options*

| このストアード・プロシージャで使用できる各種のオプションを指定する文字ストリング。

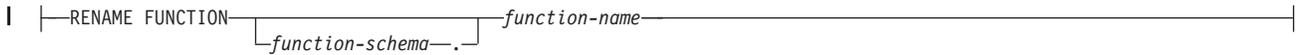
| このパラメーターのデータ・タイプは VARCHAR(32000) です。

| このパラメーターをヌルにすることはできません。

| **options:**



| **rename-function-information:**



| 名前を変更するユーザー定義関数を指定します。

| *function-schema.function-name*

| ユーザー定義関数のスキーマと名前を指定します。

| このオプションでは、索引の作成時に指定した関数を変更します。関数を変更すると、SYSTS_UPDATE では、テキスト列の索引作成時にその新しい関数を使用されるようになります。

| 関数を変更しても、SYSTS_UPDATE によって索引の既存のデータが変更されるわけではありません。最後の更新時以降に変更された新しいデータだけが処理されます。

| **update-characteristics:**



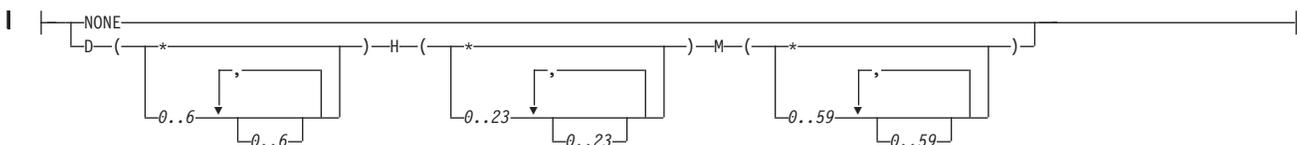
| **update-characteristics**

| テキスト検索索引の自動更新の頻度を指定します。さらに、指定のタイミングでテキスト検索索引のインクリメンタル更新を実行する前に、テキスト文書で発生していなければならない変更の最小数も指定します。

| **UPDATE FREQUENCY** *update-frequency*

| テキスト検索索引の自動更新を実行する頻度を指定します。デフォルト値は NONE です。このオプションは、今後変更が発生しないテキスト列で使用できます。update-frequency オプションでは、2 種類の形式を使用できます。

| **update-frequency (形式 1):**



| **NONE** NONE を指定すると、索引は更新されません。更新する場合は、手動で操作を開始する必要があります。このオプションは、今後変更する計画がないテキスト列で使用できます。

- | **D** 索引を更新する曜日 (複数可) を指定します。アスタリスク (*) を指定すると、すべての曜日が対象になります。0 を指定すると、日曜日が対象になります。
- | **H** 索引を更新する時刻 (複数可) を指定します。アスタリスク (*) を指定すると、毎時が対象になります。
- | **M** 索引を更新する分 (複数可) を指定します。アスタリスク (*) を指定することはできません。最小更新頻度は 5 分です。

| **例:** 索引の更新を 30 分ごとに実行するように指定する例を以下に示します。

```
| UPDATE FREQUENCY D(*) H(*) M(0,30)
```

| **update-frequency (形式 2、日時順):**



| *update-frequency* (日時順) オプションは、5 つの値をブランク・スペースで区切ったリストの形式になります。その 5 つの値では、分、時、日、月、曜日 (起点は日曜日) をそれぞれ指定します。

| 値の間隔またはアスタリスク (*) を指定する場合は、定義する間隔の末尾でスラッシュ (/) を使用してステップ値を指定することもできます。

| **例:** 毎月、月曜日から金曜日に (* * 1-5)、午前 8 時から午後 6 時 45 分までの間の偶数の時刻 (8-18/2 は 8,10,12,14,16,18 と等価) を対象として、1 時間の 4 分の 1 ごとに (0,15,30,45) 索引の更新を実行するように指定する例を以下に示します。

```
| 0,15,30,45 8-18/2 * * 1-5
```

| *minute* テキスト検索索引を更新する分を指定します。5 分ごとの間隔を意味するアスタリスク (*) を指定するか、0 から 59 の整数を指定できます。値を繰り返すことはできません。最小更新頻度は 5 分です。「1,4,8」という値は指定できません。

| **update-frequency (minute):**



| *hour* テキスト検索索引を更新する時刻を指定します。毎時を意味するアスタリスク (*) を指定するか、0 から 23 の整数を指定できます。値を繰り返すことはできません。

| **update-frequency (hour):**



dayOfMonth

テキスト検索索引を更新する日を指定します。毎日という意味のアスタリスク (*) を指定するか、1 から 31 の整数を指定できます。値を繰り返すことはできません。

update-frequency (dayOfMonth):



monthOfYear

テキスト検索索引を更新する月を指定します。毎月という意味のアスタリスク (*) を指定するか、1 から 12 の整数を指定できます。値を繰り返すことはできません。

update-frequency (monthOfYear):



dayOfWeek

テキスト検索索引を更新する曜日を指定します。毎日という意味のアスタリスク (*) を指定するか、0 から 7 の整数を指定できます。0 と 7 はいずれも日曜日を意味する有効な値です。値を繰り返すことはできません。

update-frequency (dayOfWeek):



UPDATE MINIMUM minchanges

update-frequency オプションで指定するタイミングでテキスト検索索引のインクリメンタル更新を実行する前に、基礎表で発生していなければならない記録変更の最小数を指定します。値は、1 から 2147483647 の整数でなければなりません。

index-configuration-options:



COMMENT

DB2 for i OmniFind テキスト検索サーバーコレクションに関する説明として
QSYS2.SYSTEXTINDEXES 管理表の REMARKS 列に格納するコメントを指定します。

このオプションの値は、512 バイト以下のストリング値です。

UPDATEAUTOCOMMIT

索引の更新中に文書を取り出すときに実行するコミット操作の頻度を指定します。値として 0 (ゼロ) を指定すると、処理の最後の時点でのみコミット操作が発生します。

値は、0 (ゼロ) と 2147483647 の間の整数でなければなりません。

パフォーマンスに関するヒント: UPDATEAUTOCOMMIT の値は、索引の更新処理のパフォーマンスに大きな影響を及ぼす可能性があります。指定の間隔でコミット操作を実行すれば、中断が発生した場合に索引の更新処理を再開するためのチェックポイントを一貫して確保できます。一方、コミットが発生すれば、更新処理が一時的に中断することにもなります。UPDATEAUTOCOMMIT 値を大きくすると (あるいは 0 に設定すると)、特に初期更新の場合に更新のパフォーマンスが高くなります。索引の更新処理の頻度に基づいて、パフォーマンスに関する要件とリカバリーに関する要件のバランスを考えながら、値を指定する必要があります。

関連資料

21 ページの『SYSPROC.SYSTS_CREATE』

SYSPROC.SYSTS_CREATE ストアド・プロシージャを呼び出すことによって、テキスト検索索引を作成できます。このストアド・プロシージャでは、テキスト列に基づいてテキスト検索索引を作成します。そのテキスト検索索引は、CONTAINS 関数または SCORE 関数を組み込んだ SQL 照会で使用できます。

37 ページの『SYSPROC.SYSTS_UPDATE』

SYSPROC.SYSTS_UPDATE ストアド・プロシージャを呼び出すことによって、テキスト列の現在の内容に合わせてテキスト検索索引を更新できます。

107 ページの『QSYS2.SYSTEXTINDEXES 管理表』

QSYS2.SYSTEXTINDEXES 管理表では、各テキスト検索索引の情報を確認できます。それぞれのテキスト検索索引ごとに、名前、スキーマ名、テキスト検索サーバーにある関連コレクション名が記述されています。

SYSPROC.SYSTS_DROP

SYSPROC.SYSTS_DROP ストアド・プロシージャを呼び出すことによって、

SYSPROC.SYSTS_CREATE ストアド・プロシージャで定義したテキスト検索索引を削除できます。

表を削除する前に、SYSPROC.SYSTS_DROP ストアド・プロシージャを使用してテキスト検索索引を削除することをお勧めします。

テキスト検索索引に対応するビューを削除すると、そのテキスト検索索引の削除が試行されます (ビューの削除が DROP TABLE CASCADE ステートメントの結果であっても、その動作は同じです)。ただし、コミットメント制御の下ではテキスト検索索引を削除できないので、コミットメント制御の下では SQL ビューも削除できないことになります。

テキスト検索サーバーにアクセスできない場合は、そのサーバーにあるコレクションが孤立する可能性があります。そうなった場合は、そのコレクションを手動で削除しなければなりません。そのサーバーが再び使用可能な状態になった時点で、DB2 for i OmniFind テキスト検索サーバーの管理ツールを使用して、そのサーバーにあるコレクションを削除してください。

74 ページの『管理ツール』では、孤立した索引を識別するためのツールや、孤立した索引を削除するための STSPROC.SYSTS_REMOVE ストアド・プロシージャに関する情報を確認できます。

前提条件

SYSPROC.SYSTS_DROP ストアド・プロシージャを呼び出す前に、以下の前提条件を確認します。

- SYSPROC.SYSTS_START ストアド・プロシージャを呼び出すことによって、DB2 のテキスト検索機能を開始しておく必要があります。
- テキスト検索索引を (SYSPROC.SYSTS_CREATE ストアド・プロシージャの呼び出しによって) 作成しておく必要があります。
- 削除するテキスト検索索引に対して、SYSPROC.SYSTS_CREATE、SYSPROC.SYSTS_UPDATE、SYSPROC.SYSTS_DROP の各ストアド・プロシージャが実行中の状態になっていないことを確認する必要があります。

権限

ステートメントの許可 ID が保持する特権には、以下のうち最低 1 つの特権が含まれている必要があります。

- 以下のシステム権限:
 - 削除対象のテキスト索引に対するシステム権限 *OBJOPR と *OBJEXIST
 - 削除対象のテキスト索引が含まれているライブラリーに対する *EXECUTE システム権限
- 管理権限

SQL 特権に対応するシステム権限については、『GRANT (表またはビュー特権)』を参照してください。

構文

```
SYSTS_DROP ( ( indexSchema , indexName ) )
```

Diagram description: The text shows the syntax for SYSTS_DROP. The parameter `indexSchema` is underlined and has a bracket pointing to a `null` value below it. The parameter `indexName` is also underlined. The entire parameter list is enclosed in parentheses, and the whole statement is preceded by `SYSTS_DROP` and followed by a long arrow pointing to the right.

スキーマ修飾子は SYSPROC です。

パラメーター

indexSchema

テキスト検索索引のスキーマを識別します。このパラメーターがヌルの場合、呼び出し側の CURRENT SCHEMA 特殊レジスターの値が使用されます。

このパラメーターのデータ・タイプは VARCHAR(128) です。

indexName

テキスト検索索引の名前を識別します。索引スキーマと一緒にテキスト検索索引の名前を記述すると、DB2 サブシステムでテキスト検索索引を一意的に識別できます。このパラメーターには、ヌル以外の値を指定する必要があります。

このパラメーターのデータ・タイプは VARCHAR(128) です。

関連資料

21 ページの『SYSPROC.SYSTS_CREATE』

SYSPROC.SYSTS_CREATE ストアド・プロシージャを呼び出すことによって、テキスト検索索引を作成できます。このストアド・プロシージャでは、テキスト列に基づいてテキスト検索索引を作成します。そのテキスト検索索引は、CONTAINS 関数または SCORE 関数を組み込んだ SQL 照会で使用できます。

17 ページの『SYSPROC.SYSTS_START』

SYSPROC.SYSTS_START ストアド・プロシージャを呼び出すことによって、DB2 のテキスト検索機能を有効にできます。

『SYSPROC.SYSTS_UPDATE』

SYSPROC.SYSTS_UPDATE ストアド・プロシージャを呼び出すことによって、テキスト列の現在の内容に合わせてテキスト検索索引を更新できます。

77 ページの『SYSPROC.SYSTS_REMOVE』

SYSPROC.SYSTS_REMOVE SQL ストアド・プロシージャを使用して、孤立した索引を削除できます。

SYSPROC.SYSTS_UPDATE

SYSPROC.SYSTS_UPDATE ストアド・プロシージャを呼び出すことによって、テキスト列の現在の内容に合わせてテキスト検索索引を更新できます。

テキスト検索索引の更新は大規模な操作になるので、テキスト検索索引は、DB2 表の更新と同時に更新されるわけではありません。DB2 表の列の変更は、トリガーによってステージング表に取り込まれます。テキスト検索索引が更新されるのは、SYSPROC.SYSTS_UPDATE ストアド・プロシージャの次の呼び出し時、または UPDATE FREQUENCY オプションで指定されている更新の時点になります。したがって、検索要求の結果に表の最新の更新内容が反映されない場合もあります。

このストアド・プロシージャは、DB2 for i OmniFind テキスト検索サーバー でテキスト検索索引のすべての更新処理が完了した後に戻ります。その所要時間は、SYSTS_UPDATE の呼び出し時にステージング表に存在していた項目の数によって異なります。更新処理中も、テキスト検索索引の検索は可能です。

基本表に含まれている文書の索引を作成するときに問題が発生すると、ステージング表の TOBEDELETED の列の値が E (エラー) または W (警告) に設定されます。それらのレコードを取得するには、以下の照会を組み込んだストアド・プロシージャを使用します。

```
SELECT s.TOBEDELETED, based_on_columns
FROM based_on_table t INNER JOIN QSYS2.stagingtables
ON (QQQ_TEXTSEARCH_KEY(t.k1, t.k2, t.k3, ...) = s.KEYID)
WHERE s.TOBEDELETED IN('E','W')
```

この場合、based_on_columns は、based_on_table で確認する必要がある列のリストです。based_on_table は、索引生成の対象になっている表です。staging table は、テキスト検索索引のカatalogで指定されているステージング表です。k1、k2、k3... は、テキスト検索索引を作成するために使用する主キー、行 ID、ユニーク・キーのいずれかになっているキー列のリストです。それらの文書のエラーを修正してから、更新を再度実行してください。

索引の更新中にその同じ索引で別の更新を要求すると、エラーが返されます。索引の更新は、1 つずつしか実行できません。

前提条件

- SYSPROC.SYSTS_UPDATE ストアド・プロシージャーを呼び出す前に、以下の前提条件を確認します。
- テキスト検索索引を (SYSPROC.SYSTS_CREATE ストアド・プロシージャーの呼び出しによって) 作成しておく必要があります。
- 更新するテキスト検索索引に対して、SYSPROC.SYSTS_CREATE、SYSPROC.SYSTS_UPDATE、SYSPROC.SYSTS_DROP の各ストアド・プロシージャーが実行中の状態になっていないことを確認する必要があります。
- SYSTS_START ストアド・プロシージャーを使用して、索引が存在するテキスト検索サーバーを開始しておく必要があります。つまり、QSYS2.SYSTEXTSERVERS の SERVERSTATUS 列の値が「0」(開始済み) になっていなければなりません。

権限

ステートメントの許可 ID が保持する特権には、以下のうち最低 1 つの特権が含まれている必要があります。

- このスキーマでの作成特権。詳細については、『権限、特権、およびオブジェクト所有権』を参照してください。
- 管理権限

ステートメントの許可 ID が保持する特権には、以下のうち最低 1 つの特権が含まれている必要があります。

- 以下のシステム権限:
 - 論理ファイル作成 (CRTLF) コマンドに対する *USE
 - データ・ディクショナリーに対する *CHANGE (テキスト索引の作成先のライブラリーがデータ・ディクショナリー付きの SQL スキーマの場合)
- 管理権限

また、ステートメントの許可 ID が保持する特権には、以下のうちの最低 1 つの特権も含まれている必要があります。

- 参照先の表に対する権限:
 - その表に対する INDEX 特権
 - その表が含まれているライブラリーに対する *EXECUTE システム権限
- 管理権限
- テキスト索引の作成先のライブラリーと同じ名前のユーザー・プロファイルが存在し、その名前がステートメントの許可 ID とは異なっている、という状況で SQL 名を指定する場合は、ステートメントの許可 ID が所有する特権に少なくとも以下のいずれかの特権が含まれている必要があります。
 - その名前のユーザー・プロファイルに対する *ADD システム権限
 - 管理権限

特殊タイプを参照する場合は、ステートメントの許可 ID が所有する特権に少なくとも以下のいずれかの特権が含まれている必要があります。

- ステートメントで指定する特殊タイプごとに必要な権限:
 - その特殊タイプに対する USAGE 特権、および
 - その特殊タイプが含まれているライブラリーに対する *EXECUTE システム権限

| • 管理権限

| SQL 特権に対応するシステム権限については、『GRANT (表またはビュー特権)』を参照してください。

| 構文

| ►► `SYSTS_UPDATE` (`indexSchema`, `indexName`, `options`)

|

| スキーマ修飾子は SYSPROC です。

| パラメーター

| *indexSchema*

| テキスト検索索引のスキーマを識別します。このパラメーターがヌルの場合、呼び出し側の CURRENT SCHEMA 特殊レジスターの値が使用されます。

| このパラメーターのデータ・タイプは VARCHAR(128) です。

| *indexName*

| テキスト検索索引の名前を識別します。索引スキーマと一緒にテキスト検索索引の名前を記述すると、DB2 サブシステムで全文索引を一意的に識別できます。このパラメーターには、ヌル以外の値を指定する必要があります。

| このパラメーターのデータ・タイプは VARCHAR(128) です。

| *options*

| このストアード・プロシージャで使用できるオプションを指定する文字ストリング。

| 使用可能なオプションは、USING UPDATE MINIMUM です。このオプションでは、SYSPROC.SYSTS_CREATE ストアード・プロシージャで指定した USING UPDATE MINIMUM 設定を使用します。変更数が指定の数に達した場合にのみ、インクリメンタル更新が開始されます。デフォルトは、更新処理を無条件で開始する、という動作です。

| **USING UPDATE MINIMUM:**

| `USING UPDATE MINIMUM`

|

関連概念

16 ページの『文書の切り捨て』
DB2 for i OmniFind テキスト検索サーバーでは、テキスト文書ごとに索引生成の対象にできる文字の数が制限されています。この文字の長さ制限によって、テキスト検索索引で大きなテキスト文書が切り捨てられることもあります。

関連資料

21 ページの『SYSPROC.SYSTS_CREATE』
SYSPROC.SYSTS_CREATE ストアド・プロシージャを呼び出すことによって、テキスト検索索引を作成できます。このストアド・プロシージャでは、テキスト列に基づいてテキスト検索索引を作成します。そのテキスト検索索引は、CONTAINS 関数または SCORE 関数を組み込んだ SQL 照会で使用できます。

17 ページの『SYSPROC.SYSTS_START』
SYSPROC.SYSTS_START ストアド・プロシージャを呼び出すことによって、DB2 のテキスト検索機能を有効にできます。

35 ページの『SYSPROC.SYSTS_DROP』
SYSPROC.SYSTS_DROP ストアド・プロシージャを呼び出すことによって、SYSPROC.SYSTS_CREATE ストアド・プロシージャで定義したテキスト検索索引を削除できます。

テキスト検索索引による検索

OMNIFINDでは、SQL ステートメントに CONTAINS 関数と SCORE 関数を組み込んで使用できます。CONTAINS と SCORE を使用すれば、指定する検索指数の基準に基づいてテキスト検索索引を検索できるようになります。さらに、XML 文書の索引生成や検索も可能です。

CONTAINS

CONTAINS 関数では、検索指数として指定する基準を使用して、テキスト検索索引を検索できます。この関数は、一致項目が見つかったかどうかを示す結果を返します。

▶▶ CONTAINS (*column-name* , *search-argument* [, *string-constant*]) (1)

注:

1 *string-constant* は、*search-argument* オプションの規則に準拠する必要があります。

search-argument-options:

(1)

QUERYLANGUAGE	=	value
RESULTLIMIT	=	value
SYNONYM	=	OFF ON

注:

1 同じ文節を複数回指定することはできません。

スキーマは QSYS2 です。

| *column-name*

| 検索するテキスト検索索引が付いている列の修飾名または非修飾名を指定します。その列は、ステートメントの FROM 文節で指定する表またはビューの中に存在している必要があります。その表の列、またはそのビューの基礎となる基本表の列には、テキスト検索索引が関連付けられていなければなりません (SQLSTATE 38H12)。ビューの列の基礎式は、基礎表の列を直接的に、または別のネスト・ビューによって参照する単純な列参照でなければなりません。

| *search-argument*

| 検索で使用する語を組み込んだストリング値を返す式を指定します。その式として、すべてブランクの式または空ストリングの式を使用することはできません (SQLSTATE 38H14)。ストリングの実際の長さが 32704 バイトを超えることはできません。その長さは、テキスト検索サーバーのサポート内容によってさらに制限される場合もあります (SQLSTATE 38H10)。その値は、テキスト検索索引の検索で使用される前に Unicode に変換されます。

| *string-constant*

| この関数で有効な検索索引オプションを指定したストリング定数を記述します。

| *search-argument-options* の一部として指定できるオプションは、以下のとおりです。

| **QUERYLANGUAGE = value**

| 照会言語を指定します。サポートされているいずれかの言語コードを *value* として指定できます。QUERYLANGUAGE オプションを指定しない場合のデフォルト値は、この関数を呼び出すときに使用するテキスト検索索引の言語値になります。テキスト検索索引の言語の値が AUTO である場合、QUERYLANGUAGE のデフォルト値は en_US です。

| **RESULTLIMIT = value**

| 基礎検索エンジンから返す結果の最大数を指定します。1 から 2,147,483,647 の整数値を *value* として指定できます。RESULTLIMIT オプションを指定しない場合は、この照会で結果の制限が有効になりません。

| 最適化プログラムが選択するプランによっては、このスカラー関数が結果表の行ごとに呼び出されない場合もあります。つまり、基礎検索エンジンに対する照会全体でこの関数が 1 回だけ呼び出される可能性があります。その場合は、一致するすべての主キーの結果セットが返されます。その結果セットは、結果行を識別するための列が含まれている表に結合されます。したがって、RESULTLIMIT 値は、基礎テキスト検索エンジンからの FETCH FIRST ?? ROWS のような動作になり、最適化の処理として RESULTLIMIT を使用することが可能になります。最適なプランとして、結果の行ごとに検索エンジンが呼び出される場合は、RESULTLIMIT オプションが無視されます。

| **SYNONYM = OFF** または **SYNONYM = ON**

| テキスト検索索引に関連付けられている同義語ディクショナリーを使用するかどうかを指定します。同義語ディクショナリーをコレクションに追加するには、同義語ツールを使用します。

| **OFF** OFF はデフォルト値です。

| **ON** テキスト検索索引に関連付けられている同義語ディクショナリーを使用します。

| この関数の結果は長精度整数になります。第 2 の引数をヌルにできる場合は、結果をヌルにできます。第 2 の引数をヌルにすると、結果がヌル値になります。

| 検索索引で指定した検索基準の一致項目が文書に含まれている場合は、結果が 1 になります。そうでない場合の結果は 0 です。列がヌルの場合の結果も 0 です。検索索引をヌルにすると、結果がヌル値になります。

| CONTAINS 関数は、常に同じ結果を返すとは限りません。

例 1

履歴書に「COBOL」が含まれているすべての従業員を検出するステートメントを以下に示します。

```
| SELECT EMPNO
| FROM EMP_RESUME
| WHERE RESUME_FORMAT = 'ascii'
| AND CONTAINS(RESUME, 'COBOL') = 1
```

例 2

検索指数は、ストリング定数でなくともかまいません。検索指数として、ホスト変数に含まれているストリングを組み込んだ SQL ストリング式を指定することもできます。COMMENT 列で「ate」という語の完全一致項目を検索するステートメントを以下に示します。

注: 「ate」という語の完全一致項目だけを検索し、言語処理上のバリエーションを考慮に入れない場合は、その語を二重引用符で区切る必要があります。

```
| char search_arg[100]; /* input host variable */
| ...
| EXEC SQL DECLARE C3 CURSOR FOR
| SELECT CUSTKEY
| FROM K55ADMIN.CUSTOMERS
| WHERE CONTAINS(COMMENT, :search_arg)= 1
| ORDER BY CUSTKEY;
| strcpy(search_arg, "%ate%");
| EXEC SQL OPEN C3;
| ...
```

例 3

「化石燃料」に相当するスペイン語の語句「combustible fósil」が含まれているオンライン・エッセイを書いた 10 人の学生をランダムに検出するステートメントを以下に示します。ここで選び出すのは、ラジオ・インタビューに出演する学生です。さらに、関連するテキスト検索索引のために作成した同義語ディクショナリーも使用します。必要な学生は 10 人だけなので、照会を最適化するために RESULTLIMIT オプションを使用して、基礎テキスト検索サーバーから抽出する結果の数を制限します。

```
| SELECT FIRSTNAME, LASTNAME
| FROM STUDENT_ESSAYS
| WHERE CONTAINS(TERM_PAPER, 'combustible fósil',
| 'QUERYLANGUAGE= es_ES RESULTLIMIT = 10 SYNONYM=ON') = 1
```

関連タスク

16 ページの『テキスト検索索引の検索』

CONTAINS 関数または SCORE 関数を組み込んだ SQL ステートメントを使用して、テキスト検索索引を検索できます。その関数で検索指数の基準を指定します。

関連資料

『SCORE』

SCORE 関数では、検索指数として指定する基準を使用して、テキスト検索索引を検索できます。この関数は、文書が照会内容に一致する比率を測定した関連度スコアを返します。

45 ページの『検索指数の構文』

テキスト文書に含まれている語を検索するための条件として検索指数を指定できます。検索指数には検索パラメーターと 1 つ以上の検索語を組み込みます。検索指数を使用する SQL スカラー・テキスト検索関数は、CONTAINS と SCORE です。

SCORE

SCORE 関数では、検索指数として指定する基準を使用して、テキスト検索索引を検索できます。この関数は、文書が照会内容に一致する比率を測定した関連度スコアを返します。

(1)

►► SCORE ((*column-name* , *search-argument* [, *string-constant*]))

注:

1 *string-constant* は、*search-argument* オプションの規則に準拠する必要があります。

search-argument-options:



注:

1 同じ文節を複数回指定することはできません。

スキーマは QSYS2 です。

column-name

検索するテキスト検索索引が付いている列の修飾名または非修飾名を指定します。その列は、ステートメントの FROM 文節で指定する表またはビューの中に存在する必要があります。その表の列、またはそのビューの基礎となる基本表の列には、テキスト検索索引が関連付けられていなければなりません (SQLSTATE 38H12)。ビューの列の基礎式は、基礎表の列を直接的に、または別のネスト・ビューによって参照する単純な列参照でなければなりません。

search-argument

検索で使用する語を組み込んだストリング値を返す式を指定します。その式として、すべて空白の式または空ストリングの式を使用することはできません (SQLSTATE 38H14)。ストリングの実際の長さが 32704 バイトを超えることはできません。その長さは、テキスト検索サーバーのサポート内容に

よってさらに制限される場合もあります (SQLSTATE 38H10)。その値は、テキスト検索索引の検索で使用される前に Unicode に変換されます。search-argument をヌルにすると、結果がヌル値になります。

string-constant

この関数で有効な検索引数オプションを指定したストリング定数を記述します。

search-argument-options の一部として指定できるオプションは、以下のとおりです。

QUERYLANGUAGE = value

照会言語を指定します。サポートされているいずれかの言語コードを value として指定できます。QUERYLANGUAGE オプションを指定しない場合のデフォルト値は、この関数を呼び出すときに使用するテキスト検索索引の言語値になります。テキスト検索索引の言語の値が AUTO である場合、QUERYLANGUAGE のデフォルト値は en_US です。

RESULTLIMIT = value

基礎検索エンジンから返す結果の最大数を指定します。1 から 2,147,483,647 の整数値を value として指定できます。RESULTLIMIT オプションを指定しない場合は、この照会で結果の制限が有効になりません。

最適化プログラムが選択するプランによっては、このスカラー関数が結果表の行ごとに呼び出されない場合もあります。つまり、基礎検索エンジンに対する照会全体でこの関数が 1 回だけ呼び出される可能性があります。その場合は、一致するすべての主キーの結果セットが返されます。その結果セットは、結果行を識別するための列が含まれている表に結合されます。したがって、RESULTLIMIT 値は、基礎テキスト検索エンジンからの FETCH FIRST ?? ROWS のような動作になり、最適化の処理として RESULTLIMIT を使用することが可能になります。最適なプランとして、結果の行ごとに検索エンジンが呼び出される場合は、RESULTLIMIT オプションが無視されます。

SYNONYM = OFF または SYNONYM = ON

テキスト検索索引に関連付けられている同義語ディクショナリーを使用するかどうかを指定します。同義語ディクショナリーをコレクションに追加するには、同義語ツールを使用します。

OFF OFF はデフォルト値です。

ON テキスト検索索引に関連付けられている同義語ディクショナリーを使用します。

この関数の結果は、倍精度浮動小数点数になります。第 2 の引数をヌルにできる場合は、結果をヌルにできます。第 2 の引数をヌルにすると、結果がヌル値になります。

検索引数で指定した検索基準の一致項目が列に含まれている場合は、0 より大きく 1 より小さい結果になります。一致項目が見つかる頻度が高ければ高いほど、結果の値は大きくなります。列に一致項目が含まれていない場合は、結果が 0 になります。列がヌルの場合のスコアも 0 です。

SCORE 関数は、常に同じ結果を返すとは限りません。

例

履歴書が programmer AND (java OR cobol) という基準を満たしている従業員のリストを生成し、その一致の比率に基づいて従業員を配列するステートメントを以下に示します。さらに、関連度の値を 0 (ゼロ) から 100 の値に正規化して返します。

```
SELECT EMPNO, INTEGER(SCORE(RESUME, 'programmer AND
(java OR cobol)') * 100) AS RELEVANCE
FROM EMP_RESUME
WHERE RESUME_FORMAT = 'ascii'
ORDER BY RELEVANCE DESC
```

関連タスク

16 ページの『テキスト検索索引の検索』

CONTAINS 関数または SCORE 関数を組み込んだ SQL ステートメントを使用して、テキスト検索索引を検索できます。その関数で検索指数の基準を指定します。

関連資料

40 ページの『CONTAINS』

CONTAINS 関数では、検索指数として指定する基準を使用して、テキスト検索索引を検索できます。この関数は、一致項目が見つかったかどうかを示す結果を返します。

21 ページの『SYSPROC.SYSTS_CREATE』

SYSPROC.SYSTS_CREATE ストアド・プロシージャを呼び出すことによって、テキスト検索索引を作成できます。このストアド・プロシージャでは、テキスト列に基づいてテキスト検索索引を作成します。そのテキスト検索索引は、CONTAINS 関数または SCORE 関数を組み込んだ SQL 照会で使用できます。

『検索指数の構文』

テキスト文書に含まれている語を検索するための条件として検索指数を指定できます。検索指数には検索パラメーターと 1 つ以上の検索語を組み込みます。検索指数を使用する SQL スカラー・テキスト検索関数は、CONTAINS と SCORE です。

検索指数の構文

テキスト文書に含まれている語を検索するための条件として検索指数を指定できます。検索指数には検索パラメーターと 1 つ以上の検索語を組み込みます。検索指数を使用する SQL スカラー・テキスト検索関数は、CONTAINS と SCORE です。

検索時の言語固有の処理については、検索指数のオプションとして QUERYLANGUAGE パラメーターの値を指定できます。サポートされているいずれかの言語コードを値として指定できます。

QUERYLANGUAGE パラメーターを指定しない場合のデフォルト値は、この関数を呼び出すときに使用するテキスト検索索引の言語値になります。

テキスト検索索引の言語の値が AUTO である場合、QUERYLANGUAGE のデフォルト値は en_US です。

制限

CONTAINS 関数と SCORE 関数を SQL 制約や索引定義で使用することはできません。SQL 照会ステートメントやビュー定義ではそれらの関数を使用できますが、以下の制約があります。

- ビュー、ネストされた表の式、共通表式で、スカラー関数 CONTAINS または SCORE の対象になるテキスト検索列を用意する場合に、その該当するビュー、ネストされた表の式、共通表式の最外部の SELECT ステートメントで DISTINCT 文節を使用するのであれば、SELECT リストに、テキスト検索索引のすべての対応するキー・フィールドを組み込む必要があります。その条件を満たさない場合は、SQL メッセージ 38H12 が返されます。

- ビュー、ネストされた表の式、共通表式で、スカラー関数 CONTAINS または SCORE の対象になるテキスト検索列を用意する場合は、その該当するビュー、ネストされた表の式、共通表式の最外部の SELECT レベルで、UNION、EXCEPT、INTERSECT の各ステートメントを使用できません。その条件を満たさない場合は、SQL メッセージ 38H12 が返されます。

- 共通表式で、スカラー関数 CONTAINS または SCORE の対象になるテキスト検索列を用意する場合には、その共通表式を照会全体の中で再度参照できるのは、その参照によって、スカラー関数 CONTAINS または SCORE の対象になるテキスト検索列を用意しない場合に限られます。その条件を満たさない場合は、SQL メッセージ 38H12 が返されます。
- スカラー関数 CONTAINS または SCORE をソースにした関数は作成できません。その条件を満たさない場合は、SQL メッセージ SQL0457 が返されます。
- 照会を実行するには、SQL Query Engine (SQE) を使用します。

単純な検索

単純なキーワード検索を実行する場合は、照会で 1 つ以上のキーワードを入力します。検索エンジンからは、そのすべてのキーワードまたはキーワードのバリエーションが含まれている文書が返されます。

例えば、「king」を入力すると、検索エンジンからは、「king」または「kings」という語が含まれているすべての文書が返されます。「king lear」という照会を入力すると、検索エンジンからは、「king」と「lear」の両方の語が含まれている文書が返されます。

検索結果を絞り込むには、より具体的なキーワードを使用します。例えば、「coffee」ではなく「French roast coffee」を使用するか、「Hawaiian vacations」ではなく「Kauai hiking tours」を使用する、といった具合です。

単純なキーワード検索によって対象外の文書があまりにも多く返される場合は、演算子を使用して検索を洗練できます。

検索語の除外

語を除外するには、負符号 (-) を使用します。例えば、「lear」という語が含まれていて、「edward」が含まれていない文書を検索する場合は、「lear -edward」という照会を入力します。

負符号 (-) の適用対象は、語とその変化形です。例えば、「-edward」という照会を使用すると、「edward's」という語が含まれている文書も除外されます。

フレーズ検索

複数の検索語が入力の順序のとおりにならんでいるフレーズを検索対象にする場合は、二重引用符を使用します。例えば、「king lear」というフレーズの完全一致だけを検索し、「kingly lear」や「king and queen lear」などの関連フレーズを除外する場合は、「"king lear"」と入力します。この検索では大/小文字の区別はありませんが、語の変化形は一致項目と見なされません。

検索で使用するワイルドカード文字

完全なスペルがわからない検索語や検索語のバリエーションが含まれている文書を探す場合は、ワイルドカード文字 (*) を使用できます。例えば、「czech*」という照会を使用すると、「czech」、「czechoslovakia」、「czechoslovakian」、「czech republic」などの語句が含まれている文書が返されます。

フレーズ検索でワイルドカード文字を使用することもできます。例えば、「"John * Kennedy"」という照会を使用すると、「John Fitzgerald Kennedy」や「John F Kennedy」というフレーズが含まれている文書が返されますが、「John Kennedy」が含まれている文書は返されません。「Mi*1 Gorbachev」という照会からは、「Mikhail Gorbachev」が返されます。

「*zech」のように照会の先頭にワイルドカード文字を追加すると、検索エンジンから結果が返されるまでの時間が長くなる可能性があります。

複数の語のうちのいずれか 1 つを対象にした検索

論理演算子 OR を指定すると、照会で使用する語のいずれか 1 つが含まれている文書が返されます。例えば、「(othello OR otello)」という照会を使用すると、「othello」または「otello」のいずれかの語が含まれている文書が返されます。

括弧を使用して、論理演算子の AND、OR、NOT を組み合わせて指定することもできます。例えば、「cougar OR (jaguar AND NOT car)」という照会を使用すると、「cougar」が含まれているという条件、または「car」が含まれていなくて「jaguar」が含まれているという条件のいずれかを満たす文書が返されます。

論理演算子の AND、OR、NOT はすべて大文字で入力しなければなりません。グループを指定する場合は、括弧を使用します。

関連概念

50 ページの『XML 検索』

XML 文書の索引生成や検索が可能です。XML 検索の構文では、テキスト検索用の拡張機能を追加した W3 XPath 言語のサブセットを使用します。その拡張機能によって、XML の属性や要素に関連した数値、日付値、日時値の範囲検索を実行できます。構造要素を別個に使用することも、照会の中でフリー・テキストと組み合わせて使用することも可能です。

関連資料

40 ページの『CONTAINS』

CONTAINS 関数では、検索索引として指定する基準を使用して、テキスト検索索引を検索できます。この関数は、一致項目が見つかったかどうかを示す結果を返します。

43 ページの『SCORE』

SCORE 関数では、検索索引として指定する基準を使用して、テキスト検索索引を検索できます。この関数は、文書が照会内容に一致する比率を測定した関連度スコアを返します。

単純な照会の例

CONTAINS 関数と SCORE 関数を使用した単純な照会では、テキスト検索索引に含まれている 1 つまたは複数の語を検索します。

検索エンジンは、文字間の空白文字を無視します。検索ストリングを空ストリングにしたり、すべてブランクにしたりすることはできません (SQLSTATE 38H14)。

単純な検索照会の例を以下の表にまとめます。

表 3. 単純な照会の例

検索語のタイプ	例	照会結果
1 つの単語	king	「king」または「kings」という語が含まれているすべての文書が返されます。この照会では、ルート語「king」のさまざまな言語処理上のバリエーションが一致項目になります。大/小文字の区別はありません。

表 3. 単純な照会の例 (続き)

検索語のタイプ	例	照会結果
複数の単語	king lear	「king」と「lear」が両方含まれているすべての文書が返されます。デフォルトの演算子は、論理演算子 AND です。

どの照会でも、演算子 **AND** と **+** が暗黙的に使用されます。例えば、「King Lear」という照会からは、「King AND Lear」や「King + Lear」と同じ結果が返されます。

論理演算子の **NOT**、**AND**、**OR** はすべて大文字で入力しなければなりません。

関連資料

40 ページの『CONTAINS』

CONTAINS 関数では、検索引数として指定する基準を使用して、テキスト検索索引を検索できます。この関数は、一致項目が見つかったかどうかを示す結果を返します。

43 ページの『SCORE』

SCORE 関数では、検索引数として指定する基準を使用して、テキスト検索索引を検索できます。この関数は、文書が照会内容に一致する比率を測定した関連度スコアを返します。

拡張検索演算子

拡張検索演算子を使用して、CONTAINS 関数と SCORE 関数の検索結果をさらに洗練できます。

以下の表の第 1 列では、検索照会で使用できる演算子を列挙しています。論理演算子の NOT、AND、OR はすべて大文字で入力しなければなりません。第 2 列では、入力できるサンプル照会を示しています。第 3 列では、照会例から得られる結果のタイプについて説明しています。

表 4. 拡張検索演算子と複雑な照会の例

演算子	例	照会結果
AND	"King Lear" AND "Othello" "King Lear" "Othello"	どちらの照会からも、「King Lear」と「Othello」の両方の語が含まれている文書が返されます。AND 演算子は、デフォルトの結合演算子です。2 つの語の間に論理演算子が存在しない場合は、AND 演算子が使用されます。例えば、「King Lear」という照会は、「King AND Lear」という照会と同じ意味になります。
OR	"King Lear" OR Lear	「King Lear」または「Lear」が含まれている文書が返されます。OR 演算子は、2 つの語をリンクして、どちらかの語が含まれている文書を一致項目として検出します。
NOT	"King Lear" NOT "Norman Lear"	「King Lear」が含まれていて、「Norman Lear」が含まれていない文書が返されます。

表 4. 拡張検索演算子と複雑な照会の例 (続き)

演算子	例	照会結果
" "	第 1 の照会: "King Lear" 第 2 の照会: "king"	第 1 の照会からは、「King Lear」という語句の完全一致項目が返されます。 第 2 の照会からは、「king」という語だけが返され、「kings」や「kingly」などの語は返されません。
*	test* te*t	「test」、「tests」、「tester」などと、「test」、「text」などの組み合わせが含まれている文書が一致項目として返されます。
^	第 1 の照会: "King Lear"^4 "Richard III" 第 2 の照会: title: (software download)^5 pdf viewer -shipping	第 1 の照会では、「King Lear」という語句が含まれている文書が検索結果リストの高い位置に強制的に表示されます。 第 2 の照会では、「software download」というタイトルの文書が結果リストの高い位置に強制的に表示されます。 ランキング調整係数は、正の数でなければなりません。1 未満の数 (0.2 など) も指定できます。ランキング調整係数に制限はありません。
+	+Lear King	「Lear」と「King」が含まれているすべての文書が返されます。「Lear AND King」という照会と同じ意味になります。
-	"King Lear" -"Lear Jet"	「King Lear」が含まれていて、「Lear Jet」が含まれていない文書が返されます。
()	(King OR Lear) AND plays	「King」または「Lear」のいずれかと「plays」が含まれている文書が返されます。この括弧を使用することによって、「plays」が存在し、さらに「King」または「Lear」のいずれかの語が存在する文書を検出できます。
\	¥(1¥+1¥)¥:2	「(1+1):2」が含まれている文書が返されます。特殊文字を、照会構文の中で文字そのものとして使用するには、¥ をエスケープ文字として使用します。特殊文字は、+、-、&&、 、!、(、)、{、}、[、]、^、"、~、*、?、:、¥ です。特殊文字の前にエスケープ文字を付けると、特殊文字が照会の一部と見なされ、分析の対象になります。

関連資料

40 ページの『CONTAINS』

CONTAINS 関数では、検索引数として指定する基準を使用して、テキスト検索索引を検索できます。この関数は、一致項目が見つかったかどうかを示す結果を返します。

43 ページの『SCORE』

SCORE 関数では、検索引数として指定する基準を使用して、テキスト検索索引を検索できます。この関数は、文書が照会内容に一致する比率を測定した関連度スコアを返します。

CONTAINS 関数と SCORE 関数の使用例

同じ照会で CONTAINS 関数と SCORE 関数の両方を使用できます。その照会は、テキスト検索索引を検索し、テキスト文書が検索引数の基準に一致するかどうか、一致する場合はその一致の頻度がどれほどかを示す値を返します。

以下の例では、基本表 BOOKS のデータを使用します。その基本表には、ISBN (VARCHAR(20))、ABSTRACT (VARCHAR(10000))、PRICE (INTEGER) の各列が含まれています。

表 5. 基本表 BOOKS

ISBN	ABSTRACT	PRICE
i1	"a b c"	7
i2	"a b d"	10
i3	"a e a"	8

以下の照会を実行します。

```
SELECT ISBN, SCORE(ABSTRACT,'"b"')
FROM BOOKS
WHERE CONTAINS (ABSTRACT,'"b"') = 1
```

この照会からは、以下の 2 つの行が返されます。

```
i1, 0.3
i3, 0.4
```

スコア値は、テキスト列の内容によって異なる可能性があります。

関連資料

40 ページの『CONTAINS』

CONTAINS 関数では、検索引数として指定する基準を使用して、テキスト検索索引を検索できます。この関数は、一致項目が見つかったかどうかを示す結果を返します。

43 ページの『SCORE』

SCORE 関数では、検索引数として指定する基準を使用して、テキスト検索索引を検索できます。この関数は、文書が照会内容に一致する比率を測定した関連度スコアを返します。

XML 検索

XML 文書の索引生成や検索が可能です。XML 検索の構文では、テキスト検索用の拡張機能を追加した W3 XPath 言語のサブセットを使用します。その拡張機能によって、XML の属性や要素に関連した数値、日付値、日時値の範囲検索を実行できます。構造要素を別個に使用することも、照会の中でフリー・テキストと組み合わせて使用することも可能です。

XML 照会構文を使用して索引を検索するには、まず文書の索引を作成して XML マークアップを組み込む必要があります。そのためには、文書索引作成時に「FORMAT XML」オプションを使用して索引を作成します。

以前のリリースで作成した索引を使用して検索を実行することも可能です。ただし、以前のリリースで索引を作成した文書には、新しいリリースで使用できるすべての XML 検索機能で必要な情報が含まれていないわけではありません。新しいリリースへのアップグレード後にテキスト検索索引で追加/更新した文書には、その追加情報が組み込まれています。

アップグレード後に、以前のリリースで索引を作成した文書が一部の検索結果に組み込まれなくなる、という可能性もあります。SYSPROC.SYSTS_REPRIMEINDEX ストアド・プロシージャを使用すれば、索引を再作成してこの問題を解決できます。

OMNIFIND の CONTAINS および SCORE 組み込み関数を使用して XML データを検索する場合は、照会ストリングの先頭に @xmlxp: という照会接頭部を付ける必要があります。この接頭部の後に、有効な XML 検索照会式を記述します。@xmlxp というあいまい条件接頭部は、照会パス式を使用して検索が実行されることを示す指標になります。

例えば、CONTAINS(columnname, '@xmlxp: 'query_expression''') という照会があるとします。

query_expression を囲む単一引用符 ' ' は、SQL ストリングの中に入っており、いわばストリング内のストリングになっているので、2 つ連続して記述する必要があります。

以前のリリースの DB2 for i OmniFind テキスト検索サーバーで使用していた @xpath: というあいまい条件接頭部は、旧バージョンとの互換性を確保するために現在でもサポートされています。ただし、非推奨扱いになっているので、使用しないでください。

XML 検索の主な機能を以下のリストにまとめます。

XML 構造の検索

XML の特殊なあいまい条件を照会に組み込むことによって、XML 文書の構造要素とその構造要素のスコープに含まれているテキストを検索できます。ここでいう構造要素とは、タグ名、属性名、属性値です。要素名とタグ名には大/小文字の区別があります。

XML 照会のトークン化

トークン化とは、入力を解析してトークンに変換する処理のことです。XML の照会条件の中に含まれているフリー・テキストは、XML 以外の照会条件の場合と同じ要領でトークン化されます。ただし、ネストしたあいまい条件はサポートされない、という例外があります。フリー・テキスト検索では、大/小文字の区別はありません。

XML スキーマと DTD

XML 文書に関連した XML スキーマは、ダウンロードされません。デフォルト値も索引作成の対象になりません。

数値

属性または要素の値を数値と比較する述部がサポートされています。

要素値

要素値を数値または日付値と比較する述部がサポートされています。日付値または数値が入っている要素は、日付値または数値だけが入っている XML 要素でなければなりません。前後の空白文字は無視されます。

| スtring値

述部のString引数で = 演算子を使用する場合は、指定するテキスト・スパンの各トークンがStringのすべてのキーワードと一致する完全一致項目が存在しなければなりません。突き合わせの実行時に、トークンの順序は意味を持ちません。

| 日時値

日付または日時の属性/要素を比較する述部がサポートされています。

| パス式:

表 6. パス式

@xmlxp 式	説明
TagName	TagName という名前のタグとそのタグのすべての子を選択します。
@AttributeName	@AttributeName という名前の属性を選択します。
/	ルート・ノードから選択します。
//	現在位置の下位にあるタグと属性から、式に一致するタグと属性を選択します。
.	自身: 現在のタグまたは要素のノード。

表 7. パス式の例:

@xmlxp 式	結果
/Document	最上位タグが Document になっているすべての文書を返します。
//Document	いずれかのレベルのタグが Document になっているすべての文書を返します。
/Document/Child1	最上位タグが Document になっていて、その直下の子タグが Child1 になっているすべての文書を返します。
/Document//Child1	最上位タグが Document になっていて、そのいずれかのレベルの下位タグが Child1 になっているすべての文書を返します。
/Root/@attr1	最上位タグが Root になっていて、その属性が attr1 になっているすべての文書を返します。
/Root//@attr1	最上位タグが Root になっていて、そのルート・タグまたはいずれかの下位タグの属性が attr1 になっているすべての文書を返します。
//@attr1	いずれかのレベルの属性が @attr1 になっているすべての文書を返します。

注: XML 検索式では、相対パス式の中に実際のタグ名または属性名を組み込む必要があります。/ と // だけでは有効な検索照会になりません。

1 つの軸だけに沿った順方向のパス式だけが有効です。

パス式のワイルドカード・サポート

パス式では、特殊なワイルドカード文字 * を使用して、任意の名前の 1 つのタグを対象として指定できます。

パス式の末尾ワイルドカードは無視されます。

パス式のワイルドカードの以下のような使用方法は、サポートされていません。そのような方法で使用する
と、エラーになります。

- ワイルドカードだけを参照し、具体的な要素や属性を参照していない式。

- あらゆるレベルでのワイルドカードの属性 (/Tag/@* など)。

- 述部式の直前のワイルドカード (/Root/*[//anytag] など)。

- 述部比較で使用するワイルドカード (/Root[* > 5] など)。

- XML 名前空間接頭部として使用するワイルドカード (//*:tagname など)。

- XML 名前空間接頭部の後のワイルドカード (//ns:* など)。

- タグ名の一部として使用するワイルドカード文字 (/start* など)。

表 8. パス式でワイルドカードを使用する例:

@xmlxp 式	結果
/Root/*/T1	最上位タグが Root になっていて、1 つの中間レベルの下位タグが T1 になっているすべての文書。
/Root/*/T1	最上位タグが Root になっていて、1 つ以上の中間レベルの下位タグが T1 になっているすべての文書。

述部

述部では、要素または属性のノードで満たさなければならない条件または値を指定します。述部は常に、大括弧 [] で囲みます。

表 9. 述部の例:

@xmlxp 式	結果
/Book[Sentences]	最上位タグが Book で、直下の子 Sentences が必要です。
/Book[.//Sentences and .//Author]	最上位タグが Book で、下位タグとして Sentences と Author の両方が必要です。それぞれの下位タグは、Book の下のどのレベルに存在していてもかまいません。

パス式は常に順方向であり、1 回のアクセスに限定されているので、述部の中で記述するパス式は、現在のノードを基準にした相対パス式でなければなりません。例えば、/Book[/Root] と /Book[//Root] は、どちらの場合も述部のパス式が現在のノードではなく最上位タグ「Root」から始まることになるので、無効です。

数値の比較

OMNIFIND では、要素および属性を、整数値および浮動小数点数値と比較するための演算子として、=、

<=、>=、>、<、!= がサポートされています。

単純要素の場合は、数値だけが索引作成の対象になります。要素に追加の文字 (空白文字以外) が含まれていてはならず、下位要素が存在していてもなりません。複合要素の場合は、テキストとしてのみ索引が作成されます。

表 10. 数値の比較の例:

@xmlxp 式	結果
/Book[@id_num = 12345]	最上位タグが Book で、id_num 属性の値が 12345 になっていることが必要です。
/Book[Cost <= 100.50]	最上位タグが Book です。Book の直下の子要素 Cost の数値が 100.50 以下になっていることが必要です。

日付と日時の比較

OMNIFIND では、要素および属性を、日付値および日時値と比較するための演算子として、=、<=、>=、>、<、!= がサポートされています。

単純要素の場合は、日時値だけが索引作成の対象になります。要素に追加の文字 (空白文字以外) が含まれていてはならず、下位要素が存在していてもなりません。複合要素の場合は、テキストとしてのみ索引が作成されます。

索引作成時には、単純 XML タグに含まれている属性値とテキストが検証されます。テキストが ISO の日付形式または日時形式に合致していると判断されると、そのテキストの索引が日付または日時として作成され、述部で検索できるようになります。

検索時には、関数呼び出し xs:date() または xs:dateTime() の中で日付値または日時値を囲んで、それらの値が正しいデータ・タイプとして認識されるようにする必要があります。

XML 文書で使用する XML 日時データ・タイプでは、時間帯の値を指定できます。ただし、日時の索引作成時には、テキスト検索サーバーによって時間帯の値が切り捨てられます。したがって、日付または日時のデータ・タイプに関する XML 検索では、時間帯は対象になりません。

さらに、24 という時刻の日時値を指定できるのは、分と秒がゼロの場合に限られます。その値は、当日の最後の時点と翌日の最初の時点の間の値と見なされます。

XML 検索述部で日付値または日時値を指定するときに時間帯を組み込むと、構文エラーになります。

日時データ・タイプでは、秒の小数点以下の桁数として最大 12 桁がサポートされています。

表 11. 日付と日時の比較の例:

@xmlxp 式	結果
/Book[@publishDate > xs:date("2000-01-01")]	最上位タグが Book です。Book の publishDate 属性の値が 2000-01-01 より大きい日付値になっていることが必要です。
/Book[purchaseTime > xs:dateTime("2009-05-20T13:00:00")]	最上位タグが Book です。Book の直下の子 purchaseTime の値が 2009-05-20T13:00:00.000000 より大きい日時式になっていることが必要です。

XML マークアップを対象にした *contains* と *excludes*

contains 関数と *excludes* 関数を使用して、XML マークアップを対象にしたフルテキスト検索を実行できます。*contains* から true が返されるのは、ターゲット・ノードに照会内容が含まれている場合であり、*excludes* から true が返されるのは、ターゲット・ノードに照会内容が含まれていない場合です。

例えば、`email` という最上位タグの `body` という直下の下位タグに「Department budget」というフレーズのバリエーションが含まれているすべての文書を検索する場合は、以下のようにします。

```
@xnxo:'/email[body contains ("department budget")]'
```

contains 関数または *excludes* 関数に渡すフリー・テキストは、他のあらゆるフリー・テキスト検索の場合と同じ要領で処理されます。この検索では大/小文字の区別はありません。言語処理上のバリエーションも対象になります。先ほどの照会では、「departments budgets」も「budget for the department」も一致項目になります。

従来の引用符を使用して、検索結果を完全一致項目に限定することも可能です。例えば、`@xmlxp:'/email[body contains("department budget")]'` のように記述します。完全一致を意味する引用符が *contains* のフリー・テキスト・ストリングの末尾と解釈されないようにするには、その引用符を 2 つ連続して記述する必要があります。

表 12. *contains* と *excludes* の例:

@xmlxp 式	結果
<code>/Book[abstract contains("cat AND dog")]</code>	最上位タグ Book の子タグ abstract に cat と dog という語の両方の言語処理上のバリエーションが含まれていることが必要です。
<code>/Book[abstract contains("cat AND dog")]</code> <code>/Book/@title[. contains("cat OR dog")]</code>	最上位タグ Book の title 属性に cat または dog の言語処理上のバリエーションが含まれていることが必要です。
<code>/Book/Title[. contains("All good dogs go to heaven")]</code>	最上位タグ Book の直下の子 Title に all good dogs go to heaven という語句がその順序で含まれていることが必要です。言語処理上のバリエーションは対象になりません。
<code>/Book[abstract excludes("cat AND dog")]</code>	最上位タグ Book の子タグ abstract に cat と dog という語の両方の言語処理上のバリエーションが含まれていないことが必要です。

ストリング完全一致演算子

述部のストリング引数で = 演算子を使用する場合は、指定するテキスト・スパンのすべてのトークンがストリングのすべてのトークンと一致する完全一致項目が存在しなければなりません。言語処理上のバリエーションは対象になりません。検索対象語の順序は意味を持ちません。要素または属性に検索対象テキストだけが含まれていなければならない、という要件はありません。

表 13. ストリング完全一致演算子の例:

@xmlxp 式	結果
<code>/Book[@author = "Nicholas Lawrence"]</code>	最上位タグが Book で、属性が author になっていることが必要です。author には、Nicholas Lawrence という語句が入っていないとなりません。それらの語句の言語処理上のバリエーションは、一致項目と見なされません。

表 13. スtring完全一致演算子の例: (続き)

@xmlxp 式	結果
/Book[author = ""Nicholas Lawrence""]	最上位タグが Book で、直下の下位タグが author になっていることが必要です。author には、Nicholas Lawrence という語句がその順序で入っていなければなりません。それらの語句の言語処理上のバリエーションは、一致項目と見なされません。

論理演算子

述部では、論理演算子の *and* と *or* を使用できます。

表 14. 論理演算子の例:

@xmlxp 式	結果
/Book[@author = ""Nicholas Lawrence""]/Price[. < 1000 and @unit = "dollars"]	<p>最上位タグが Book で、属性が author になっていることが必要です。author には、Nicholas Lawrence という語句がその順序で入っていなければなりません。それらの語句の言語処理上のバリエーションは、一致項目と見なされません。</p> <p>Book の直下の子 Price の値が 1000 より小さい値になっていることが必要です。さらに、Price ノードの @unit 属性の値が dollars になっていることも必要です。</p>

演算子優先順位

XML 検索述部では、包含演算子と比較演算子が論理演算子よりも優先されます。すべての論理演算子の優先順位は同じです。

- 包含演算子は、*contains* と *excludes* です。
- 比較演算子は、*=*、*!=*、*<*、*>*、*<=*、*>=* です。
- 論理演算子は、*and* と *or* です。

優先順位を確実に指定するために括弧を使用することもできます。

関連資料

45 ページの『検索索引数の構文』

テキスト文書に含まれている語を検索するための条件として検索索引数を指定できます。検索索引数には検索パラメーターと 1 つ以上の検索語を組み込みます。検索索引数を使用する SQL スカラー・テキスト検索関数は、CONTAINS と SCORE です。

79 ページの『SYSPROC.SYSTS_REPRIMEINDEX』

SYSPROC.SYSTS_REPRIMEINDEX ストアード・プロシージャを使用すれば、索引を再作成して初期更新を開始できます。基本表からデータを復元する場合に、このストアード・プロシージャを使用します。

40 ページの『CONTAINS』

CONTAINS 関数では、検索索引数として指定する基準を使用して、テキスト検索索引を検索できます。この関数は、一致項目が見つかったかどうかを示す結果を返します。

43 ページの『SCORE』

SCORE 関数では、検索索引数として指定する基準を使用して、テキスト検索索引を検索できます。この関数は、文書が照会内容に一致する比率を測定した関連度スコアを返します。

XML 検索の名前空間のサポート

文書では、要素と属性のスコープを定義するために名前空間を使用できます。名前空間を使用すれば、照会検索の結果を文書内の意味のある要素に限定できます。

概要

XML では、開発者が要素名と属性名を選択します。別々のアプリケーションに由来する XML 文書を混合すると、名前の競合が生じる場合があります。

したがって、特にさまざまな文書タイプの索引が作成されている可能性がある場合は、照会検索の結果を文書内の意味のある要素に限定するのが望ましいといえます。そのようにして検索結果を限定するために使用できるのが、名前空間です。

名前空間によって、文書の要素と属性のスコープを定義すれば、値を正しく解釈できるようになります。名前空間を記述するときには、ロング・ネーム (URI) を使用します。オプションとして、Qname (修飾名) というショート・ネームを使用することもできます。

```
<?xml version='1.0'?>
<doc xmlns:x="http://example.com/ns/abc">
  <x:p/>
</doc>
```

http://example.com/ns/abc が名前空間のロング・ネームであり、x が Qname 接頭部です。Qname 接頭部は、各要素参照の名前空間の短縮名として使用できます。

要素 p は、http://example.com/ns/abc 名前空間で修飾されます。

デフォルトの名前空間

XML 要素のデフォルトの名前空間を指定できます。デフォルトの名前空間は、現在のタグとその下位タグに適用されます。名前空間に含まれている非修飾タグは、デフォルトの名前空間を継承します。

```
<?xml version='1.0'?>
<doc xmlns="http://example.com/ns/abc">
  <p/>
</doc>
```

この場合は、doc 要素と p 要素の両方が http://example.com/ns/abc 名前空間にあります。

属性の名前空間

属性の名前空間と属性に関連する要素の名前空間が異なる場合もあります。

要素と属性 (修飾):

```
<dog xmlns:an="http://example.org/animals" xmlns:sz="http://example.org/sizes">
  <an:breed sz:size="Medium">Mutt</an:breed>
</dog>
```

名前空間を明示的に指定しない場合に、要素と属性が名前空間を継承する方法には、それぞれ違いがあります。非修飾の要素は、自身が存在するスコープのデフォルトの名前空間を継承します。非修飾の属性は、名前空間を継承しません。

要素と属性 (非修飾):

```
<dog xmlns:an="http://example.org/animals">
  <breed size="Medium">Mutt</an:breed>
</dog>
```

この例では、breed 要素は、http://example.org/animals 名前空間を継承します。一方、size 属性には名前空間が関連付けられません。

XML の名前空間の詳細については、World Wide Web Consortium(W3C) (<http://www.w3.org>)  にある XML の名前空間に関する W3C の勧告 (英語) を参照してください。

予約済みの QName 接頭部

xml, xs, xsi, fn, local は、予約済みの QName 接頭部であり、ユーザー定義の要素や属性を修飾するために使用しないでください。

検索での名前空間の使用

検索語の @xmlxp 照会プロローグで QName 接頭部と要素のデフォルトの名前空間を定義する必要があります。

名前空間 ns1 を URI "http://mycompany.com" に対応付けるプロローグの例

```
declare namespace ns1 = "http://mycompany.com";
```

すべての非修飾要素を URI "http://mycompany.com" で修飾することを指定するプロローグの例:

```
declare default element namespace "http://mycompany.com"
```

照会で名前空間の QName 接頭部や要素のデフォルトの名前空間を宣言しなければ、名前空間が照会の対象になることはありません。要素名または属性名は、どの名前空間に属する場合でも一致項目と見なされません。

QName 接頭部やデフォルトの名前空間を宣言した場合に、要素名または属性名が一致項目と見なされるのは、その要素名または属性名が指定の名前空間に属する場合に限られます。

次に、以下の構文を取り上げます。

```
declare default element namespace "";
```

この構文を使用すれば、非修飾タグがどの名前空間にも属さないことを指定できます。

| XML 検索ストリングで使用する QName 接頭部が、XML 文書で使用されている QName 接頭部と一致する必要はありません。一致しているかどうかの判断は、ロング・ネームの URI だけにに基づきます。

| 例:

| 検索の対象を test 要素の attr 属性に限定します。この場合は、test 要素を "http://posample.org" 名前空間に対応付けます。attr 属性は、どの名前空間にも属していません。デフォルトの名前空間を使用して、構文を簡略化します。

```
| CONTAINS(myxmlcol, '@xmlns:'declare default element namespace "http://myexample.org";  
| /test[@attr > xs:date("2005-01-01")]')
```

| 検索の対象を test 要素の attr 属性に限定します。この場合は、test 要素の名前空間を "http://myexample.org" にします。QName 接頭部 abc を使用して、明示的な名前空間構文を使用します。

```
| CONTAINS(myxmlcol, '@xmlns:'declare namespace abc = "http://myexample.org";  
| /abc:test[@attr < xs:date("2009-01-01")]')
```

| 検索の対象を purchaseOrder 要素の子要素である shipTo name と billTo name に限定します。この場合は、QName 接頭部 ns1 によって purchaseOrder 要素を "http://myexample.org" 名前空間に明示的に対応付けます。さらに、shipTo、name、billTo に適用するデフォルトの名前空間も定義します ("http://mastsample.org")。

```
| CONTAINS(myxmlcol, '@xmlns:' declare default namespace "http://mastsample.org";  
| declare namespace ns1 = "http://posample.org"; /ns1:purchaseOrder[shipTo/name = "Jane"  
| and billTo/name = "Jason"]')
```

| 検索の対象を shipTo 要素 (デフォルトの名前空間 "http://mastsample.org" に属する) の name 属性 ("http://posample.org" 名前空間で明示的に定義する) に限定します。その要素は、purchaseOrder 要素 ("http://posample.org" 名前空間で明示的に定義する) の子です。さらに、shipTo、billTo、name の各要素に適用するデフォルトの名前空間 "http://mastsample.org" も定義します。

```
| CONTAINS(myxmlcol, '@xmlns:' declare default namespace "http://mastsample.org";  
| declare namespace ns1 = "http://posample.org"; /ns1:purchaseOrder/shipTo[@ns1:name =  
| "Jane" and billTo/name = "Jason"]')
```

| XML 検索の例

| • XMLTEST というスキーマで、XML 文書を格納するための XML_DOCUMENTS という表を作成します。

```
| CREATE TABLE XMLTEST.XML_DOCUMENTS (ID INT, XML_DATA XML, PRIMARY KEY (ID));
```

| • XML 列に対応する XML_INDEX というテキスト検索索引を作成します。

```
| call SYSPROC.SYSTS_CREATE('XMLTEST', 'XML_INDEX', 'XMLTEST.XML_DOCUMENTS(XML_DATA)', '');
```

| • いくつかの XML 文書を挿入します。

```
| INSERT INTO XMLTEST.XML_DOCUMENTS (ID, XML_DATA)  
| VALUES(1,  
| '<<BOOK publication_date="2009-01-01">' ||  
| '<<TITLE> OmniFind_Text Search Server for DB2 </TITLE>' ||  
| '<<ID_NUMBER> 1 </ID_NUMBER>' ||  
| '<<CHAPTER>' ||  
| '<<NUMBER> 1 </NUMBER>' ||  
| '<<TITLE> Introduction </TITLE>' ||  
| '<<ABSTRACT> This chapter will introduce the reader to the capabilities of OmniFind  
| for DB2 for IBM i </ABSTRACT>' ||  
| '</CHAPTER>' ||  
| '<CHAPTER>' ||  
| '<NUMBER> 2 </NUMBER>' ||
```

```

|         ' <TITLE> Creating a Text Search Index </TITLE>' ||
|         ' <ABSTRACT> This chapter will explain how to create a text search index </ABSTRACT>' ||
|         ' </CHAPTER>' ||
|         '</BOOK>');
|
| INSERT INTO XMLTEST.XML_DOCUMENTS (ID, XML_DATA)
| VALUES(2,
|         '<BOOK publication_date="2010-02-01">' ||
|         ' <TITLE> Using the XML data type for DB2 for IBM i </TITLE>' ||
|         ' <ID_NUMBER> 2 ></ID_NUMBER>' ||
|         ' <CHAPTER>' ||
|         ' <NUMBER> 1 </NUMBER>' ||
|         ' <TITLE> Introduction </TITLE>' ||
|         ' <ABSTRACT> This chapter will introduce the reader to the DB2 XML data type </ABSTRACT>' ||
|         ' </CHAPTER>' ||
|         ' <CHAPTER>' ||
|         ' <NUMBER> 2 </NUMBER>' ||
|         ' <TITLE> Inserting XML data into a DB2 table </TITLE>' ||
|         ' <ABSTRACT> This chapter will explain how to insert XML data into a DB2 table </ABSTRACT>' ||
|         ' </CHAPTER>' ||
|         ' <CHAPTER>' ||
|         ' <NUMBER> 3 </NUMBER>' ||
|         ' <TITLE> Searching XML data </TITLE>' ||
|         ' <ABSTRACT> This chapter will explain how to query data in XML columns
|         using the CONTAINS and SCORE built-in functions </ABSTRACT>' ||
|         ' </CHAPTER>' ||
|         '</BOOK>');
|
| INSERT INTO XMLTEST.XML_DOCUMENTS (ID, XML_DATA)
| VALUES(3,
|         '<BOOK xmlns="http://www.ibm.com/digital_media_library" ' ||
|         ' publication_date="2010-02-01">' ||
|         ' <TITLE> Using Namespaces with OmniFind Text Search Server for DB2 for IBM i </TITLE>' ||
|         ' <ID_NUMBER> 2 </ID_NUMBER>' ||
|         ' <CHAPTER>' ||
|         ' <NUMBER> 1 </NUMBER>' ||
|         ' <TITLE> Introduction </TITLE>' ||
|         ' <ABSTRACT> This chapter will introduce the reader to XML namespaces </ABSTRACT>' ||
|         ' </CHAPTER>' ||
|         ' <CHAPTER>' ||
|         ' <NUMBER> 2 </NUMBER>' ||
|         ' <TITLE> Using default namespaces </TITLE>' ||
|         ' <ABSTRACT> This chapter will explain how to use a namespace in an XML search </ABSTRACT>' ||
|         ' </CHAPTER>' ||
|         '</BOOK>');
|
| • 索引を更新します。
|
| CALL SYSPROC.SYSTS_UPDATE('XMLTEST', 'XML_INDEX', '');

```

照会の例

例 1:

DB2 を含む直接の子孫 TITLE を持ち、ルート要素が BOOK のすべての文書を検索します。

```

| SELECT ID
| FROM XMLTEST.XML_DOCUMENTS
| WHERE CONTAINS(XML_DATA, '@xmlns:''/BOOK/TITLE[. contains("DB2")]'' ) = 1;

```

検索語で名前空間プロローグを指定していないので、名前空間は検索の対象になりません。

表 15. 結果

ID
1

表 15. 結果 (続き)

ID
2
3

例 2:

DB2 を含む直接の子孫 TITLE を持ち、ルート要素が BOOK のすべての文書を検索します。デフォルトの名前空間要素を使用して、BOOK と TITLE が "http://www.ibm.com/digital_media_library" という名前空間に属していなければならない、という条件を指定します。

```
SELECT ID
FROM XMLTEST.XML_DOCUMENTS
WHERE CONTAINS(XML_DATA, '@xmlns:'declare default element namespace
"http://www.ibm.com/digital_media_library";
/BOOK/TITLE[. contains("DB2")]'' ') = 1;
```

表 16. 結果

ID
3

例 3:

ルート要素が BOOK で、その publication_date 属性の値が "2010-01-01" より後の日付になっていて、子要素 TITLE の値に DB2 が含まれているすべての文書を検索します。BOOK タグと TITLE タグがどの名前空間にも属してはならない、という条件で検索結果を限定します。

```
SELECT ID
FROM XMLTEST.XML_DOCUMENTS
WHERE CONTAINS(XML_DATA, '@xmlns:'declare default element namespace "";
/BOOK[@publication_date > xs:date("2010-01-01")]/TITLE[. contains("DB2")]'' ') = 1;
```

表 17. 結果

ID
2

例 4:

ルート要素が BOOK (どの名前空間にも属さない) で、直下の下位要素 CHAPTER (やはりどの名前空間にも属さない) に XML 表へのデータの挿入に関する情報が含まれているすべての文書を検索します。

```
SELECT ID
FROM XMLTEST.XML_DOCUMENTS
WHERE CONTAINS(XML_DATA, '@xmlns:'declare default element namespace "";
/BOOK/CHAPTER[. contains("inserting XML data into a table")]'' ') = 1;
```

注:

- CHAPTER のテキストには、CHAPTER の下位要素である ABSTRACT と TITLE のテキストも含まれます。
- この検索ストリングでは大/小文字の区別はありません。検索語の言語処理上のバリエーションも対象になります。

| 表 18. 結果

ID
2

| 例 5:

| ルート要素が BOOK ("http://www.ibm.com/digital_media_library" 名前空間に属する) で、直下に下位要素 CHAPTER (やはり "http://www.ibm.com/digital_media_library" 名前空間に属する) があるすべての文書を検索します。CHAPTER に XML 名前空間の検索に関するテキスト情報が含まれていて、直下の下位要素 NUMBER ("http://www.ibm.com/digital_media_library" 名前空間に属する) の値が 1 になっていることが必要です。

```
| SELECT ID  
| FROM XMLTEST.XML_DOCUMENTS  
| WHERE CONTAINS(XML_DATA, '@xmlns:'declare namespace ns1 = "http://www.ibm.com/digital_media_library";  
| /ns1:BOOK/ns1:CHAPTER[. contains("search XML using a namespace") and NUMBER = 1]'') = 1;
```

| 文書 #3 は、各タグが正しい名前空間に入っている唯一の文書ですが、キーワードの一致項目が数値 1 ではなく 2 の章にのみ含まれています。

| 返される行はありません。

| 表 19. 結果

ID

| 例 6:

| ルート要素が BOOK ("http://www.ibm.com/digital_media_library" 名前空間に属する) で、直下に下位要素 CHAPTER ("http://www.ibm.com/digital_media_library" 名前空間に属する) があるすべての文書を検索します。CHAPTER の直下の下位要素 NUMBER ("http://www.ibm.com/digital_media_library" 名前空間に属する) の値が 1 になっていることが必要です。さらに、BOOK の下位要素 CHAPTER (必ずしも下位要素として NUMBER が存在しなくてもよい) に XML 名前空間の検索に関するテキスト情報が含まれていることも必要です。

```
| SELECT ID  
| FROM XMLTEST.XML_DOCUMENTS  
| WHERE CONTAINS(XML_DATA, '@xmlns:'declare namespace ns1 = "http://www.ibm.com/digital_media_library";  
| /ns1:BOOK[ns1:CHAPTER contains("search XML using a namespace")]/ns1:CHAPTER[ns1:NUMBER = 1]'^') = 1;
```

| 文書 3 には、CONTAINS の基準に合致する CHAPTER 要素があり、さらに CHAPTER 要素の下位要素 NUMBER の値が 1 になっています。したがって、文書 3 がこの照会の一致項目になります。

| 表 20. 結果

ID
3

関連資料

21 ページの『SYSPROC.SYSTS_CREATE』

SYSPROC.SYSTS_CREATE ストアド・プロシージャを呼び出すことによって、テキスト検索索引を作成できます。このストアド・プロシージャでは、テキスト列に基づいてテキスト検索索引を作成します。そのテキスト検索索引は、CONTAINS 関数または SCORE 関数を組み込んだ SQL 照会で使用できます。

37 ページの『SYSPROC.SYSTS_UPDATE』

SYSPROC.SYSTS_UPDATE ストアド・プロシージャを呼び出すことによって、テキスト列の現在の内容に合わせてテキスト検索索引を更新できます。

40 ページの『CONTAINS』

CONTAINS 関数では、検索引数として指定する基準を使用して、テキスト検索索引を検索できます。この関数は、一致項目が見つかったかどうかを示す結果を返します。

XML 検索の例

この例には、XML 文書の表、その表の XML 列に対応したテキスト検索索引、CONTAINS を使用した 6 つの SQL テキスト検索が含まれています。

注: コーディング例を使用すると、115 ページの『コードに関するライセンス情報および特記事項』の条件に同意したものとみなされます。

- XMLTEST というスキーマで、XML 文書を格納するための XML_DOCUMENTS という表を作成します。

```
CREATE TABLE XMLTEST.XML_DOCUMENTS (ID INT, XML_DATA XML, PRIMARY KEY (ID));
```

- XML 列に対応する XML_INDEX というテキスト検索索引を作成します。

```
call SYSPROC.SYSTS_CREATE('XMLTEST', 'XML_INDEX', 'XMLTEST.XML_DOCUMENTS(XML_DATA)', '');
```

- いくつかの XML 文書を挿入します。

```
INSERT INTO XMLTEST.XML_DOCUMENTS (ID, XML_DATA)
VALUES(1,
'
```

```
INSERT INTO XMLTEST.XML_DOCUMENTS (ID, XML_DATA)
VALUES(2,
'
```

```

|         '</CHAPTER>' ||
|         '<CHAPTER>' ||
|         '<NUMBER> 3 </NUMBER>' ||
|         '<TITLE> Searching XML data </TITLE>' ||
|         '<ABSTRACT> This chapter will explain how to query data in XML columns
|         using the CONTAINS and SCORE built-in functions </ABSTRACT>' ||
|         '</CHAPTER>' ||
|         '</BOOK>');
|
| INSERT INTO XMLTEST.XML_DOCUMENTS (ID, XML_DATA)
| VALUES(3,
|         '<BOOK xmlns="http://www.ibm.com/digital_media_library" ||
|         publication_date="2010-02-01">' ||
|         '<TITLE> Using Namespaces with OmniFind Text Search Server for DB2 for IBM i </TITLE>' ||
|         '<ID_NUMBER> 2 </ID_NUMBER>' ||
|         '<CHAPTER>' ||
|         '<NUMBER> 1 </NUMBER>' ||
|         '<TITLE> Introduction </TITLE>' ||
|         '<ABSTRACT> This chapter will introduce the reader to XML namespaces </ABSTRACT>' ||
|         '</CHAPTER>' ||
|         '<CHAPTER>' ||
|         '<NUMBER> 2 </NUMBER>' ||
|         '<TITLE> Using default namespaces </TITLE>' ||
|         '<ABSTRACT> This chapter will explain how to use a namespace in an XML search </ABSTRACT>' ||
|         '</CHAPTER>' ||
|         '</BOOK>');

```

- 索引を更新します。

```

| CALL SYSPROC.SYSTS_UPDATE('XMLTEST', 'XML_INDEX', '');

```

照会の例

検索 1:

DB2 を含む直接の子孫 TITLE を持ち、ルート要素が BOOK のすべての文書を検索します。

```

| SELECT ID
| FROM XMLTEST.XML_DOCUMENTS
| WHERE CONTAINS(XML_DATA, '@xmlns:='/BOOK/TITLE[. contains("DB2")]'' ') = 1;

```

検索語で名前空間プロローグを指定していないので、名前空間は検索の対象になりません。

表 21. 結果

ID
1
2
3

検索 2:

DB2 を含む直接の子孫 TITLE を持ち、ルート要素が BOOK のすべての文書を検索します。デフォルトの名前空間要素を使用して、BOOK と TITLE が "http://www.ibm.com/digital_media_library" という名前空間に属していなければならない、という条件を指定します。

```

| SELECT ID
| FROM XMLTEST.XML_DOCUMENTS
| WHERE CONTAINS(XML_DATA, '@xmlns:='declare default element namespace
| "http://www.ibm.com/digital_media_library";
| /BOOK/TITLE[. contains("DB2")]'' ') = 1;

```

表 22. 結果

ID
3

検索 3:

ルート要素が BOOK で、その publication_date 属性の値が "2010-01-01" より後の日付になっていて、子要素 TITLE の値に DB2 が含まれているすべての文書を検索します。BOOK タグと TITLE タグがどの名前空間にも属してはならない、という条件で検索結果を限定します。

```
SELECT ID
FROM XMLTEST.XML_DOCUMENTS
WHERE CONTAINS(XML_DATA, '@xmlns:declare default element namespace "";
/BOOK[@publication_date > xs:date("2010-01-01")]/TITLE[. contains("DB2")]'' ') = 1;
```

表 23. 結果

ID
2

検索 4:

ルート要素が BOOK (どの名前空間にも属さない) で、直下の下位要素 CHAPTER (やはりどの名前空間にも属さない) に XML 表へのデータの挿入に関する情報が含まれているすべての文書を検索します。

```
SELECT ID
FROM XMLTEST.XML_DOCUMENTS
WHERE CONTAINS(XML_DATA, '@xmlns:declare default element namespace "";
/BOOK/CHAPTER[. contains("inserting XML data into a table")]'' ') = 1;
```

注:

- CHAPTER のテキストには、CHAPTER の下位要素である ABSTRACT と TITLE のテキストも含まれます。
- この検索ストリングでは大/小文字の区別はありません。検索語の言語処理上のバリエーションも対象になります。

表 24. 結果

ID
2

検索 5:

ルート要素が BOOK ("http://www.ibm.com/digital_media_library" 名前空間に属する) で、直下に下位要素 CHAPTER (やはり "http://www.ibm.com/digital_media_library" 名前空間に属する) があるすべての文書を検索します。CHAPTER に XML 名前空間の検索に関するテキスト情報が含まれていて、直下の下位要素 NUMBER ("http://www.ibm.com/digital_media_library" 名前空間に属する) の値が 1 になっていることが必要です。

```
SELECT ID
FROM XMLTEST.XML_DOCUMENTS
WHERE CONTAINS(XML_DATA, '@xmlns:declare namespace ns1 = "http://www.ibm.com/digital_media_library";
/ns1:BOOK/ns1:CHAPTER[. contains("search XML using a namespace") and NUMBER = 1]'' ') = 1;
```

| 文書 #3 は、各タグが正しい名前空間に入っている唯一の文書ですが、キーワードの一致項目が数値 1 で
| はなく 2 の章にのみ含まれています。

| 返される行はありません。

| 表 25. 結果

ID

| 検索 6:

| ルート要素が BOOK ("http://www.ibm.com/digital_media_library" 名前空間に属する) で、直下に下位要
| 素 CHAPTER ("http://www.ibm.com/digital_media_library" 名前空間に属する) があるすべての文書を検
| 索します。CHAPTER の直下の下位要素 NUMBER ("http://www.ibm.com/digital_media_library" 名前空間に
| 属する) の値が 1 になっていることが必要です。さらに、BOOK の下位要素 CHAPTER (必ずしも下位要素と
| して NUMBER が存在しなくてもよい) に XML 名前空間の検索に関するテキスト情報が含まれていること
| も必要です。

```
| SELECT ID  
| FROM XMLTEST.XML_DOCUMENTS  
| WHERE CONTAINS(XML_DATA, '@xmlns:'declare namespace ns1 = "http://www.ibm.com/digital_media_library";  
| /ns1:BOOK[ns1:CHAPTER contains("search XML using a namespace")]/ns1:CHAPTER[ns1:NUMBER = 1]'') = 1;
```

| 文書 3 には、CONTAINS の基準に合致する CHAPTER 要素があり、さらに CHAPTER 要素の下位要素
| NUMBER の値が 1 になっています。したがって、文書 3 がこの照会の一致項目になります。

| 表 26. 結果

ID
3

| XML 検索照会の構文

| XML 検索の構文は、Extended Backus-Naur Form (EBNF) 構文で定義されている XPath 言語のサブセット
| に基づいています。サポートされている構文に準拠していない照会は、照会パーサーによって拒否されま
| す。

| EBNF 構文は、以下の方法で簡略化されています。

- | • 述部式では絶対パス名を使用できなくなっています。
- | • 1 つの軸 (タグ) だけを認識し、順方向に進むだけになっています。
- | • ワイルドカード文字の使用に追加の意味制限が適用されています (50 ページの『XML 検索』の『パス
| 式のワイルドカード・サポート』のセクションを参照してください)。
- | • 名前空間を明示的または暗黙的に使用する前に、検索ストリングで名前空間の宣言を指定することが必
| 要になっています。名前空間の宣言を組み込まないと、名前空間が検索の対象になりません。
- | • 相対パス式にタグ名または属性名を組み込むことが必要になっています。ルート・ノードを選択するた
| めの「/」という照会や、すべてのノードを選択するための「//」という照会は、無効な式です。

| EBNF 記法でサポートされている構文を以下の表にまとめます。

| 表 27. EBNF 記法でサポートされている照会構文

記号	実動
----	----

表 27. EBNF 記法でサポートされている照会構文 (続き)

XMLQuery ::=	QueryPrefix NameSpaceDeclaration QueryString QueryPrefix QueryString
QueryPrefix ::=	@xmlns:
QueryString ::=	"" PathExpr ""
PathExpr ::=	RelativePathExpr "/" RelativePathExpr? "//" RelativePathExpr
RelativePathExpr ::=	StepExpr (("/" "//") StepExpr)*
StepExpr ::=	("." AbbrevForwardStep) Predicate?
AbbrevForwardStep ::=	"@"? (QName "*")
Predicate ::=	"[" PredicateExpr "]"
PredicateExpr ::=	Expr PredicateExpr ("and" "or") "(" PredicateExpr ")"
Expr ::=	ComparisonExpr ContainmentExpr
ComparisonExpr ::=	PathExpr ComparisonOp Literal
ComparisonOp ::=	"=" "<" ">" "!=" "<=" ">="
Literal ::=	StringLiteral NumericLiteral DateLiteral
ContainmentExpr ::=	PathExpr "contains" "(" StringLiteral)" PathExpr "excludes" "(" StringLiteral)"
StringLiteral ::=	"%"* ["^"]* "%" "" ["^"]* ""
DateLiteral ::=	"xs:date(% " xmlDate "%)" "xs:dateTime(% " xmlDateTime "%)"
xmlDate ::=	yyyy"-mm"-dd
xmlDateTime ::=	yyyy"-mm"-dd [T] hh":"mm":"ss"."uuuuuu
NameSpaceDeclaration ::=	defaultNameSpace (NameSpacePrefixDeclaration)*
defaultNameSpace ::=	"declare default element namespace " StringLiteral ";"
NameSpacePrefixDeclaration ::=	"declare namespace" NameSpacePrefix "=" StringLiteral ";"
NameSpacePrefix ::=	[^:]+

DB2(r) for i OmniFind(r) テキスト検索サーバーの管理

以下の技法やツールを使用して、DB2 for i OmniFind テキスト検索サーバーを管理できます。

DB2 for i OmniFind テキスト検索サーバーの開始

SYSPROC.SYSTS_START を呼び出すことによって、DB2 for i OmniFind テキスト検索サーバーを開始できます。

| OMNIFINDは、QSYS2.SYSTEXTSERVERS の SERVERSTATUS が 0 になっている限り、必要なときにテキスト検索サーバーを自動的に開始します。このポリシーによって、ホスト・システムの始動時にテキスト検索サーバーを自動的に開始することが可能になります。ただし、必要に応じてサーバーを手動で開始することもできます。

| サーバーを開始するには、以下のようにします。

| CALL SYSPROC.SYSTS_START(serverid)

| このプロシージャを呼び出して、処理が成功すると、QSYS2.SYSTEXTSERVERS の SERVERSTATUS が 0 に設定されます。サーバーがローカルであれば、以下のジョブがバックグラウンドでアクティブになります。

- | • QJVAEXEC userx BCI 0.0 JVM-com.ibm.es
- | • QJVAEXEC userx BCI 0.0 PGM-OutsideInP

| userx は、このストアード・プロシージャを呼び出した管理者のユーザー ID です。

| このすべてのジョブがアクティブになって、テキスト・サーバーを使用できるようになるまでには、数分の時間がかかる場合もあります。

| 関連資料

| 17 ページの『SYSPROC.SYSTS_START』

| SYSPROC.SYSTS_START ストアード・プロシージャを呼び出すことによって、DB2 のテキスト検索機能を有効にできます。

| 110 ページの『QSYS2.SYSTEXTSERVERS 管理表』

| QSYS2.SYSTEXTSERVERS 管理表では、テキスト検索サーバーのインストール先に関する情報を確認できます。

| DB2 for i OmniFind テキスト検索サーバーの停止

| 用意されているシャットダウン・スクリプトを使用して、DB2 for i OmniFind テキスト検索サーバーを手動で停止できます。

| DB2 for i OmniFind テキスト検索サーバーをサービスとしてインストールした場合は、ホスト・システムのシャットダウン時にテキスト検索サーバーも自動的に停止します。ただし、DB2 for i OmniFind テキスト検索サーバーをサービスとしてインストールした場合でも、サーバーを手動で停止することは可能です。

| DB2 for i OmniFind テキスト検索サーバーを停止するには、以下のようにします。

| 1. SYSPROC.SYSTS_STOP を呼び出して、サーバーを停止することを SYSTEXTSERVER カタログで示します。

- | • すべてのサーバーを停止する場合は、CALL SYSPROC.SYSTS_STOP() を使用します。
- | • 特定のサーバーを停止する場合は、以下のようにします。
 - | a. サーバー・カタログの照会によって、停止するサーバーの serverid を取得します。

```
| SELECT SERVERID,SERVERPORT,SERVERSTATUS,SERVERPATH
|
| FROM QSYS2.SYSTEXTSERVERS
```

注: SERVERPATH では、サーバー・パスを確認できます。 SERVERSTATUS では、サーバーが現時点でアクティブ (0) なのか非アクティブ (1) なのかを確認できます。

- b. 停止するサーバーを数値の *serverid* または別名で指定して、SYSPROC.SYSTS_STOP を呼び出します。

```
CALL SYSPROC.SYSTS_STOP(serverid).
```

2. (オプション) シャットダウン・スクリプトを呼び出して、サーバー自体を停止します。サーバーを停止すると、ホスト・システムのすべてのテキスト検索サーバー・ジョブが終了します。 Qshell 環境でサーバーを停止します。

ローカル・サーバーをシャットダウンする場合は、コマンド行で以下のコマンドを入力します。

```
QSH CMD('cd /QOpenSys/QIBM/ProdData/TextSearch/server1/bin; shutdown.sh')
```

シャットダウンするサーバーが、インストール・プロセスで作成したデフォルトのローカル・サーバーでない場合は、QSYS2.SYSTEXTSERVERS から正しい SERVERPATH 値を取得する必要があります。その SERVERPATH を /QOpenSys/QIBM/ProdData/TextSearch/server1/bin の代わりに使用してください。

シャットダウン・スクリプトを使用してサーバーを停止した場合は、SERVERSTATUS カタログ値が非アクティブ (1) 状況に変更されません。次回、SYSTS_CREATE、SYSTS_UPDATE、SYSTS_DROP の各ストアード・プロシージャを呼び出したときや、CONTAINS または SCORE の各組み込み関数を SQL 照会の一部として呼び出したときに、サーバーは自動的に開始します。

関連資料

19 ページの『SYSPROC.SYSTS_STOP』

SYSPROC.SYSTS_STOP ストアード・プロシージャを呼び出すことによって、DB2 のテキスト検索機能を停止できます。このストアード・プロシージャを実行すると、QSYS2.SYSTEXTSERVERS カタログの SERVERSTATUS 値が 1 (停止) に設定されます。

110 ページの『QSYS2.SYSTEXTSERVERS 管理表』

QSYS2.SYSTEXTSERVERS 管理表では、テキスト検索サーバーのインストール先に関する情報を確認できます。

テキスト検索索引の保管と復元

テキスト検索索引を保管して復元できます (データを組み込むことも組み込まないことも可能です)。

テキスト検索索引をデータなしで保管して復元する操作

テキスト検索索引の構造を索引データなしで保管して復元することもできます。保管と復元のプロセスを実行するには、CL コマンドの SAVOBJ と SAVLIB を使用できます。

SYSTS_CREATE を使用してテキスト検索索引を作成すると、索引のスキーマに基づいて DB2 ビューが作成されます (索引の名前がビュー名になります)。そのビューは、索引の構造を保管して復元するメカニズムとしての役割を果たします。

ユーザーは、データベース表とデータベース・ビューを保管する場合と同じ方法でそのビューを保管できます (CL コマンド SAVOBJ または SAVLIB を参照してください)。そのビューを保管すると、復元時に索引を再作成するために必要な追加情報も自動的に保管されます。

そのビューを復元するには、CL コマンド RSTOBJ または RSTLIB を使用します。DB2 for i は、そのビューがテキスト検索索引に対応していることを認識して、索引を再作成します。索引の構造の再作成が完了した後に、更新要求がバックグラウンド・ジョブに送信され、索引のデータが再び取り込まれます。

復元プロセスに関する追加の注意点を以下にまとめます。

1. テキスト検索サーバーを始動できない場合や、必要な製品がシステムにインストールされていない場合は、復元処理が失敗します。必要な製品のリストについては、ソフトウェア要件を参照してください。
2. テキスト検索索引がシステムに存在する場合は、以下のアクションが発生します。
 - a. 既存の索引情報が復元対象の索引と完全に一致する場合は、復元が成功します。索引の再作成は発生しません。
 - b. 既存の索引情報が復元対象の索引と一致しないときに、索引の再作成なしで一致のための変更ができない場合は、復元が失敗します。
 - c. 既存の索引情報が復元対象の索引と一致しないときに、SYSTS_ALTER によって一致のための変更ができる場合は、保管済みの索引に合わせて既存の索引が変更されます。索引の再作成は発生しません。
3. 索引を復元すると、その索引は、保管時に使用されていたのと同じテキスト検索サーバーを使用するようになります。保管時に使用されていたサーバーが定義されていない場合は、現時点で使用可能なサーバーが選択されます。保管時のサーバーが定義されていても、使用可能な状態になっていない場合は、復元が失敗します。
4. 基本表の列が非互換であるなど、他の何らかの理由でテキスト検索索引を作成できない場合は、復元が失敗します。
5. QSYS2 に含まれているステージング表の名前、基本表に追加されたトリガーの名前、テキスト検索サーバーにあるコレクションの名前は、変わる可能性があります。それらのオブジェクトに関して付与されている特殊権限は、保存されません。
6. テキスト検索索引の同義語ディクショナリーに追加された同義語は、保存されません。
7. 復元時に索引がシステム・カタログに存在している場合に、ビューが現時点でシステムに存在しなければ、ビューだけが復元されます。ステージング表、テキスト検索サーバーのコレクション、基本表のトリガーは、作成されません。

その場合は、テキスト検索索引が大きな復元処理の一部と見なされます。つまり、索引の個々の部分がユーザーによって明示的に保管され、そのすべてが復元される、という処理 (システム全体の復元など) の一部と見なされる、ということです。

索引が正しく機能するためには、索引のすべての必要な部分を復元しなければなりません。索引のすべての部分を同期した状態にしておくのは、ユーザーの責任です。

テキスト検索索引では、据え置きオブジェクト復元 (RSTDFROBJ) コマンドを使用できます。CL コマンド RSTOBJ と RSTLIB では、DFRID パラメーターを使用することをお勧めします。そのパラメーターを使用すれば、索引の作成が失敗する原因になっていた一般的な状態を修正した後に、RSTDFROBJ コマンドによってテキスト検索索引を復元できるようになります。

索引の据え置きの原因になる状態の例を以下にまとめます。

- 必要な製品がインストールされていませんでした。
- テキスト検索サーバーが使用可能な状態になっていないか、定義されていませんでした。
- 基本表が存在しませんでした。

関連資料

21 ページの『SYSPROC.SYSTS_CREATE』

SYSPROC.SYSTS_CREATE ストアド・プロシージャを呼び出すことによって、テキスト検索索引を作成できます。このストアド・プロシージャでは、テキスト列に基づいてテキスト検索索引を作成します。そのテキスト検索索引は、CONTAINS 関数または SCORE 関数を組み込んだ SQL 照会で使用できます。

30 ページの『SYSPROC.SYSTS_ALTER』

SYSPROC.SYSTS_ALTER ストアド・プロシージャを呼び出すことによって、SYSPROC.SYSTS_CREATE で作成した索引の属性を変更できます。変更できるのは、このプロシージャで明示的に指定した属性に限られます。索引の他の属性はすべて変更されません。

関連情報

オブジェクト保管 (SAVOBJ)

ライブラリー保管 (SAVLIB)

オブジェクト復元 (RSTOBJ)

ライブラリー復元 (RSTLIB)

据え置きオブジェクト復元 (RSTDFROBJ)

テキスト検索索引をデータと一緒に保管して復元する操作

テキスト検索索引をデータと一緒に保管して復元する操作は、データなしの操作よりも完成度の高い操作です。

以下のオブジェクトを保管する必要があります。

- テキスト検索索引 (統合ファイル・システムに格納します)。
- 基本表 (索引生成の基になる表) のレコード変更を追跡管理するためのログ・ファイルとして使用するステージング表。ステージング表は、QSYS2 ライブラリーにあります。名前の先頭は QDBTS になっています (QDBTS00001 など)。
- ビュー (テキスト索引に相当するデータベース・オブジェクト)。ビューの名前は、テキスト索引の名前と同じです。
- 索引生成の基になる基本表。
- 索引を追跡管理するための情報を格納する SQL カタログ。

以下の手順を実行して、テキスト検索索引を保管します。

1. 推奨事項: テキスト検索索引の更新操作 (SYSTS_UPDATE) を最初に実行して、索引を最新の状態にしてください。
2. 標準的な保管手法 (SAVOBJ コマンドなど) を使用して、基本表とビューを保管します。
3. 標準的な保管手法を使用して、QSYS2 に含まれているステージング表を保管します。例えば、SAVOBJ LIB(QSYS2) OBJ(QDBTS*) のように記述します。
4. QSYS2 に含まれているテキスト検索索引のカタログを保管します。

すべてのカタログ名の先頭は SYSTXT になっています (例えば、SYSTXTSRVR)。QSYS2 に含まれている他の SQL カタログの場合と同じく、自分の責任でバックアップ・コピーを保管して使用可能な状態にしておく必要があります。

そのバックアップ・コピーは、以下の 2 つの方法のいずれかで作成できます。

- | a. SAVLIB コマンドで LIB パラメーターを *ALLUSR または *IBM に指定して、ライブラリー全体を保管します。
- | b. SAVOBJ コマンドで LIB(QSYS2) と OBJ((SYSTXT*)) を指定して、特定のテキスト検索カタログを保管します。
- | 5. テキスト検索索引の情報を統合ファイル・システムに保管します。その情報には、テキスト・サーバー・パスの下にある *config* ディレクトリーの内容全体が含まれます。

| テキスト・サーバー・パスは、対象サーバーの SYSTXTSRVR カタログの SERVERPATH 列に対する照会で確認できます。サーバー・パスには bin ディレクトリーが追加されているので、そのディレクトリーを config ディレクトリーに置き換えてください。

| 一般的な保管手法は SAV コマンドですが、どんなタイプの保管圧縮でも利用できます。

| 注: この保管情報が該当するのは、IBM i で実行するテキスト・サーバーに限られます。

| 例:

| OMNIFINDによって作成したデフォルトのテキスト・サーバーに関連したすべてのテキスト索引を保管する
| とします。 QGPL/MYDOCIX というテキスト索引の生成の基になっているのは、QGPL/MYDOCS という
| 表です。この例で使用する保管メディアは、保管ファイルです。

| 以下の手順を実行します。

- | 1. QSYS2 に含まれているすべてのステージング表と OMNIFINDカタログを保管します。

```
| SAVOBJ OBJ(QDBTS* SYSTXT*) LIB(QSYS2) DEV(*SAVF) SAVF(QGPL/SAVFQSYS2)
```

- | 2. 基本表とビューを保管します。

```
| SAVOBJ OBJ(MYDOCS MYDOCIX) LIB(QGPL) DEV(*SAVF) SAVF(QGPL/SAVFMYFILE)
```

- | 3. SQL を使用して、テキスト・サーバーのパス名を取得します。この例では、serverid = 2 です。

```
| SELECT SERVERPATH FROM systxtsrvr WHERE serverid=2
```

| 返される SERVERPATH 値は、/QOpenSys/QIBM/ProdData/TextSearch/server1/bin/ です。

| 注: 照会の対象が正しいサーバーであることを確認してください。

- | 4. bin/ を config に置き換えて、テキスト索引を保管します。

```
| SAV DEV('/QSYS.LIB/QGPL.LIB/SAVIFS.FILE') OBJ((' /QOpenSys/QIBM/ProdData/TextSearch/server1/  
| config'))
```

| テキスト索引が保管ファイル QGPL/SAVFMYFILE、QGPL/SAVFQSYS2、QGPL/SAVIFS に保管されまし
| た。

| テキスト索引の復元は、保管と同じ順序で実行しなければなりません。QSYS2 のカタログを最初に復元す
| る必要があります。

| 関連資料

- | 37 ページの『SYSPROC.SYSTS_UPDATE』
- | SYSPROC.SYSTS_UPDATE ストアード・プロシージャーを呼び出すことによって、テキスト列の現在の内容に合わせてテキスト検索索引を更新できます。

| 関連情報

- | オブジェクト保管 (SAVOBJ)
- | ライブラリー保管 (SAVLIB)

| 問題判別

- | ログに記録されたシステム・メッセージとトレース・メッセージを使用して、発生した可能性がある問題の原因を確認できます。

- | DB2 for i OmniFind テキスト検索サーバーのサーバー・ログは、<INSTALL_HOME>/log ディレクトリーにあります。インストール時に作成されるデフォルトのサーバー・ログは、/QOpenSys/QIBM/ProdData/TextSearch/server1/log ディレクトリーにあります。

- | デフォルトでは、トレース・ログはオフになり、システム・ログのレベルは `informational` に設定されます。構成ツールを使用すれば、トレースとログ・レベルのオプションを変更できます。

- | サーバー・ログは、サイズに基づいて循環します。8 MB 以下のサーバー・ログの 5 つの最新コピーが格納されます。サーバー・ログを表示/保管するには、用意されているスクリプトを使用できます。

- | IBM i サーバーまたは Linux サーバーでは、スクリプトは `logformatter.sh` です。Windows サーバーでは、スクリプトは `logformatter.bat` です。

| オプション

- | このスクリプトには以下のオプションがあります。

| `-f logfile`

- | 書式を設定するサーバー・ログ・ファイルを指定します。

| `-l locale`

- | 書式を再設定したメッセージを書き込むときに使用するロケールを指定します。例えば、英語であれば `en_US` を指定し、日本語であれば `ja_JP` を指定します。この値はオプションです。デフォルト値は `en_US` です。

| `-o outputfile`

- | 書式を再設定したログ・メッセージを UTF-8 エンコード方式で書き込むための出力ファイルを指定します。この値はオプションです。このオプションを指定しない場合は、標準出力ファイルが使用されます。

- | `-?` ヘルプ・メッセージを出力します。この値はオプションです。

- | `-v` デバッグ・メッセージを表示するモードを指定します。この値はオプションです。

| サーバー・ログの表示と保管

- | 用意されているスクリプトを使用して、サーバー・ログを表示/保管できます。このログは、問題の原因を確認するために役立ちます。

- | サーバー・ログを表示/保管するには、以下のようにします。

1. サーバー・ログを表示するには、以下のいずれかのコマンドを実行します。

オプション	説明
IBM i (QSH 環境)	<code>bin/logformatter.sh -f log/System.0.log</code>
Linux サーバー	<code>bin/logformatter.sh -f log/System.0.log</code>
Windows サーバー	<code>bin/logformatter.bat -f log/System.0.log</code>

2. サーバー・ログをファイルに保管し、ファイル・エディターでそのログを読み取れるようにするには、以下のいずれかのコマンドを実行します。

オプション	説明
IBM i (QSH 環境)	<code>bin/logformatter.sh -f log/System.0.log -o <output filename></code>
Linux サーバー	<code>bin/logformatter.sh -f log/System.0.log -o <output filename></code>
Windows サーバー	<code>bin/logformatter.bat -f log/System.0.log -o <output filename></code>

管理ツール

- OMNIFINDには、一般的なタスクを実行するために使用できる各種のツールが用意されています。例えば、追加のテキスト検索サーバーの構成と管理や、コレクションへの同義語ディクショナリーの追加などのタスクがあります。

- これらのツールは、CL コマンドではなくシェル・スクリプトです。QSH 開始 (STRQSH) CL コマンドまたは QSH (QSH) CL コマンドで開始するスクリプト環境で、それらのスクリプトを呼び出せます。

- これらのツールは、ユーザー ID の認証を行いません。ただし、これらのツールを実行できるのは、テキスト検索サーバーに対する有効なアクセス権のあるユーザーに限られます。

関連情報

- QSH 開始 (STRQSH)
- QSH 開始 (QSH)

構成ツール

- DB2 for i OmniFind テキスト検索サーバーをインストールしたら、構成ツールを使用して構成設定をカスタマイズできます。

- ほとんどの構成設定については、構成ツールでカスタマイズする前に、テキスト検索サーバーを停止する必要があります。

- ただし、サーバーの実行中でも、以下のオプションを表示することは可能です。

- 現在の認証トークン
- サーバーのポート
- 現在のシステム・プロパティ

configServerAndDB2 ツール

configServerAndDB2 (configServerAndDB2.sh) ツールは、/QOpenSys/QIBM/ProdData/TextSearch 統合ファイル・システム・ディレクトリにあります。このツールを使用して、DB2 カタログ・ファイル SYSTEXTSERVERS の項目を作成/変更できます。

特定のサーバーに関連した認証トークンやポート番号を構成するために使用することも可能です。このツールを実行すれば、DB2 カタログ・ファイル SYSTEXTSERVERS の SERVERAUTHTOKEN と SERVERPORT の値を変更/設定できます。

ローカル・システムで実行する追加のサーバーを作成する場合は、代わりに 85 ページの『ServerInstance ツール』を使用します。

configServerAndDB2 (configServerAndDB2.sh) ツールを呼び出すときには、5 つのパラメーターを指定します。

1. 第 1 のパラメーターは、**generateToken** または **configureHTTPListener** です。
2. 第 2 のパラメーターは、**-serverPath** です。
3. 第 3 のパラメーターは、サーバー関連の情報を格納する統合ファイル・システムのルート・ノードのパスです。例えば、/QOpenSys/QIBM/ProdData/TextSearch/server2 のように記述します。
4. 第 4 と第 5 のパラメーターは、第 1 のパラメーターの値によって異なります。
 - 第 1 のパラメーターが **generateToken** であれば、第 4 のパラメーターは **-seed** であり、その後に第 5 のパラメーターとして整数 (1 など) を指定します。
 - 第 1 のパラメーターが **configureHTTPListener** であれば、第 4 のパラメーターは **-adminHTTPPort** です。第 5 のパラメーターは、サーバーのソケット・ポートとして使用する整数値です。

以下に 2 つの例を示します。

```
• STRQSH
  cd /QOpenSys/QIBM/ProdData/TextSearch
  configServerAndDB2.sh generateToken -serverPath /QOpenSys/QIBM/ProdData/TextSearch/server2 -seed 1
• STRQSH
  cd /QOpenSys/QIBM/ProdData/TextSearch
  configServerAndDB2.sh configureHTTPListener -serverPath /QOpenSys/QIBM/ProdData/TextSearch/server2
  -adminHTTPPort 9997
```

configTool スクリプト

configTool.sh スクリプトは、各ローカル・サーバーで使用できます。サーバー項目を変更するときを使用することは、お勧めできません。サーバー情報を出力するために使用することも可能です (printAll や printToken など)。

表 28. 構成ツールを実行するためのコマンド

IBM i:
configTool.sh <mandatory_command_option> <mandatory_global_arguments> <optional_global_arguments> <optional_command_options>
Linux サーバー

表 28. 構成ツールを実行するためのコマンド (続き)

<pre>configTool.sh <mandatory_command_option> <mandatory_global_arguments> <optional_global_arguments> <optional_command_options></pre>
<p>Windows サーバー</p> <pre>configTool.bat <mandatory_command_option> <mandatory_global_arguments> <optional_global_arguments> <optional_command_options></pre>

コマンド・オプション

構成ツールでは、以下のコマンド・オプションがサポートされています。

configureParams

構成できるシステム・パラメーターを指定します。以下のパラメーターを構成できます。

-configPath

config.xml ファイルが含まれている構成フォルダーの絶対パスを指定します。

-adminHTTPPort

管理 HTTP ポート番号を指定します。エラーが発生した場合は、エラー・コード -3 が返されます。

-logPath

ログ・ディレクトリーの絶対パスを指定します。

-tempDirPath

一時ディレクトリーの絶対パスを指定します。

-numberOfIndexers

同時に使用するテキスト検索索引サブシステムの数を指定します。

-numberOfTokenizers

入力を解析してトークンを作成するために使用する同時実行サブシステムの数を指定します。

-maxDocumentSize

文書の索引生成の対象にする文字の最大数を指定します。エラーが発生した場合は、エラー・コード -3 が返されます。

-logLevel

ログ・ファイル内のシステム・メッセージのログ・レベルを指定します。デフォルトのレベルは、`informational` です。追加のオプションとして、`warning` と `severe` があります。

-maxHeapSize

Java 仮想マシンで受け入れられる形式でヒープ・サイズを開始/終了します。エラーが発生した場合は、エラー・コード -5 が返されます。

printToken

現在の認証トークンと暗号鍵を出力します。

printAll

このツールで構成できるオプションのすべての現行値を出力します。

| *printAdminHTTPPort*
| 管理 HTTP ポートの現行値を出力します。

| *generateToken*
| 認証トークンを生成します。

| **グローバル引数**

| *-configPath*
| config.xml ファイルが含まれている構成フォルダーの絶対パスを指定します。このグローバル引数は必須です。

| *-locale*
| トレース・ファイルにメッセージを書き込むための 5 文字のロケール設定を指定します。この設定を指定しない場合は、デフォルト値として *-en_US* が使用されます。

| **例**

| Linux サーバーで、現在の認証トークンを出力するには、以下のコマンドを使用します。

| `configTool.sh printToken -configPath <path> <optional_global_arguments>`

| **関連情報**

| QSH 開始 (QSH)

| **SYSPROC.SYSTS_REMOVE**

| SYSPROC.SYSTS_REMOVE SQL ストアード・プロシージャを使用して、孤立した索引を削除できます。

| **権限**

| 孤立している可能性がある索引のコレクション名 (collection-name) を確認するには、QDBTS_LISTINXSTS ユーザー定義表関数 (UDTF) を使用できます。

| このステートメントの許可 ID には、少なくとも以下のいずれかの特権が必要です。

- | • *JOBCTL 権限
- | • QIBM_DB_SQLADM セキュリティー特殊関数使用権限

| **構文**

| `>>-SYSPROC.SYSTS_REMOVE (collection-name) -><`

| **パラメーター**

| *collection-name*
| 削除するコレクションの名前を識別するストリング・リテラルを指定します。

| **注:** このプロシージャでは、`adminTool.sh` シェル・スクリプトを使用して、コレクションのディレクトリーを削除します。そのシェル・スクリプトを使用するには、サーバーが稼働中の状態になっている必要があります。サーバーを始動していない状態でこのプロシージャを実行すると、エラー・メッセージが返されます。

| SYSTS_REMOVE の SQL

```
| CREATE PROCEDURE SYSPROC.SYSTS_REMOVE(  
|     IN COLLECTIONNAME VARCHAR(255) CCSID 1208)  
|     EXTERNAL NAME QDBTSLIB.DSN5RMCOLL  
|     DYNAMIC RESULT SETS 0  
|     LANGUAGE C++  
|     PARAMETER STYLE SQL  
|     PROGRAM TYPE MAIN  
|     COMMIT ON RETURN NO  
|     INHERIT SPECIAL REGISTERS;
```

| 例

| • 0_65_2815_2008_06_02_11_58_22_901726 というコレクション名の孤立した索引を ASP グループ *SYSBASE から削除する場合は、SQL インターフェースから以下のコマンドを入力します。

```
| CALL SYSPROC.SYSTS_REMOVE('0_65_2815_2008_06_02_11_58_22_901726')
```

| SYSTS_REMOVE ストアド・プロシージャは、その索引の情報が QSYS2.SYSTEXTINDEXES カタログ表に存在するかどうかを確認します。存在する場合 (true の場合) は、エラー・メッセージ DSX_INDEX_EXIST が返されます。存在しなければ、このプロシージャは、サーバー 65 の config/collections ディレクトリーの下を検索します。

| コレクションが存在しなければ、エラー・メッセージ DSX_COLLECTION_NOT_FOUND が返されます。コレクションが存在すれば、このプロシージャは、adminTool.sh を呼び出してそのコレクションを削除します。

| その後、このプロシージャは、対象のディレクトリーを再びチェックして、コレクションが削除されたかどうかを確認します。コレクションが削除されていない場合は、エラー・メッセージ DSX_REMOVE_COLLECTION_FAILED がユーザーに返されます。

| 注: テキスト検索サーバーにあるコレクションが独立 ASP グループに存在している場合は、SYSTS_REMOVE ストアド・プロシージャを呼び出すスレッドを独立 ASP の名前空間で実行する必要があります。ASP グループ設定 (SETASPGRP) コマンドを使用してください。

| • 33_7_26_2008_06_18_21_28_39_407824 というコレクション名の孤立した索引を独立 ASP iaspXXX から削除する場合は、以下のコマンドを使用できます。

| CL:

```
| SETASPGRP(isapXXX)
```

| SQL:

```
| CALL SYSPROC.SYSTS_REMOVE(' 33_7_26_2008_06_18_21_28_39_407824')
```

| 注: System i® ナビゲーターを使用する場合は、独立 ASP のデータベース名を右クリックして、SQL スクリプトを実行します。

| 関連資料

- | 107 ページの『QSYS2.SYSTEXTINDEXES 管理表』
- | QSYS2.SYSTEXTINDEXES 管理表では、各テキスト検索索引の情報を確認できます。それぞれのテキスト検索索引ごとに、名前、スキーマ名、テキスト検索サーバーにある関連コレクション名が記述されています。
- | 83 ページの『孤立した索引と欠落した索引の検出』
- | QDBTS_LISTINXSTS という名前の SQL ユーザー定義表関数 (UDTF) を使用して、孤立した索引と欠落した索引を検出できます。

| SYSPROC.SYSTS_REPRIMEINDEX

- | SYSPROC.SYSTS_REPRIMEINDEX ストアード・プロシージャを使用すれば、索引を再作成して初期更新を開始できます。基本表からデータを復元する場合に、このストアード・プロシージャを使用します。
- | 基本表からデータを復元する場合は、SYSTS_UPDATE ストアード・プロシージャの呼び出し中に基本表の更新後の内容に基づいて索引を生成することはできません。その場合は、SYSPROC.SYSTS_REPRIMEINDEX ストアード・プロシージャを呼び出して、索引を再作成できます。
- | 注: テキスト検索索引の同義語ディクショナリーを作成していた場合は、このプロセスによってディクショナリーが削除されます。

| 構文

```
| >>-SYSPROC.SYSTS_REPRIMEINDEX( indexSchema, indexName, options) -<<
```

- | スキーマ修飾子は SYSPROC です。

| パラメーター

| *indexSchema*

- | テキスト検索索引のスキーマを識別します。このパラメーターがヌルの場合、呼び出し側の CURRENT SCHEMA 特殊レジスターの値が使用されます。

- | このパラメーターのデータ・タイプは VARCHAR(128) です。

| *indexName*

- | テキスト検索索引の名前を識別します。索引スキーマと一緒にテキスト検索索引の名前を記述すると、DB2 サブシステムで全文索引を一意的に識別できます。このパラメーターでは、ヌル以外の値を指定する必要があります。

- | このパラメーターのデータ・タイプは VARCHAR(128) です。

| *options*

- | 将来的にこのストアード・プロシージャで追加できるオプションを指定する文字ストリング。

- | **重要:** *options* パラメーターでは、ヌル値を指定する必要があります。そうでない場合は、エラーが生成されます。*options* パラメーターを指定する方法については、以下の例を参照してください。

| SYSTS_REPRIMEINDEX の SQL

```
| CREATE PROCEDURE SYSPROC.SYSTS_REPRIMEINDEX(  
|     IN INDEXSCHEMA VARCHAR(128) CCSID 1208,  
|     IN INDEXNAME VARCHAR(128) CCSID 1208,  
|     IN OPTIONS VARCHAR(32000) CCSID 1208)  
|     EXTERNAL NAME QDBTSLIB.DSN5RPMIDX  
|     DYNAMIC RESULT SETS 0  
|     LANGUAGE C  
|     PARAMETER STYLE SQL
```

```
| MODIFIES SQL DATA
| PROGRAM TYPE MAIN
| COMMIT ON RETURN NO
| INHERIT SPECIAL REGISTERS
```

| 例

| • SQL インターフェースから索引を再作成する場合は、SQL インターフェースから以下のコマンドを入力
| します。

```
| CALL SYSPROC.SYSTS_REPRIMEINDEX('indexSchema1','indexName1','')
```

| 関連資料

| 37 ページの『SYSPROC.SYSTS_UPDATE』

| SYSPROC.SYSTS_UPDATE ストアード・プロシージャを呼び出すことによって、テキスト列の現在の内
| 容に合わせてテキスト検索索引を更新できます。

| SYSPROC.SYSTS_VALIDITYCHECK

| SYSPROC.SYSTS_VALIDITYCHECK SQL ストアード・プロシージャを使用して、索引項目の妥当性を
| 検査できます。

| 構文

| このストアード・プロシージャで *autoFix* パラメーターを指定すれば、無効な項目を修正できます。

```
| >>-SYSPROC.SYSTS_VALIDITYCHECK (indexSchema, indexName, autoFix) -><
```

| スキーマ修飾子は SYSPROC です。

| パラメーター

| *indexSchema*

| テキスト検索索引のスキーマを識別します。このパラメーターがヌルの場合、呼び出し側の CURRENT
| SCHEMA 特殊レジスターの値が使用されます。

| このパラメーターのデータ・タイプは VARCHAR(128) です。

| *indexName*

| テキスト検索索引の名前を識別します。索引スキーマと一緒にテキスト検索索引の名前を記述すると、
| DB2 サブシステムで全文索引を一意的に識別できます。このパラメーターでは、ヌル以外の値を指定
| する必要があります。

| このパラメーターのデータ・タイプは VARCHAR(128) です。

| *autoFix*

| 自動修正が必要かどうかを指定します。このパラメーターの値は、0 または 1 です。それぞれの値の
| 意味は、以下のとおりです。

| **0** 索引の妥当性だけを検査します。

| **1** 索引の妥当性を検査して、無効な項目を修正します。

| 注:

| 0 または 1 以外の値を指定した場合は、0 と見なされます。

| このパラメーターのデータ・タイプは INTEGER です。

| **制約事項:** indexSchema と indexName の両方で *NONE を指定すると、このストアード・プロシージャーは、製品の共通部分の妥当性だけを検査します。

| **SYSTS_VALIDITYCHECK の SQL**

```
| CREATE PROCEDURE SYSPROC.SYSTS_VALIDITYCHECK
|     (IN INDEXSCHEMA VARCHAR(128) CCSID 1208,
|      IN INDEXNAME VARCHAR(128) CCSID 1208,
|      IN AUTOFIX INTEGER)
|     EXTERNAL NAME QDBTSLIB.DSN5VALCHK
|     DYNAMIC RESULT SETS 0
|     LANGUAGE C
|     PARAMETER STYLE SQL
|     MODIFIES SQL DATA
|     PROGRAM TYPE MAIN
|     COMMIT ON RETURN NO
|     INHERIT SPECIAL REGISTERS
```

| **例**

| • 索引の妥当性を検査する場合は、SQL インターフェースから以下のコマンドを入力します。

```
| CALL SYSPROC.SYSTS_VALIDITYCHECK('indexSchema1','indexName1',0)
```

| • 索引の検査と修正を自動的に実行する場合は、以下のようにします。

```
| CALL SYSPROC.SYSTS_VALIDITYCHECK('indexSchema1','indexName1',1)
```

| **同義語ディクショナリー**

| 同義語ディクショナリーを使用して、検索結果の品質を改善できます。

| コレクションに同義語ディクショナリーを追加する操作は、いつでも実行できます。

| 同義語ディクショナリーには、XML ファイルで定義する同義語グループを組み込みます。例を以下に示します。

```
| <?xml version="1.0" encoding="UTF-8"?>
| <synonymgroups version="1.0">
| <synonymgroup>
|   <synonym>Paixão</synonym>
|   <synonym>amor</synonym>
|   <synonym>flor</synonym>
|   <synonym>linda</synonym>
| </synonymgroup>
| <synonymgroup>
|   <synonym>worldwide patent tracking system</synonym>
|   <synonym>wpts</synonym>
| </synonymgroup>
| </synonymgroups>
```

| **コレクションへの同義語ディクショナリーの追加:**

| 同義語ツールを使用して、同義語ディクショナリーを特定のコレクションに追加します。

| 同義語ディクショナリーで同義語グループを指定すると、テキスト検索の結果の品質が向上します。

| OMNIFIND管理者には、同義語ツールを実行するための正しい権限と特権があります。

| コレクションに同義語ディクショナリーを追加するには、以下の手順を実行します。

| 1. 同義語グループを指定して同義語 XML ファイルを作成します (以下の例を参照)。

```
| <?xml version="1.0" encoding="UTF-8"?>
| <synonymgroups version="1.0">
| <synonymgroup>
```

```

|     <synonym>Paixão</synonym>
|     <synonym>amor</synonym>
|     <synonym>flor</synonym>
|     <synonym>linda</synonym>
| </synonymgroup>
| <synonymgroup>
|   <synonym>worldwide patent tracking system</synonym>
|   <synonym>wpts</synonym>
| </synonymgroup>
| </synonymgroups>

```

2. その同義語 XML ファイルをテキスト検索サーバーの任意のディレクトリーにコピーします。
3. 同義語ツールを使用して、同義語ディクショナリーをコレクションに追加します。

append モードまたは replace モードで同義語ディクショナリーを追加できます。append モードで同義語ディクショナリーを追加すると、既存の同義語ディクショナリーに新しい同義語が追加されます。replace モードで同義語ディクショナリーを追加すると、既存の同義語が、テキスト検索索引で定義した新しい同義語に置き換えられます。

オプション	説明
IBM i では、以下のコマンドを (QSH インターフェースで) 入力します。	<pre> synonymTool.sh importSynonym -synonymFile <absolute path to synonym XML file> -collectionName <collection name> -replace <[true false]> -configPath <absolute path to configuration folder> </pre>
Linux サーバーでは、以下のコマンドを入力します。	<pre> synonymTool.sh importSynonym -synonymFile <absolute path to synonym XML file> -collectionName <collection name> -replace <[true false]> -configPath <absolute path to configuration folder> </pre>
Windows サーバーでは、以下のコマンドを入力します。	<pre> synonymTool.bat importSynonym -synonymFile <absolute path to synonym XML file> -collectionName <collection name> -replace <[true false]> -configPath <absolute path to configuration folder> </pre>

XML ファイルの形式が有効でない場合や、XML ファイルが空の場合は、エラー・コードが返されません。

コレクションからの同義語ディクショナリーの削除:

用意されているスクリプトを使用して、コレクションから同義語ディクショナリーを削除します。

OMNIFIND管理者は、同義語ディクショナリーをどのコレクションから削除するのかを指定するために、そのコレクションの名前を取得する必要があります。

スクリプトを実行して、同義語ディクショナリーをコレクションから削除します。

オプション	説明
IBM i では、以下のコマンドを (QSH インターフェースで) 入力します。	<pre> removeSynonym.sh -collectionName <collection name> -configPath <absolute path to configuration folder> </pre>

オプション	説明
Linux サーバーでは、以下のコマンドを入力します。	removeSynonym.sh -collectionName <collection name> -configPath <absolute path to configuration folder>
Windows サーバーでは、以下のコマンドを入力します。	removeSynonym.bat -collectionName <collection name> -configPath <absolute path to configuration folder>

データベースに複数のテキスト検索索引が含まれている場合は、対応するコレクションごとにこのタスクを実行する必要があります。

孤立した索引と欠落した索引の検出

QDBTS_LISTINXSTS という名前の SQL ユーザー定義表関数 (UDTF) を使用して、孤立した索引と欠落した索引を検出できます。

SYSTS_DROP ストアド・プロシージャを呼び出し、そのプロシージャの実行時にサーバーを停止すると、索引が孤立する可能性があります。

QDBTS_LISTINXSTS 関数は、現在の名前空間に含まれているすべての統合ファイル・システム・コレクションとカタログ索引を 1 つの表に結合します。この関数は、どの独立補助記憶域プール (ASP) または *SYSBASE が設定されているかを確認します。さらに、独立 ASP または *SYSBASE で各サーバーのコレクション・ディレクトリーをスキャンします。

*SYSBASE の場合は、/QOpenSys/QIBM/ProdData/TextSearch の下にある各サーバーのディレクトリーがチェックされます。独立 ASP の場合は、/the ASP number/QOpenSys/QIBM/ProdData/TextSearch の下にある各サーバーのディレクトリーがチェックされます。例えば、独立 ASP の番号が 67 であれば、/67/QOpenSys/QIBM/ProdData/TextSearch の下にある各サーバーのディレクトリーがチェックされます。

カタログ索引情報の場合は、QSYS2.SYSTEXTINDEXES カタログ表からデータを取得します。独立 ASP でサーバーをチェックする場合は、この関数を呼び出す前に、ASP グループ設定 (SETASPGRP) コマンドを実行します。

孤立した索引を検出した後に、それらの孤立した索引を統合ファイル・システムから削除する場合は、SYSPROC.SYSTS_REMOVE ストアド・プロシージャまたは 84 ページの『拡張管理』 (adminTool.sh) を使用します。

用語

孤立した索引

サーバーの統合ファイル・システム・ディレクトリーにコレクション (索引) が存在しますが、対応する索引が QSYS2.SYSTEXTINDEXES カタログに記録されていません。

欠落した索引

QSYS2.SYSTEXTINDEXES カタログに索引レコードが存在しますが、対応するコレクション・ディレクトリーが存在しません。

構文

```
>>-QDBTS_LISTINXSTS(--null--)-<<
```

戻りの形式

QDBTS_LISTINXSTS 関数は、検出した索引の情報を 1 つの表の形で返します。この UDTF を作成するための SQL コマンドは、以下のとおりです。

LISTINXSTS UDTF の SQL

```
CREATE FUNCTION QDBTSLIB.QDBTS_LISTINXSTS()  
  RETURNS TABLE(COLLECTIONNAME VARCHAR(255),  
                INDEXID INTEGER,  
                INDEXSCHEMA VARCHAR(128),  
                INDEXNAME VARCHAR(128),  
                SERVERID INTEGER)  
  SPECIFIC qdbts_listinxsts  
  SCRATCHPAD  
  NO FINAL CALL  
  LANGUAGE C++  
  PARAMETER STYLE DB2SQL  
  EXTERNAL NAME 'QDBTSLIB/QDBTSSP(checkIndex)';
```

例

- すべての孤立した索引を検出します。

```
SELECT COLLECTIONNAME, SERVERID  
  FROM TABLE(QDBTSLIB.QDBTS_LISTINXSTS()) AS T  
  WHERE T.INDEXSCHEMA IS NULL AND T.INDEXNAME IS NULL
```

- すべての欠落した索引を検出します。

```
SELECT INDEXSCHEMA, INDEXNAME  
  FROM TABLE(QDBTSLIB.QDBTS_LISTINXSTS()) AS T  
  WHERE T.COLLECTIONNAME IS NULL
```

- サーバー ID 2、独立 ASP iaspXXX で孤立した索引を検出します。

```
CONNECT TO iaspXXX
```

SQL:

```
SELECT T.COLLECTIONNAME, S.SERVERPATH  
  FROM TABLE(QDBTSLIB.QDBTS_LISTINXSTS())  
  AS T LEFT OUTER JOIN QSYS2.SYSTEXTSERVERS S ON (T.SERVERID = S.SERVERID)  
  WHERE T.INDEXSCHEMA IS NULL AND T.INDEXNAME IS NULL AND T.SERVERID = 2
```

関連資料

35 ページの『SYSPROC.SYSTS_DROP』

SYSPROC.SYSTS_DROP ストアード・プロシージャを呼び出すことによって、

SYSPROC.SYSTS_CREATE ストアード・プロシージャで定義したテキスト検索索引を削除できます。

77 ページの『SYSPROC.SYSTS_REMOVE』

SYSPROC.SYSTS_REMOVE SQL ストアード・プロシージャを使用して、孤立した索引を削除できます。

107 ページの『QSYS2.SYSTEXTINDEXES 管理表』

QSYS2.SYSTEXTINDEXES 管理表では、各テキスト検索索引の情報を確認できます。それぞれのテキスト検索索引ごとに、名前、スキーマ名、テキスト検索サーバーにある関連コレクション名が記述されています。

拡張管理

管理ツールを使用して拡張管理を実行できます。

管理ツールは、DB2 for i OmniFind テキスト検索サーバーの実行中でも使用できます。

- | 管理ツールを使用して、以下のタスクを実行できます。
- | • コレクションの状況 (存在する文書の数など) の確認
- | • 孤立したコレクションの削除
- | • サーバーのバージョンの報告
- | • テキスト検索サーバーにあるすべてのコレクションの報告

| コマンド

| 管理ツールを実行するためのコマンドは、テキスト検索サーバーのインストール先のオペレーティング・システムによって異なります。そのコマンドは、実行するタスクによっても異なります。

| 表 29. コレクションの状況を確認するためのコマンドと孤立したコレクションを削除するためのコマンド

IBM i (QSH インターフェース)	Linux サーバー	Windows サーバー
adminTool.sh <i>[-delete status]</i> -collectionName <collection name> -configPath <absolute path to configuration folder>	adminTool.sh <i>[-delete status]</i> -collectionName <collection name> -configPath <absolute path to configuration folder>	adminTool.bat <i>[-delete status]</i> -collectionName <collection name> -configPath <absolute path to configuration folder>

| 表 30. サーバーのバージョンを表示するためのコマンドとすべてのコレクションを報告するためのコマンド

IBM i (QSH インターフェース)	Linux サーバー	Windows サーバー
adminTool.sh <i>[-version]</i> -configPath <absolute path to configuration folder>	adminTool.sh <i>[-version reportAll]</i> -configPath <absolute path to configuration folder>	adminTool.bat <i>[-version reportAll]</i> -configPath <absolute path to configuration folder>

| オプション

- | *status*
| コレクションの状況を確認します。
- | *delete*
| 孤立したコレクションを削除することを指定します。
- | *version*
| サーバーのバージョンを表示します。
- | *reportAll*
| テキスト検索サーバーにあるすべてのコレクションのレポートを生成します。

| 例

- | Linux サーバーで、サーバーのバージョンを確認するには、以下のコマンドを入力します。
- | `adminTool.sh -version -s <absolute path to server config.xml>`
- | Windows サーバーを使用する場合は、対応する .bat スクリプトが用意されています。

| ServerInstance ツール

- | ServerInstance ツールを使用して、*SYSBASE または独立補助記憶域プール (ASP) でサーバーを作成/削除できます。さらに、サーバーのファイルを DB2 for i OmniFind テキスト検索サーバーのインストール先のサーバーにリンクするために、ServerInstance ツールを使用することも可能です。
- | デフォルトでは、DB2 for i OmniFind テキスト検索サーバーは、/QOpenSys/QIBM/ProdData/TextSearch/server1 ディレクトリの下にインストールされます。

- | ServerInstance ツールを使用して以下のタスクを実行してから、そのツールを使用して *SYSBASE で server1 を停止できます。
- | • *SYSBASE または独立 ASP でサーバーを作成するタスク
- | • *SYSBASE または独立 ASP でサーバーを削除するタスク
- | • サーバーのファイルを server1 にリンクするタスク

| 構文

```
| ServerInstance.sh -[create|delete|relink]  
| -servernum <server number>  
| (-port <port>)  
| (-device <device name>)
```

| コマンド・オプション

| *create*

| サーバーを作成します。

| *delete*

| サーバーを削除します。

| *relink*

| サーバーのファイルを server1 にリンクします。

| 注: プログラム一時修正 (PTF) SI31548 をシステムにインストールした後であれば、このオプションは必要ありません。その PTF をインストールすると、リンク操作がシステムによって自動的に処理されるようになります。

| パラメーター

| *servernum*

| サーバーの番号を指定します。例えば、サーバー番号 3 のサーバーを作成すると、そのサーバーのディレクトリーは、/QOpenSys/QIBM/ProdData/TextSearch/server3 になります。

| *port*

| サーバーのポートを指定します。このパラメーターは、サーバーを作成する場合にのみ必要です。

| *device*

| 独立 ASP の名前を指定します。このパラメーターは、独立 ASP で操作を実行する場合にのみ必要です。

| 例

- | • サーバー番号 2、ポート番号 50000 のサーバーを *SYSBASE で作成するには、以下のようになります。
| ServerInstance.sh -create -servernum 2 -port 50000
- | • サーバー番号 3、ポート番号 50001 のサーバーを独立 ASP iasp1 で作成するには、以下のようになります。
| ServerInstance.sh -create -servernum 3 -port 50001 -device iasp1
- | • サーバー番号 2 のサーバーを *SYSBASE で削除するには、以下のようになります。
| ServerInstance.sh -delete -servernum 2
- | • サーバー番号 3 のサーバーを独立 ASP iasp1 で削除するには、以下のようになります。
| ServerInstance.sh -delete -servernum 3 -device iasp1
- | • サーバーのファイルを *SYSBASE のサーバー番号 2 のサーバーにリンクするには、以下のようになります。
| ServerInstance.sh -relink -servernum 2 -device iasp1

| `ServerInstance.sh -relink -servernum 2`

| • サーバーのファイルを独立 ASP `iasp1` のサーバー番号 `3` のサーバーにリンクするには、以下のよう
| します。

| `ServerInstance.sh -relink -servernum 3 -device iasp1`

| **DB2 for i OmniFind テキスト検索サーバーの独立 ASP に関する注意点**

| 独立補助記憶域プール (ASP) でテキスト検索索引を管理できます。ASP は、複数のシステムの間で切り替
| えることも可能なので、追加の注意点があります。

| OMNIFINDのインストール時に、ローカル・テキスト検索サーバーを作成します。独立 ASP の場合は、管
| 理者が独立 ASP の作成後に `ServerInstance` ツール (`ServerInstance.sh`) を使用してローカル・テキスト検
| 索サーバーを作成します。

| 独立 ASP でローカル・テキスト検索サーバーを作成すると、独立 ASP のファイル・システムに索引デー
| タが存在するようになります。独立 ASP を別のシステムに切り替えた場合でも、そのデータは有効です。
| 管理者は、独立 ASP グループごとにローカル・テキスト検索サーバーを 1 回だけ作成する必要があります。
| す。

| 独立 ASP に存在するテキスト検索索引は、その独立 ASP で定義されているテキスト検索サーバーに組み
| 込む必要があります。ジョブを独立 ASP に接続したときに、別の独立 ASP グループまたはシステム ASP
| で定義されているテキスト検索サーバーを表示することはできません。

| `myiasp` という名前の独立 ASP でテキスト検索サーバーを作成するには、以下のようになります。

- | 1. 構成状況処理 (WRKCFGSTS) CL コマンドまたは System i ナビゲーターを使用して、独立 ASP をオ
| ンに変更します。
- | 2. ASP グループ設定 (SETASPGRP) CL コマンドを使用して、独立 ASP グループの名前空間に接続しま
| す。
- | 3. `ServerInstance.sh` スクリプトを使用して、テキスト検索サーバーを作成します。

| 使用できる QSH コマンドの例を以下に示します。

| `/QOpenSys/QIBM/ProdData/TextSearch/ServerInstance.sh -create`
| `-servernum 2 -port nnnnn -device myiasp`

| このコマンドの `nnnnn` は、使用するサーバーの有効なポート番号です。このポート番号は、独立 ASP
| の切り替え先として使用できるすべてのシステムで有効なポート番号でなければなりません。

| 独立 ASP グループでテキスト検索サーバーを定義したら、管理ストアード・プロシージャを使用して、
| そのテキスト検索サーバーを開始/停止できます。その種のストアード・プロシージャを使用して、テキ
| スト検索索引の作成、削除、更新の操作を実行することも可能です。

| 注: 独立 ASP に存在するスケジュール設定済みの更新操作でいずれかの索引の独立 ASP をオンに変更す
| ると、ジョブ・スケジューラー項目が追加されます。それらのジョブ・スケジューラー項目を使用すれ
| ば、独立 ASP をシステム間で切り替えた場合でも、スケジュール済みの更新操作を続行できます。

| **テキスト検索索引と独立 ASP の使用に関する制限**

- | • 独立 ASP の切り替えが可能なすべてのシステムで DB2 for i OmniFind テキスト検索サーバーをインス
| トールして、プログラム一時修正 (PTF) のレベルを統一する必要があります。
- | • 表の索引を構築する ASP 以外の ASP では、テキスト検索索引を作成しないでください。

- | • システム・カタログ
| SYSTEXTSERVERS、SYSTEXTINDEXES、SYSTEXTDEFAULTS、SYSTEXTCOLUMNS、
| SYSTEXTCONFIGURATION には、別の ASP グループやシステム ASP で定義されている索引やサーバ
| ーのレコードは組み込まれません。それらのカタログには、ジョブの接続先の独立 ASP グループで定義
| されている索引やサーバの行だけが入ります。
- | • 管理ストアード・プロシージャを使用して各種の機能を実行できるのは、ジョブの接続先の独立 ASP
| グループで定義されているテキスト検索サーバやテキスト検索索引に限られます。
- | 注: ジョブを独立 ASP グループに接続した場合は、システム ASP に存在する表に列に基づいている場合
| でも、CONTAINS と SCORE の SQL ステートメントを使用できます。

| 高可用性

- | 既存の API とコマンドを使用して、テキスト検索索引を組み込んだ高可用性ソリューションを実装できま
| す。DB2 for IBM i は、テキスト検索索引を認識できるようになり、テキスト検索索引に影響を与える
| DB2 操作の実行中に特別なアクションを実行するようになりました。

| テキスト検索索引、高可用性、データベース管理に関する特別な注意点

- | • テキスト検索索引に対応するビューに対してファイル削除 (DLTF) CL コマンドまたは DROP VIEW
| SQL ステートメントを実行すると、そのテキスト検索索引が削除されます。コミットメント制御の下で
| 削除操作を実行すると、その削除操作は失敗します。
- | • 表または物理ファイルの列に基づくテキスト検索索引を生成し、そのテキスト検索索引と一緒にその表
| または物理ファイルを保管して復元すると、その保管の前にテキスト検索索引で作成したトリガーは有
| 効になりません。テキスト検索索引を後で復元/作成する場合は、索引の作成処理時にトリガーが追加さ
| れます。この方式を使用すれば、その表に基づいて元々作成していたテキスト検索索引を復元/作成しな
| い場合でも、アプリケーションは正しく動作します。

| OmniFind 管理ストアード・プロシージャのジャーナル項目の再生

- | • テキスト検索索引に対応するビューの作成に関するジャーナル項目を (QDBRPLAY API によって) 再生
| すると、そのテキスト検索索引が作成されます。
- | • SYSTS_CREATE または SYSTS_DROP のプロシージャ呼び出しで基本表のトリガーを追加/削除する
| 場合は、ジャーナル項目がログに記録されません。
- | • SYSTS_UPDATE ストアード・プロシージャまたは SYSTS_REPRIMEINDEX ストアード・プロシージ
| ャーを呼び出すと、テキスト検索索引のビューに対するジャーナル項目がログに記録されます。
| QDBRPLAY API を使用してそのジャーナル項目を再生すると、正しいプロシージャ呼び出しが再生
| されます。
- | • スケジュール設定済みの自動更新処理を実行する場合は、索引のジャーナル項目が生成されないので、
| 再生は不可能になります。
- | • SYSTS_ALTER を呼び出すと、ビューに対する変更ジャーナル項目が生成されます。QDBRPLAY API
| を使用してそのジャーナル項目を再生すると、SYSTS_ALTER 呼び出しが再生されます。

| 高可用性に関する推奨事項

- | 高可用性ソリューションを実装する場合の推奨事項を以下にまとめます。
- | • 1 次システムでテキスト検索索引を作成した後に、QDBRPLAY API を使用してバックアップ・システ
| ムでの作成を再生できます。
- | • バックアップ・システムには重複索引が組み込まれ、スケジュール設定済みの更新処理も 1 次システム
| と同じ頻度で発生することになります。

- レコード変更をバックアップ・システムの基本表で再生すると、バックアップ・システムのステージング表にそれらのレコード変更のログが記録されます。
- 1 次システムで SYSTS_UPDATE、SYSTS_REPRIMEINDEX、SYSTS_ALTER を呼び出すと、バックアップ・システムで再生できるジャーナル項目が生成されます。
- お客様の側でテキスト検索索引の IFS ファイルのジャーナル処理や複製を実行する必要はありません。
- SYSTS_CREATE の処理時に追加したトリガーが有効になり、基本表に変更内容を記録できる状態になっていることを確認する必要があります。それらのトリガーは、トリガー名に「QDBTS」という特別な接頭部が付いているので、区別しやすくなっています。

ジャーナル処理済み変更適用 CL コマンド (APYJRNCHG)

ジャーナル処理済み変更適用 (APYJRNCHG) CL コマンドを使用して、OmniFind イベントを再生できません。そのコマンドを使用するユーザーは、ジャーナル項目を再生する順序を慎重に検討する必要があります。QSYS2 にあるステージング表は、SYSTS_UPDATE の再生/起動時に基本表と完全に同期した状態になっていなければなりません。

基本表に適用するレコード変更のログは、APYJRNCHG の処理時にステージング表に記録されません。そのコマンドの実行後に、SYSTS_REPRIMEINDEX ストアード・プロシージャーを呼び出して索引を再作成することが必要になる場合もあります。

関連資料

79 ページの『SYSPROC.SYSTS_REPRIMEINDEX』

SYSPROC.SYSTS_REPRIMEINDEX ストアード・プロシージャーを使用すれば、索引を再作成して初期更新を開始できます。基本表からデータを復元する場合に、このストアード・プロシージャーを使用します。

パフォーマンスの分析

OMNIFINDのパフォーマンスの分析には、正しい索引定義の選択、文書の効率的な処理、選択的な検索の指定などの作業が含まれます。

バックグラウンド

OmniFind の処理は、フロントエンドの「クライアント」ジョブとバックエンドの「サーバー」ジョブの処理の組み合わせであり、両方のジョブの間に通信処理が発生します。通信処理では、標準的なソケット接続を使用します。

クライアント・ジョブでは、索引の作成と保守の処理時に対象のデータベース表からレコードを読み取りまします。さらに、表の変更に関するログを処理し、解析および索引作成の対象となる文書を送信し、SQL 関数の CONTAINS または SCORE を使用したテキスト検索要求を処理します。クライアント・ジョブは、SYSTS_CREATE や SYSTS_UPDATE などのプロシージャー、または CONTAINS や SCORE を使用した照会を実行するジョブです。クライアント・ジョブのパフォーマンスは、データベース操作を高速で処理できるかどうか、テキスト文書を効率的に取得してサーバー・ジョブに転送できるかどうかによって左右されます。

サーバー・ジョブでは、クライアント・ジョブから送られてきた文書を解析し、関連するテキスト索引を挿入や削除の操作によって管理し、索引の中で検索要求を処理します。サーバー・ジョブのパフォーマンスは、クライアント・ジョブと効率的に通信できるかどうか、文書を効率的に処理できるかどうかによって左右されます。通常は、1 つのサーバー・ジョブで複数のクライアント・ジョブにサービスを提供します。したがって、サーバー・ジョブは、複数のクライアントを処理するためのマルチスレッド・ジョブになります。

テキスト索引は、基礎データベース表の変更によって直接的に管理されるわけではありません。むしろ、データベース・トリガーとステージング表の組み合わせによって表のレコード変更がログに記録されます。ステージング表には、更新のタイプ (挿入、更新、削除) と、基本表のどのレコードが変更されたのかを示す標識が記録されます。ログに記録されるのは、変更されたレコードを識別するための情報だけであり、テキストそのものが書き込まれるわけではありません。

次回の SYSTS_UPDATE の実行が成功するまで、表のレコード変更はテキスト索引に反映されません。したがって、CONTAINS または SCORE を使用した検索にも反映されません。

テキスト索引の初期更新の後の更新処理のことをインクリメンタル更新といいます。この種の更新では、最後の更新処理以降に基礎表で発生したレコード変更に基づいて、テキスト索引に文書を追加したり、テキスト索引から文書を削除したりします。

基本表で発生した変更は、ステージング表に登録されます。インクリメンタル更新では、ステージング表に基づいて、基本表のどのレコードが変更されているのかを判別します。その後、それらのレコードを基本表から読み取って、更新後の内容を索引に反映します。

初期更新は、インクリメンタル更新よりも行単位の効率が高いプロセスです。基本表を処理するとき、ステージング表も処理する必要はありません。したがって、基礎表に最初にデータを設定してからテキスト索引で初期更新を実行する、というのが、パフォーマンスの観点からして望ましい手法であるといえます。この手法を採用すれば、文書を索引に取り込むための時間を最小化できます。

パフォーマンスに関連した重要な構成オプションとして、UPDATEAUTOCOMMIT があります。この値では、データベース・クライアント・ジョブが文書処理に割り込む頻度を定義します。クライアントは、サーバー・ジョブが、送られてきたすべての文書を処理したことを確認する時点まで待機します。

UPDATEAUTOCOMMIT をチェックポイント方式として使用すれば、データベースが完了済みの処理の境界を設定できるようになります。

索引の更新がいったん中断され、後から続行される場合は、チェックポイント境界でプロセスが再開されることになります。どんな割り込みの場合も、チェックポイント境界によって文書の流れが強制的に中断され、クライアントとサーバーの間のパイプラインで文書がクリアされます。このような開始と停止のプロセスによって、パフォーマンスはかなりの悪影響を受けます。

UPDATEAUTOCOMMIT のデフォルト値は 100 であり、その場合は、チェックポイントが頻繁に発生します。通常、それよりも大きな値を設定すると、SYSTS_UPDATE 呼び出しの応答時間のパフォーマンスは向上します。一方、大きな値を設定すれば、更新をいったんキャンセルして再開する場合のリカバリー時間が長くなります。

UPDATEAUTOCOMMIT に大きな値を設定すると、応答時間のパフォーマンスは向上します (ゼロを設定した場合は、チェックポイントが発生しません)。ただし、チェックポイントが存在しない状態で初期更新をキャンセルすると、OmniFind は、索引の作成の最初の時点から処理をやり直さなければならなくなります。

一部のお客様にとっては、パフォーマンスとチェックポイントによるリカバリーの適度なバランスを考えると、5000 から 20000 程度の値を設定するのが望ましいといえます。

正しい索引定義の選択

テキスト索引には、4 つの FORMAT 構成タイプ (TEXT、HTML、XML、INSO) のいずれかを指定できます。

1. TEXT は、基本的に最も効率の高い形式です。テキストは、データベース・レコードから読み取られ、テキスト・サーバーに送信され、サーバーによって直接処理されます。
 2. HTML (HyperText Markup Language) を使用するの、文書が HTML の形式になっていることがわかっている場合です。テキストは、文書内のマークアップ制御値を無視するかどうかを判断しながら処理されます。
 3. XML (eXtensible Markup Language) を使用するの、文書が XML の形式になっていることがわかっている場合です。文書内の要素、属性、階層構造をたどって、文書の内容の構造的な性質を考慮した特別な判断が行われます。索引に XML のマーキングを設定すると、CONTAINS 関数と SCORE 関数で xpath (xpath) 検索言語を使用して XML 検索を実行することが可能になります。
 4. INSO (INSide Out) を使用するの、文書を徹底的に処理しなければならない場合です。この形式の場合、各文書の内容が単純なテキストの範ちゅうを超えた内容になっていると見なされます。事前分析によって、各文書のタイプが判別されてから、プレーン・テキストに変換されます。INSO 文書は通常、ワード・プロセッサ・プログラムによって生成されたリッチ・テキストのワード・プロセッシング文書です。
- パフォーマンスの観点からすれば、文書の索引を生成する処理は、TEXT 形式から INSO 形式に移るにつれて増えていきます。INSO 形式では、単純なテキストを処理したとしても、解釈が発生する分、TEXT 形式よりもオーバーヘッドが大きくなります。データベース列に入っている文書が単純なテキストであれば、TEXT を使用してください。

LANGUAGE 構成オプションは、絶対に必要であるというわけではありません。OmniFind が解析内容に基づいて文書の言語を判別するからです。ただし、文書の言語がわかっている場合は、LANGUAGE オプションでその言語を指定することによって、パフォーマンスを向上させることができます。そのように指定すれば、必要な解釈処理の量が制限されるからです。

文書の効率的な処理

クライアント・ジョブとサーバー・ジョブは、UTF (Unicode Transformation Format) を使用して文書の情報をやり取りします。クライアントは、文書をその形式に変換するために、データベースから文書を読み取り、そのデータを UTF-8 (別称 CCSID 1208) に変換します。すべてのテキストを UTF-8 に変換してからサーバーに送信することになります。

索引を生成するテキスト列を CCSID 1208 に設定して、その変換処理を回避すれば、パフォーマンスを改善し、文書処理プロセスの効率を高めることができます。

選択的な検索の指定

テキスト検索は、SQL 照会ステートメントに組み込まれている CONTAINS 関数または SCORE 関数に基づいて実行されます。この種の検索処理では、関数の検索基準と、検索対象の列に関連した文書とが比較されます。そのようにして一致レコードが識別され、選択されます。クライアント・ジョブは、検索要求をサーバーに送信し、一致項目が見つかったかどうかに関する応答を受信します。

どんな検索基準の場合も、選択的であればあるほど、検索の効率が高くなります。一般的な語句 (例えば、英語の「the」など) を検索すると、多数の一致項目が検出され、パフォーマンスに悪影響が及びます。現実的に考えても、その種の検索で意味のある情報を取得できる可能性は低いといえます。より選択的な検索語句を指定すれば、一致項目の数を絞って、より意味のある結果を取得できるようになります。

通常、SQL ステートメントの WHERE 文節で CONTAINS を使用する場合は、他の基準を AND で結合するとパフォーマンスが向上します。例を以下に示します。

```
| SELECT bn, pubdate, description  
| FROM myBooks  
| WHERE CONTAINS(description,'Alladin') = 1  
| AND Pubdate > '2004-01-01'
```

| CONTAINS と SCORE の使用

| 最適化プログラムは、CONTAINS 関数と SCORE 関数のパフォーマンスを改善するために、それらの組み込み関数を内部で結合して 1 つのユーザー定義表関数 (UDTF) に置き換える、という処理を実行できます。その UDTF からは、一致文書のリストが 1 つの結果として返されます。その UDTF 処理は、多くの場合、レコードごとに組み込み関数を呼び出して一致するかどうかを確認する、という代替プロセスよりもパフォーマンスが高くなります。

| 最適化プログラムによって UDTF での書き直しを実行するには、CONTAINS 関数を以下のように記述する必要があります。

- | • SQL ステートメントの WHERE 文節に組み込む必要があります。
- | • WHERE 文節に組み込む他のすべての述部を AND で結合する必要があります。
- | • 比較の結果値が 1 であるという条件を記述する必要があります。

| 例えば、以下のような文節があるとします。

```
| WHERE CONTAINS(MyDocuments, 'java performance') = 1
```

| この文節については、最適化プログラムが UDTF 形式での書き直しを実行できます。一方、以下のような文節があるとします。

```
| WHERE CONTAINS(MyDocuments, 'java performance') = 1 OR price >100
```

| この文節は、CONTAINS と他の述部が OR で結合されているので、書き直しができません。

| 注: UDTF での書き直しを有効にした場合でも、最適化プログラムは、コスト比較を実行して最適なプランを選択します。

| 関連資料

| 40 ページの『CONTAINS』

| CONTAINS 関数では、検索引数として指定する基準を使用して、テキスト検索索引を検索できます。この関数は、一致項目が見つかったかどうかを示す結果を返します。

| 43 ページの『SCORE』

| SCORE 関数では、検索引数として指定する基準を使用して、テキスト検索索引を検索できます。この関数は、文書が照会内容に一致する比率を測定した関連度スコアを返します。

| 37 ページの『SYSPROC.SYSTS_UPDATE』

| SYSPROC.SYSTS_UPDATE ストアード・プロシージャを呼び出すことによって、テキスト列の現在の内容に合わせてテキスト検索索引を更新できます。

| 21 ページの『SYSPROC.SYSTS_CREATE』

| SYSPROC.SYSTS_CREATE ストアード・プロシージャを呼び出すことによって、テキスト検索索引を作成できます。このストアード・プロシージャでは、テキスト列に基づいてテキスト検索索引を作成します。そのテキスト検索索引は、CONTAINS 関数または SCORE 関数を組み込んだ SQL 照会で使用できます。

| トランザクションに関する注意点

| データベース表の基礎データに基づいてテキスト索引を更新する頻度を決定するときには、それぞれの環境の特性を検討する必要があります。

従来型のデータベース・ユーザーにとっての大きな検討事項は、トランザクション境界とトランザクション処理の概念です。古典的な例としてよく取り上げられるのは、口座間の預金の転送という銀行取引（トランザクション）です。その転送は、1つのトランザクションと考えられます。つまり、転送は、発生するか発生しないかのいずれかです。顧客の側からすれば、1つの口座から預金が引き落とされてから、しばらくの間は別の口座にその分が入金されない、という状況は望ましくありません。一方、銀行の側からすれば、トランザクションが最終的に完了するまで、しばらくの間は両方の口座に同じ預金が計上される、という状況は望ましくありません。要は、変更が発生したら、その変更をすぐに反映する必要がある、という考え方で

一方、情報管理の分野では、その種の遅延が許容範囲内であったり、むしろ予期すべき事柄であったりする事例が多くあります。従来型のデータウェアハウスの設計では、データウェアハウスの内容が運用データよりも数時間以上の単位で遅れる、という状況がしばしば発生します。そのような遅延が許容されている背後には、データウェアハウスに対する現実的な見方や、予測可能で整合性の高いデータを望む気持ちがあります。

非構造化テキスト検索は、その2つの境界の中間あたりに位置しています。現実的に考えれば、テキスト文書の索引作成は、文書に含まれている各単語の基礎的な意味を分析して分類していく大掛かりなプロセスです。テキスト検索では、「mice」（mouseの複数形）の検索で「mouse」が含まれている文書も検出できません。そのような検索結果が得られるのは、索引作成の技法によって各単語を基礎的な意味に応じて分類できるからです。文書の索引を作成するときその分析を行うのは、その後の検索をできるだけ高速で処理できるようにするためです。

従来型のデータベース索引のユーザーは、データベース表のデータの状態が索引に反映されることを期待します。そのような期待は、テキスト索引では通用しません。テキスト索引の内容には、最後の更新処理（SYSTS_UPDATE）を実行した時点での表の状態が反映されます。

変化の激しい環境では、どの時点でも、表の現在の状態がテキスト索引に反映されている可能性は低いといえます。一方、予測可能な環境、つまり、データベース表の更新頻度が低かったり、データベース表の更新をバッチ・モードで実行したりする環境では、表の更新後というタイミングでテキスト索引の更新を実行すれば、表の状態を正確に反映することが可能になります。

テキスト索引に対する期待を適度なレベルにしておくことは重要です。テキスト索引の21ページの『SYSPROC.SYSTS_CREATE』または37ページの『SYSPROC.SYSTS_UPDATE』でUPDATE FREQUENCY オプションを使用するか、SYSTS_UPDATE プロシージャを明示的に呼び出して、テキスト索引の内容を適切な頻度で更新するようにしてください。

バルク・データ・ロードを実行するような静的な環境では、バルク・ロードの完了後というタイミングでテキスト検索の更新を実行するのが妥当です。トランザクション指向の強い環境では、UPDATE FREQUENCY に短い期間の値を設定するか、SYSTS_UPDATE プロシージャを頻繁に呼び出すようにします。一般に、更新の頻度を上げれば、その分だけマシンに対する作業負荷は大きくなります。

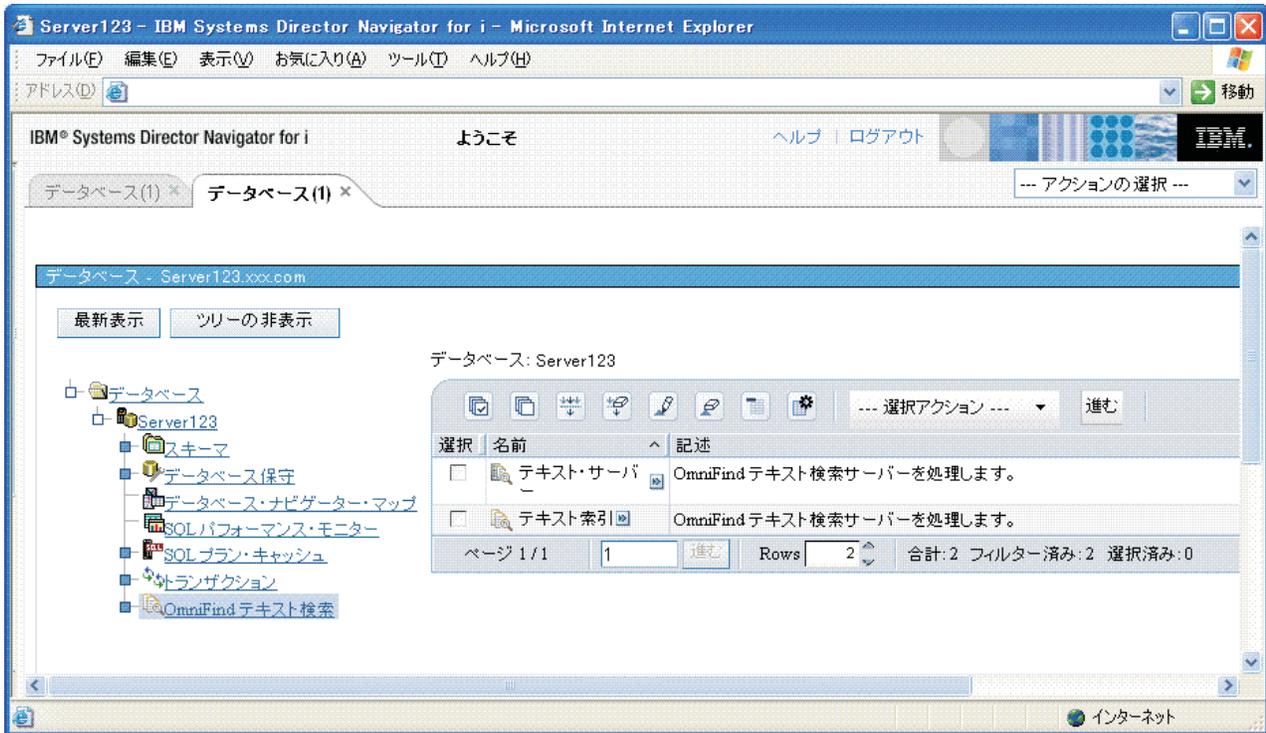
関連資料

107ページの『QSYS2.SYSTEXTINDEXES 管理表』
QSYS2.SYSTEXTINDEXES 管理表では、各テキスト検索索引の情報を確認できます。それぞれのテキスト検索索引ごとに、名前、スキーマ名、テキスト検索サーバーにある関連コレクション名が記述されています。

IBM Systems Director Navigator for i の使用

IBM Systems Director Navigator を使用して、OmniFind のテキスト検索サーバーとテキスト検索索引を管理できます。

1. 「IBM Systems Director Navigator」ウィンドウで、使用するシステムを展開します。
2. 「データベース」を展開します。
3. 処理対象のデータベースを展開します。
4. 「OmniFind テキスト検索」を選択します。

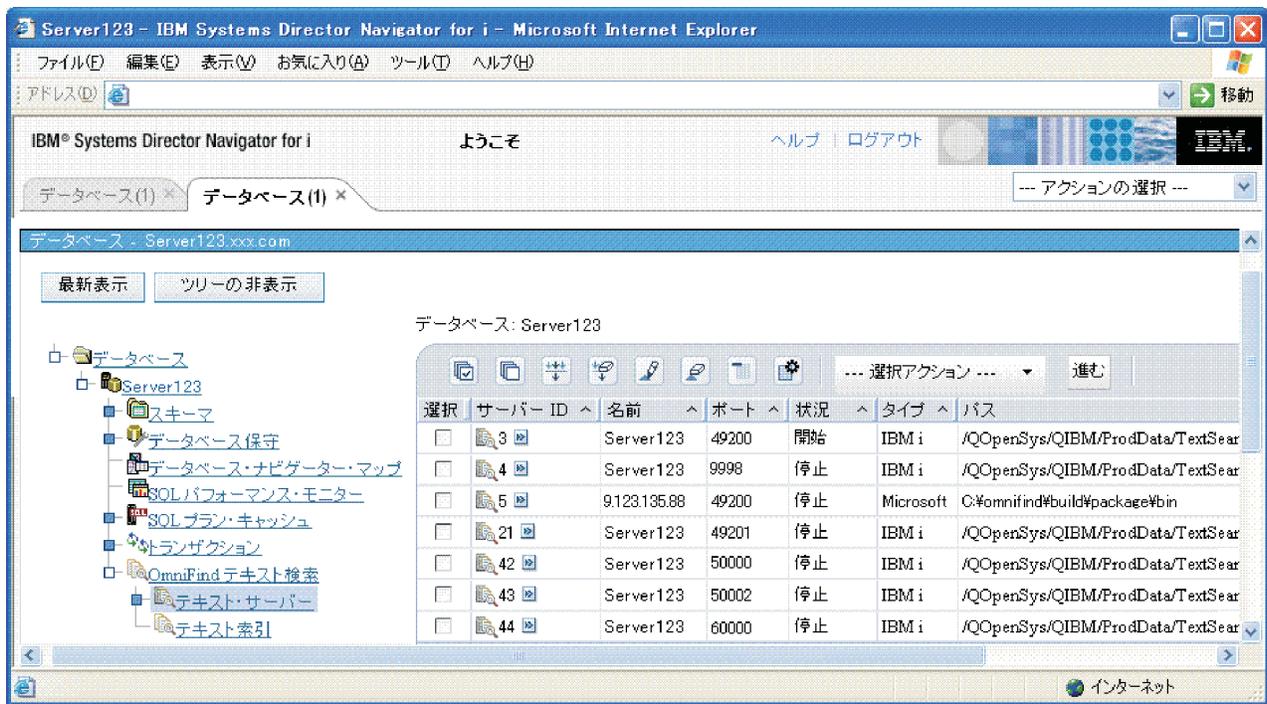


テキスト検索サーバーの操作

IBM Systems Director Navigator for i を使用して、OmniFind テキスト検索サーバーを開始/停止し、テキスト検索索引を作成できます。

OmniFind テキスト・サーバーの状況の表示:

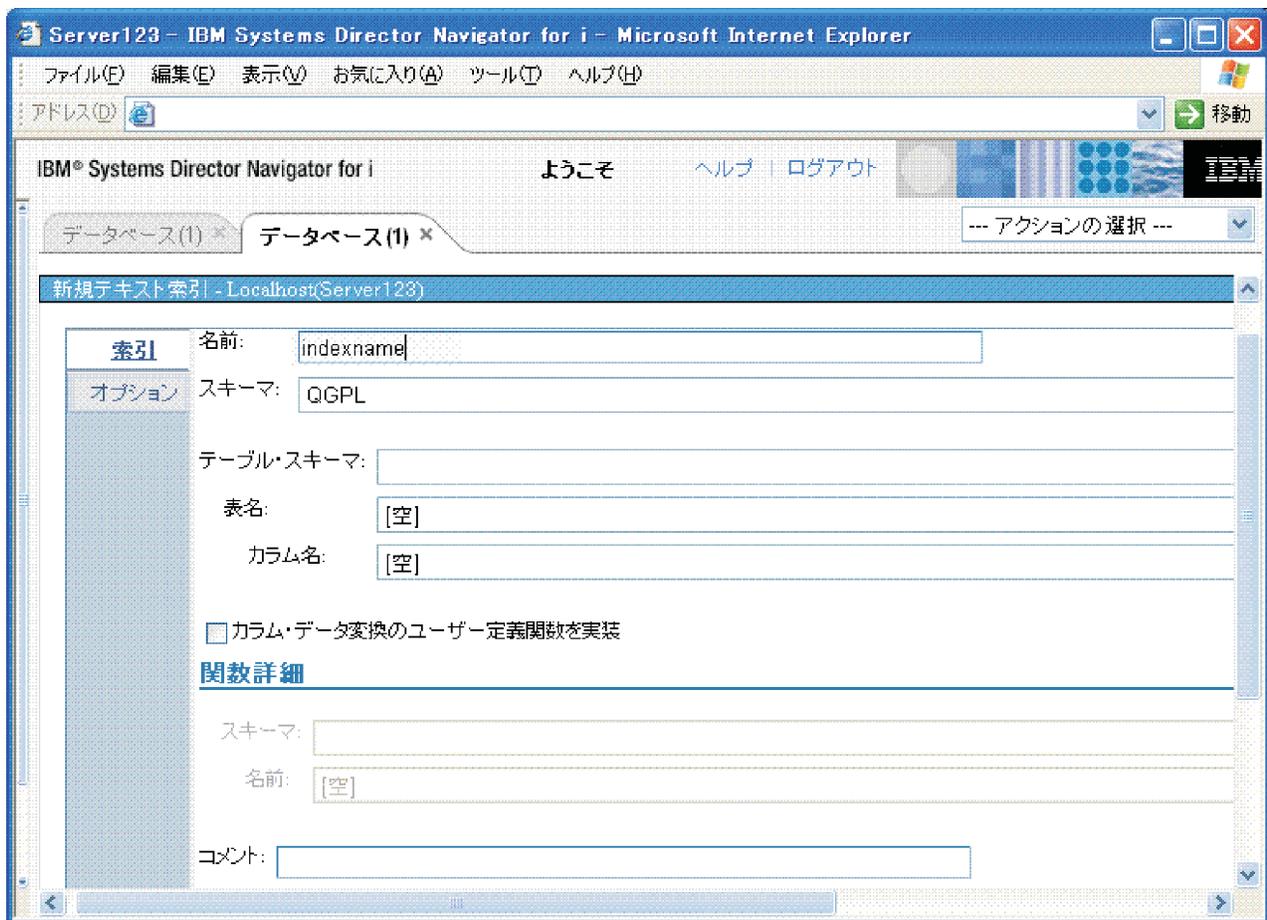
1. 右のペインで「テキスト・サーバー」フォルダーを選択します。
2. 右のペインで、システムで現在構成済みになっているテキスト検索サーバーの状況を表示します。
3. 「最新表示」を使用して、サーバーのリストを最新表示します。



OmniFind テキスト・サーバーの開始と停止:

1. 右のペインで対象のサーバー ID の前にあるボックスを選択します。
2. 「アクション」メニューから項目を選択します。
 - サーバーの開始
 - サーバーの停止
 - テキスト検索索引の作成 (Create a text search index)

テキスト検索索引の作成:

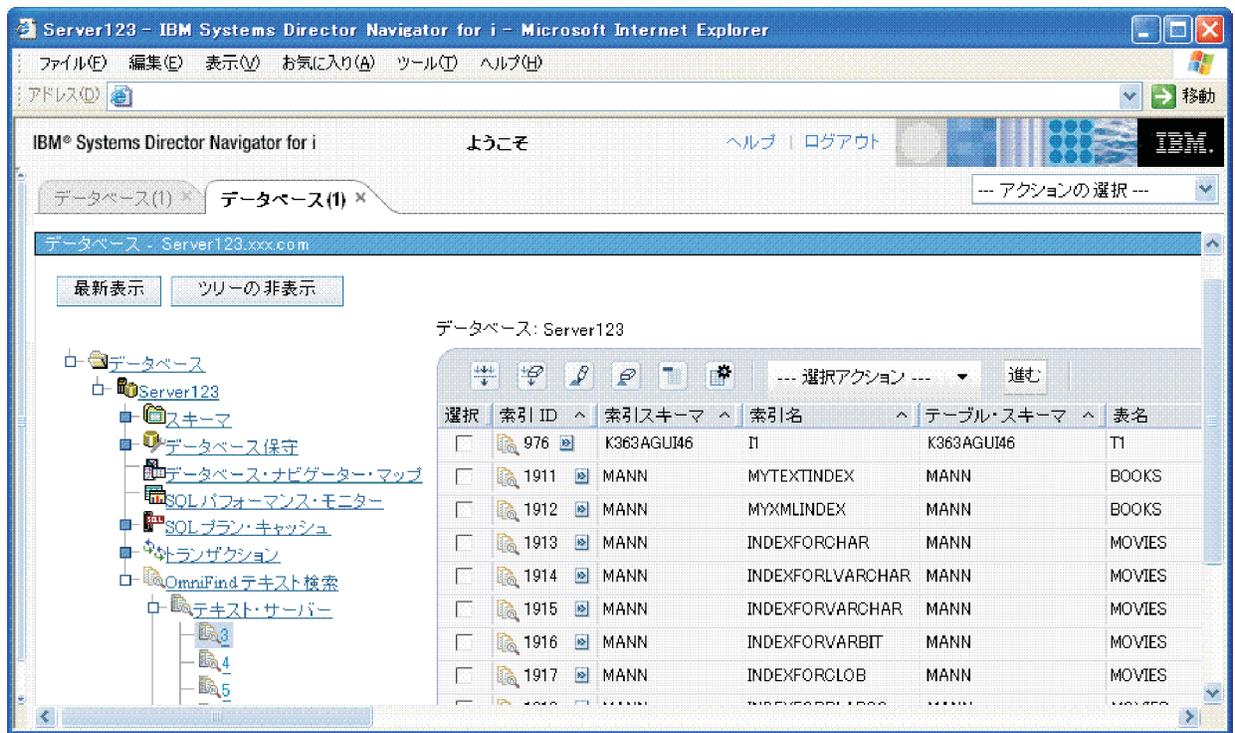


テキスト検索索引の操作

IBM Systems Director Navigator for i を使用して、システムに存在するテキスト検索索引に対して操作を行えます。

テキスト・サーバーの OmniFind テキスト索引の表示:

1. 「IBM Systems Director Navigator」ウィンドウで、使用するシステムを展開します。
2. 「データベース」を展開します。
3. 処理対象のデータベースを展開します。
4. 「OmniFind テキスト検索」を選択します。
5. 「テキスト・サーバー」を選択します。
6. 処理対象のテキスト・サーバーを選択します。そのサーバーの索引が右のペインに表示されます。



このシステムの OmniFind テキスト索引の表示:

1. 「OmniFind テキスト検索」を選択します。
2. 「テキスト索引」を選択します。このシステムの現行区画にあるすべてのテキスト検索索引が表示されます。

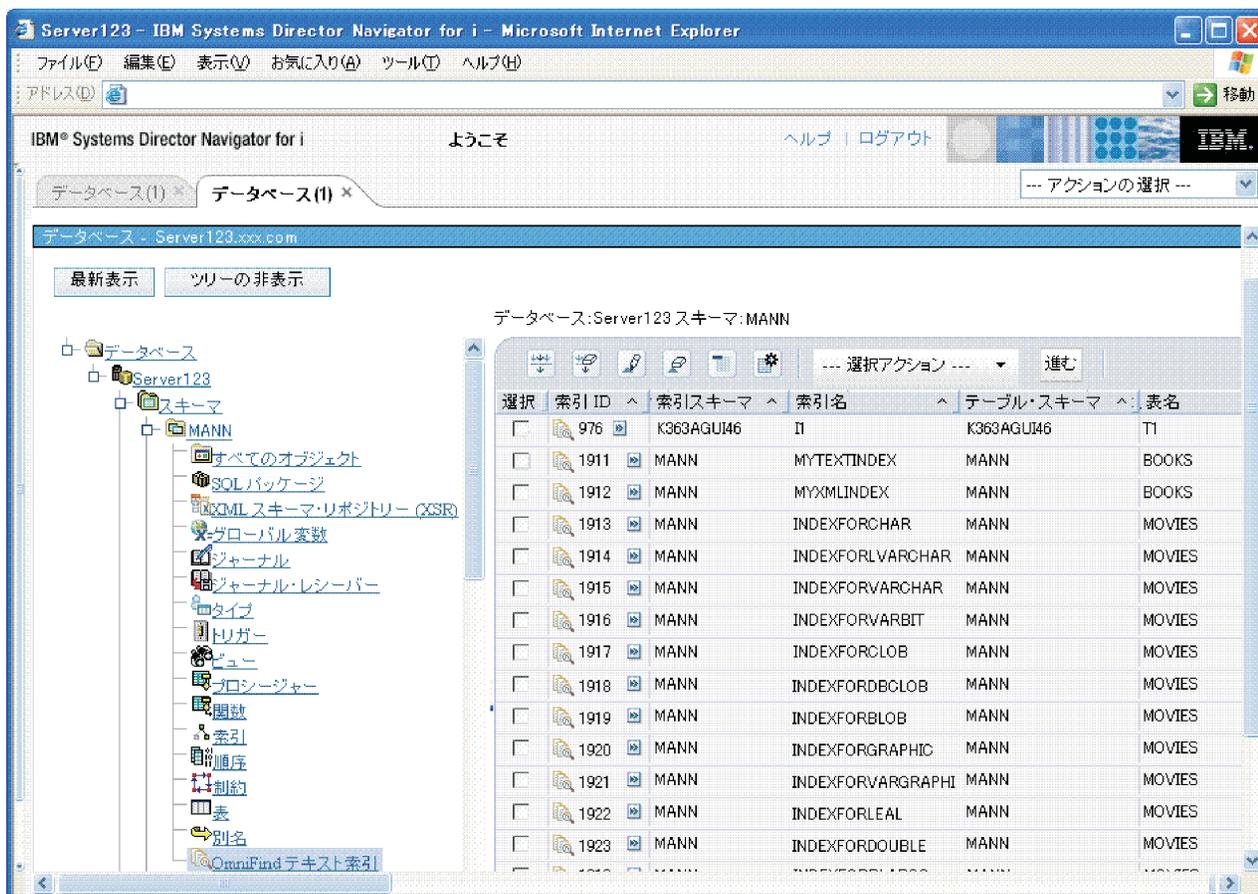


テキスト索引に対する操作の実行:

1. 右のペインで処理対象の索引のボックスを選択します。
2. ペインの上部にあるボックスからアクションを選択します。
 - 更新
 - 再作成 (reprime)
 - 削除
 - 記述の表示
 - 定義の表示
 - 定義の変更 (alter definition)

スキーマの **OmniFind** テキスト索引の表示:

1. 「**IBM Systems Director Navigator**」ウィンドウで、使用するシステムを展開します。
2. 「**データベース**」を展開します。
3. 処理対象のデータベースを展開します。
4. 「**スキーマ**」を展開します。
5. 処理対象のスキーマを展開します。
6. 「**OmniFind テキスト索引**」を選択します。このスキーマのすべてのテキスト検索索引が表示されます。



テキスト検索索引の作成状況の表示:

IBM Systems Director Navigator を使用して、データベースによってテキスト索引が作成されている状況を表示できます。このビューは、アプリケーションでテキスト検索索引を使用できるようになる時点を確認するために役立ちます。

作成中のテキスト検索索引を表示するには、以下の手順を実行します。

1. 「IBM Systems Director Navigator」ウィンドウで、使用するシステムを展開します。
2. 「データベース」を展開します。
3. 処理対象のデータベースを展開します。
4. 「データベース保守」を選択します。
5. 「テキスト索引のビルド」を選択します。

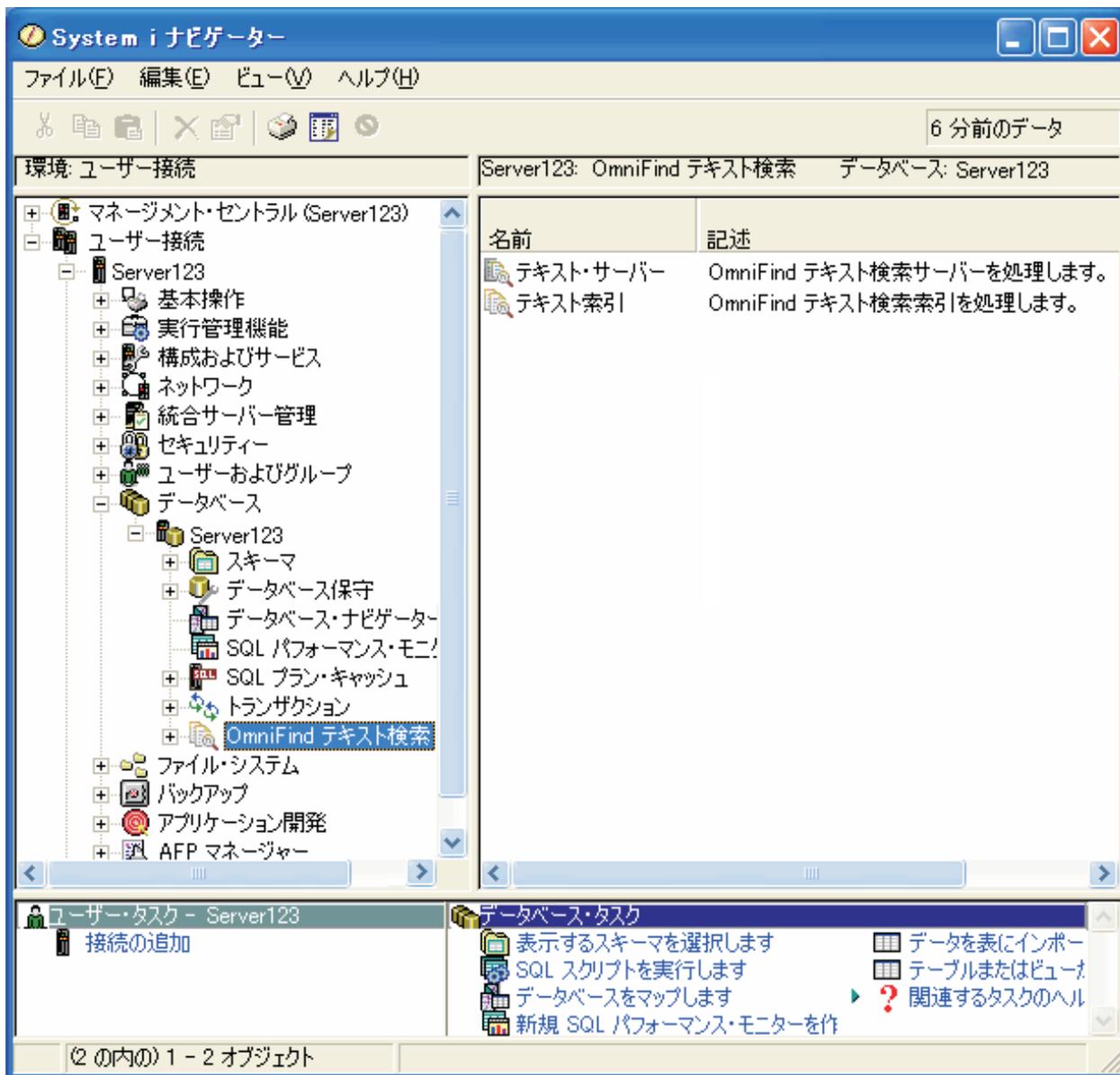
このパネルには、進行中のテキスト検索索引の更新処理だけが表示されます。システムで現在実行中の更新処理や再作成処理がない場合は、空になります。



System i ナビゲーターの使用

System i ナビゲーターを使用して、OmniFind のテキスト検索サーバーとテキスト検索索引を管理できます。

1. 「System i ナビゲーター」ウィンドウで、使用するシステムを展開します。
2. 「データベース」を展開します。
3. 処理対象のデータベースを展開します。
4. 「OmniFind テキスト検索」を選択します。

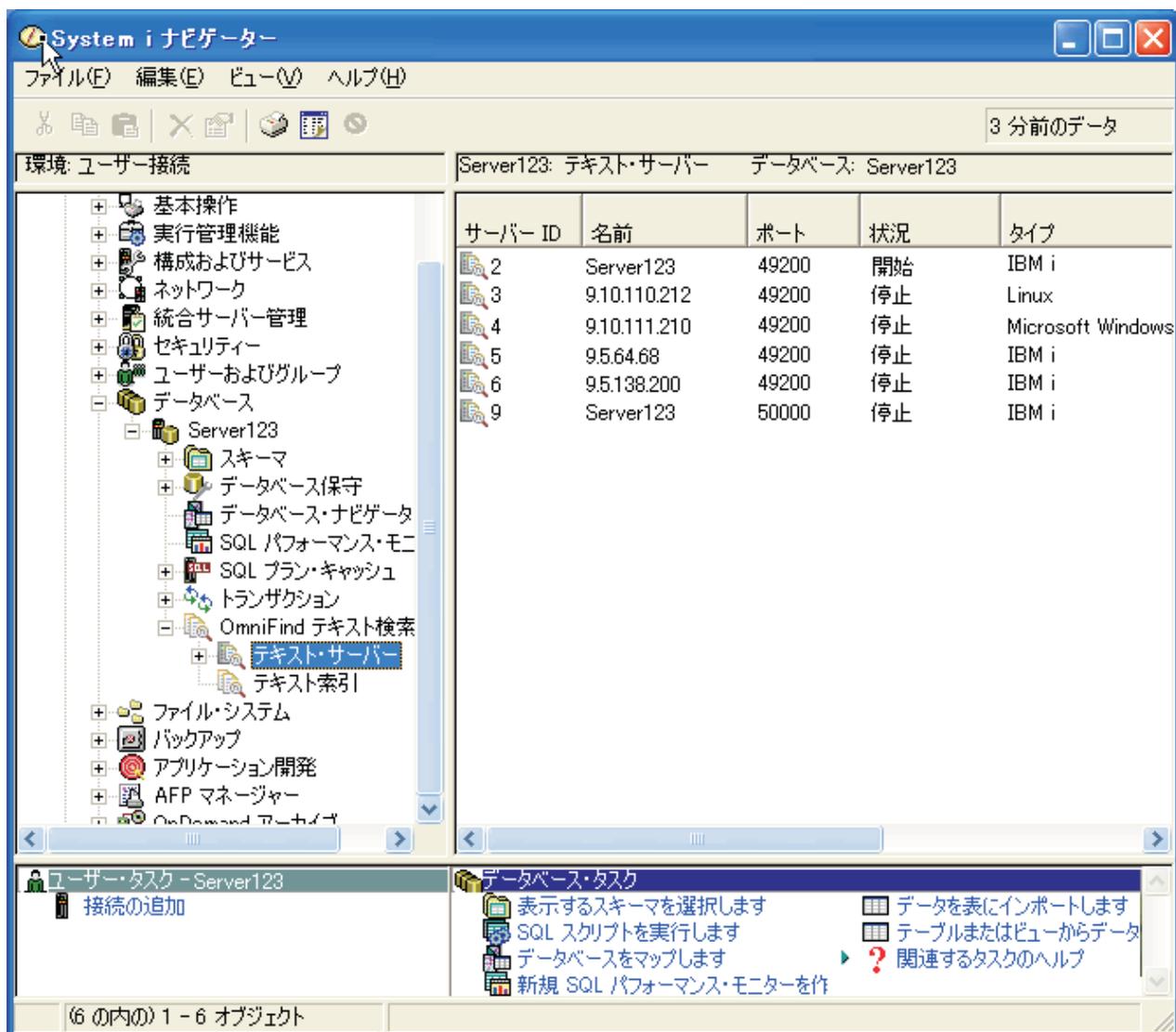


テキスト検索サーバーの操作

System i ナビゲーターを使用して、OmniFind テキスト検索サーバーを開始/停止し、テキスト検索索引を作成できます。

OmniFind テキスト・サーバーの状況の表示:

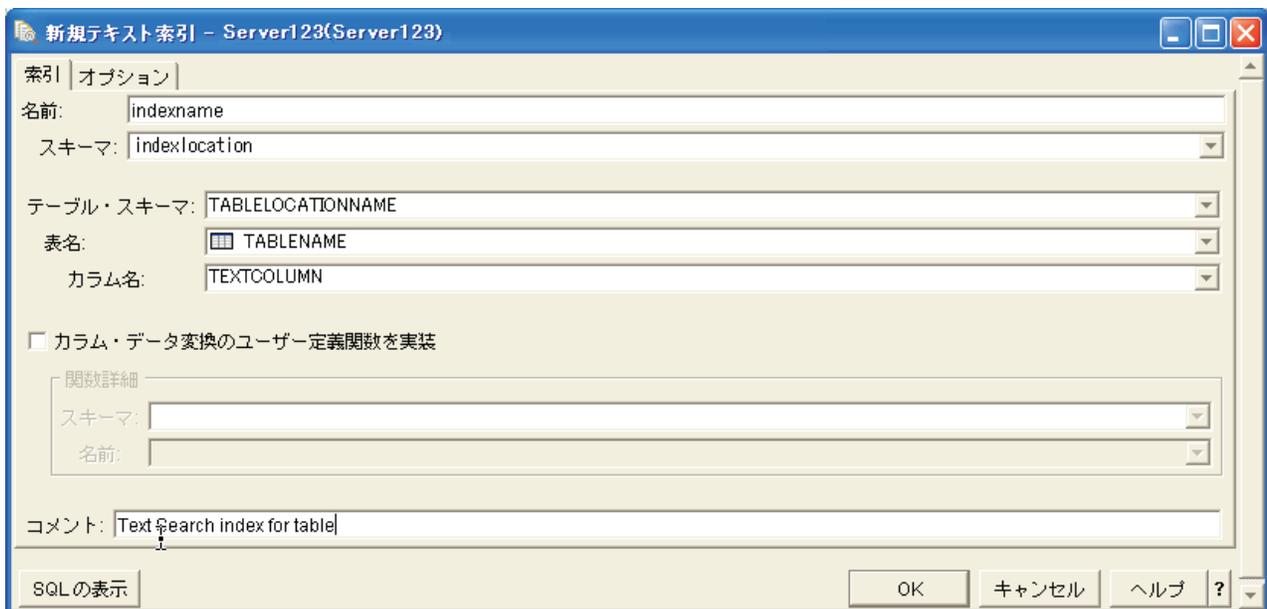
1. 右のペインで「テキスト・サーバー」フォルダーを選択します。
2. 右のペインで、システムで現在構成済みになっているテキスト検索サーバーの状況を表示します。
3. F5 を選択して、サーバーのリストを最新表示します。



OmniFind テキスト・サーバーの開始と停止:

1. 右のペインで対象のサーバー ID を選択します。
2. 右クリックによってオプションを表示します。
 - サーバーの開始
 - サーバーの停止
 - テキスト検索索引の作成 (Create a text search index)

テキスト検索索引の作成:

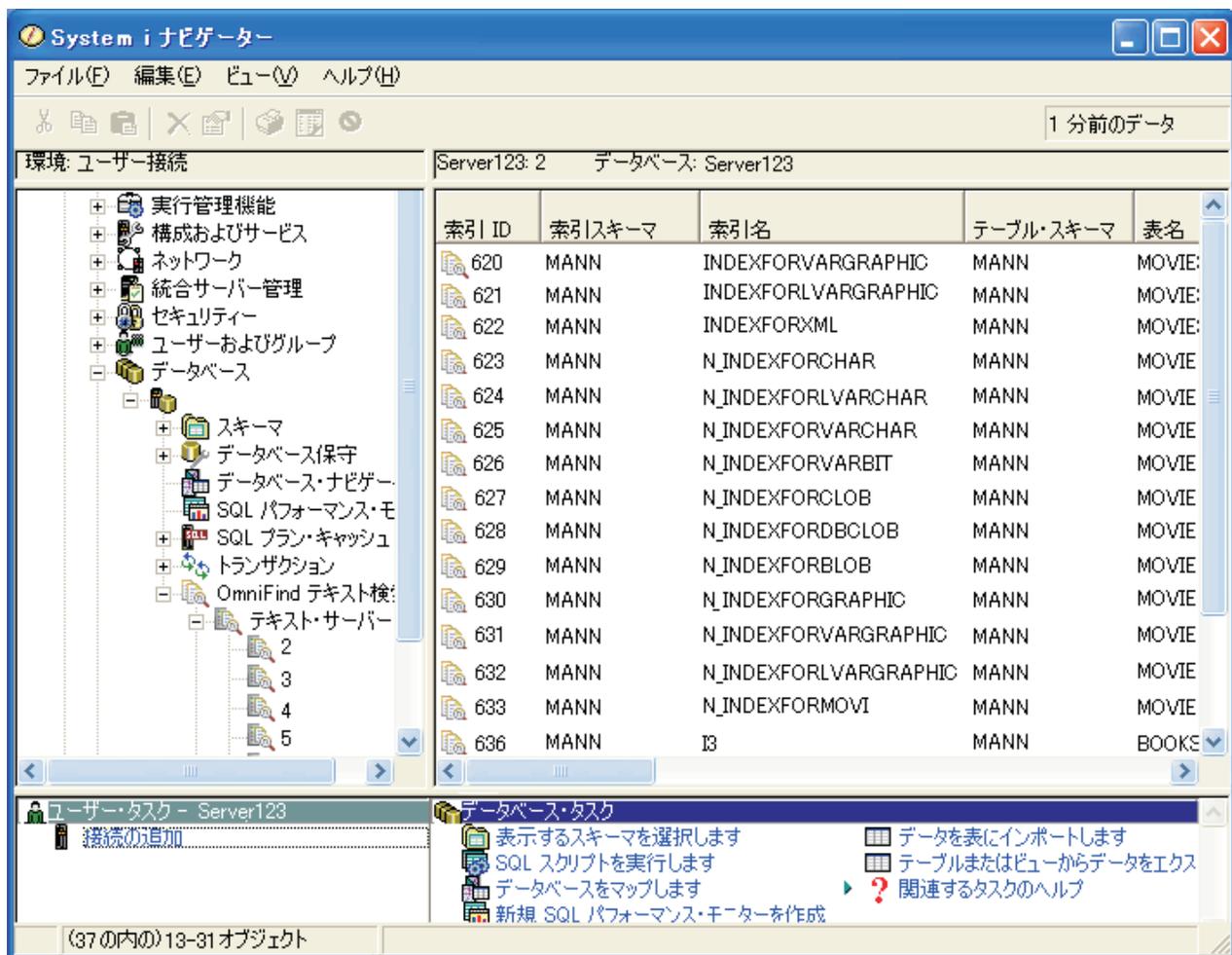


テキスト検索索引の操作

System i ナビゲーターを使用して、システムに存在するテキスト検索索引に対して操作を実行できます。

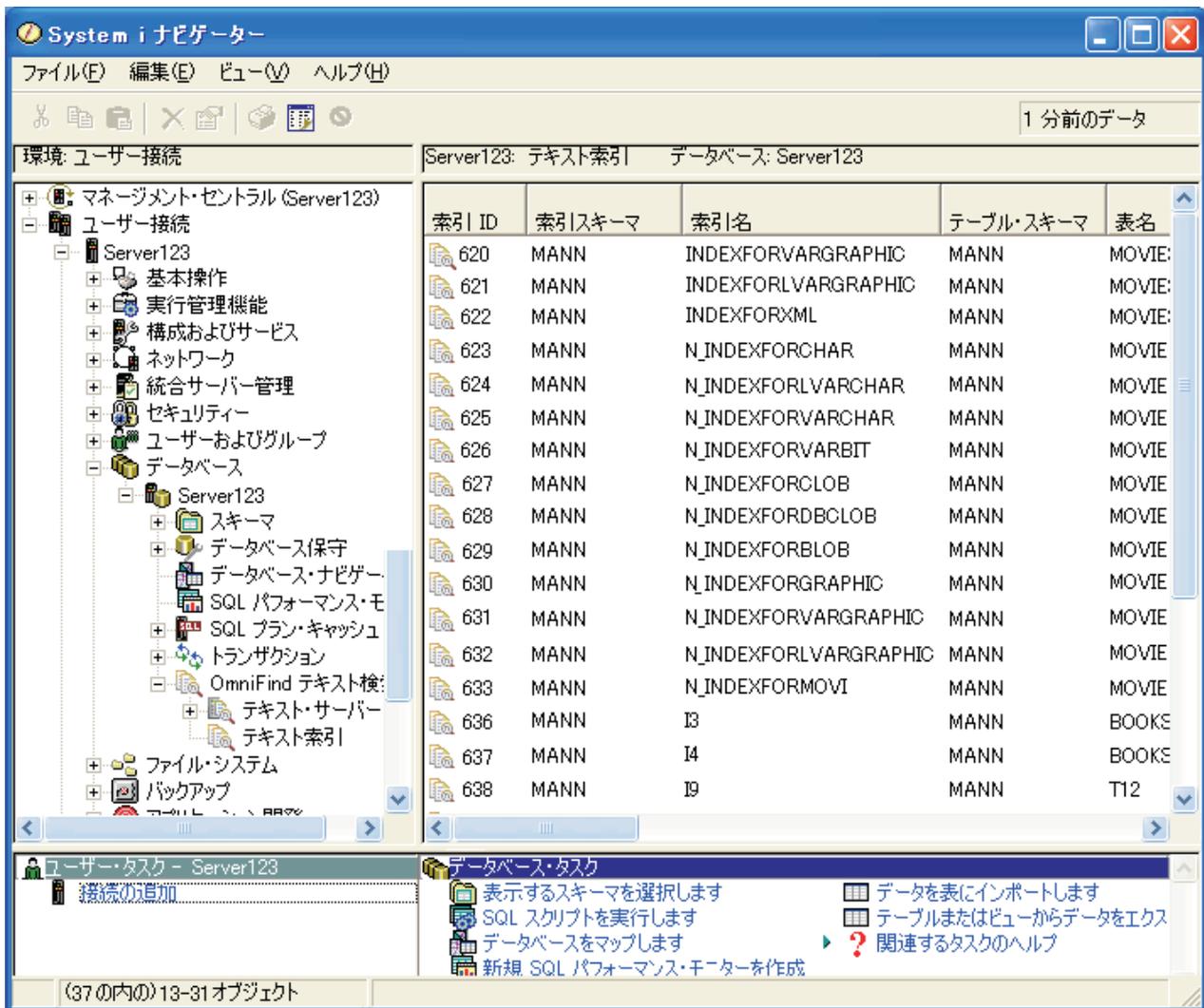
テキスト・サーバーの **OmniFind** テキスト索引の表示:

1. 「System i ナビゲーター」ウィンドウで、使用するシステムを展開します。
2. 「データベース」を展開します。
3. 処理対象のデータベースを展開します。
4. 「OmniFind テキスト検索」を選択します。
5. 「テキスト・サーバー」を選択します。
6. 処理対象のテキスト・サーバーを選択します。そのサーバーの索引が右のペインに表示されます。



このシステムの OmniFind テキスト索引の表示:

1. 「OmniFind テキスト検索」を選択します。
2. 「テキスト索引」を選択します。このシステムの現行区画にあるすべてのテキスト検索索引が表示されます。



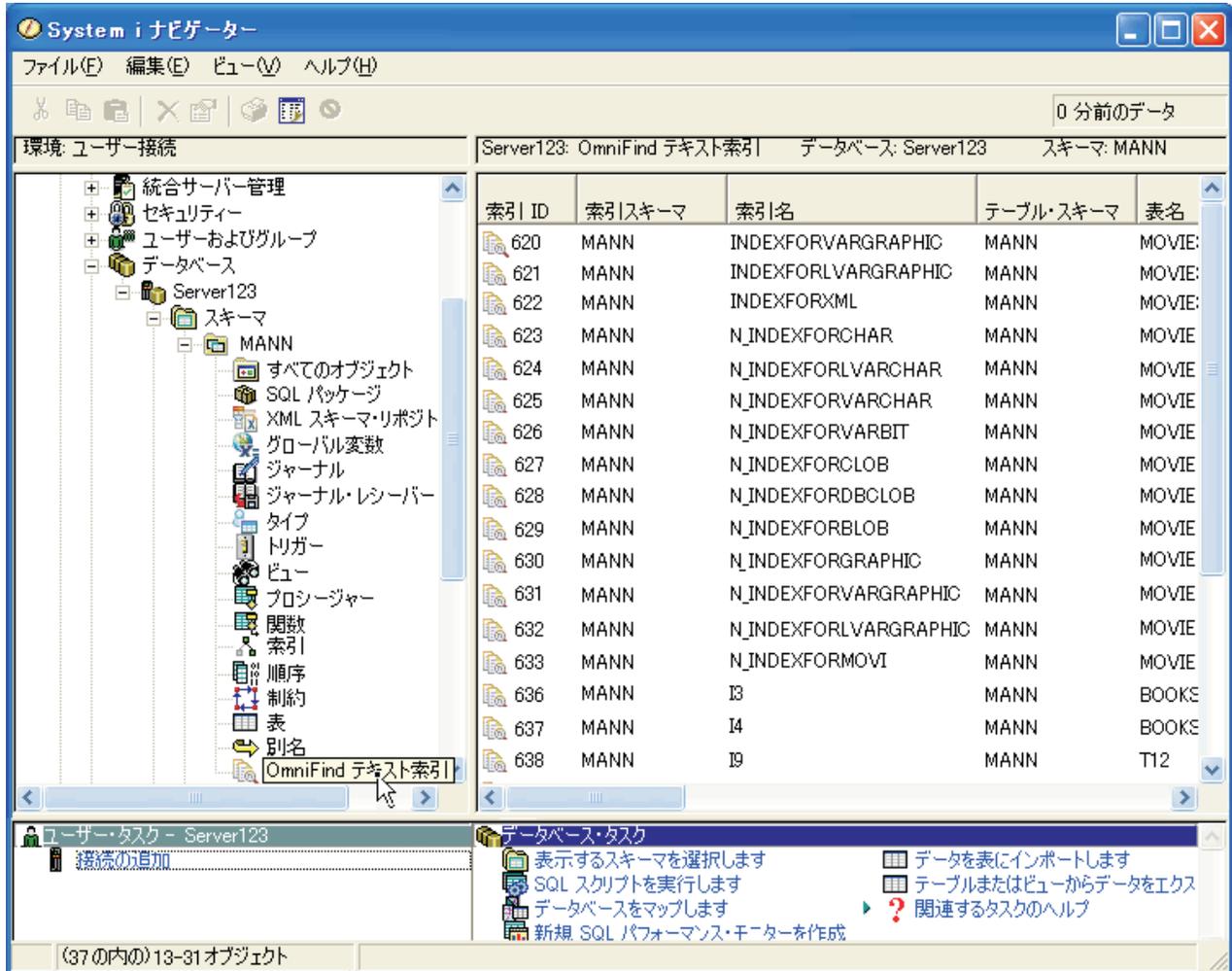
テキスト索引に対する操作の実行:

1. 右のペインで処理対象の索引を選択します。
2. 索引を右クリックします。
3. 操作を選択します。
 - 更新
 - 再作成 (reprime)
 - 削除
 - 記述の表示
 - 定義の表示
 - 定義の変更 (alter definition)

スキーマの OmniFind テキスト索引の表示:

1. 「System i ナビゲーター」ウィンドウで、使用するシステムを展開します。
2. 「データベース」を展開します。
3. 処理対象のデータベースを展開します。

4. 「スキーマ」を展開します。
5. 処理対象のスキーマを展開します。
6. 「OmniFind テキスト索引」を選択します。このスキーマのすべてのテキスト検索索引が表示されます。

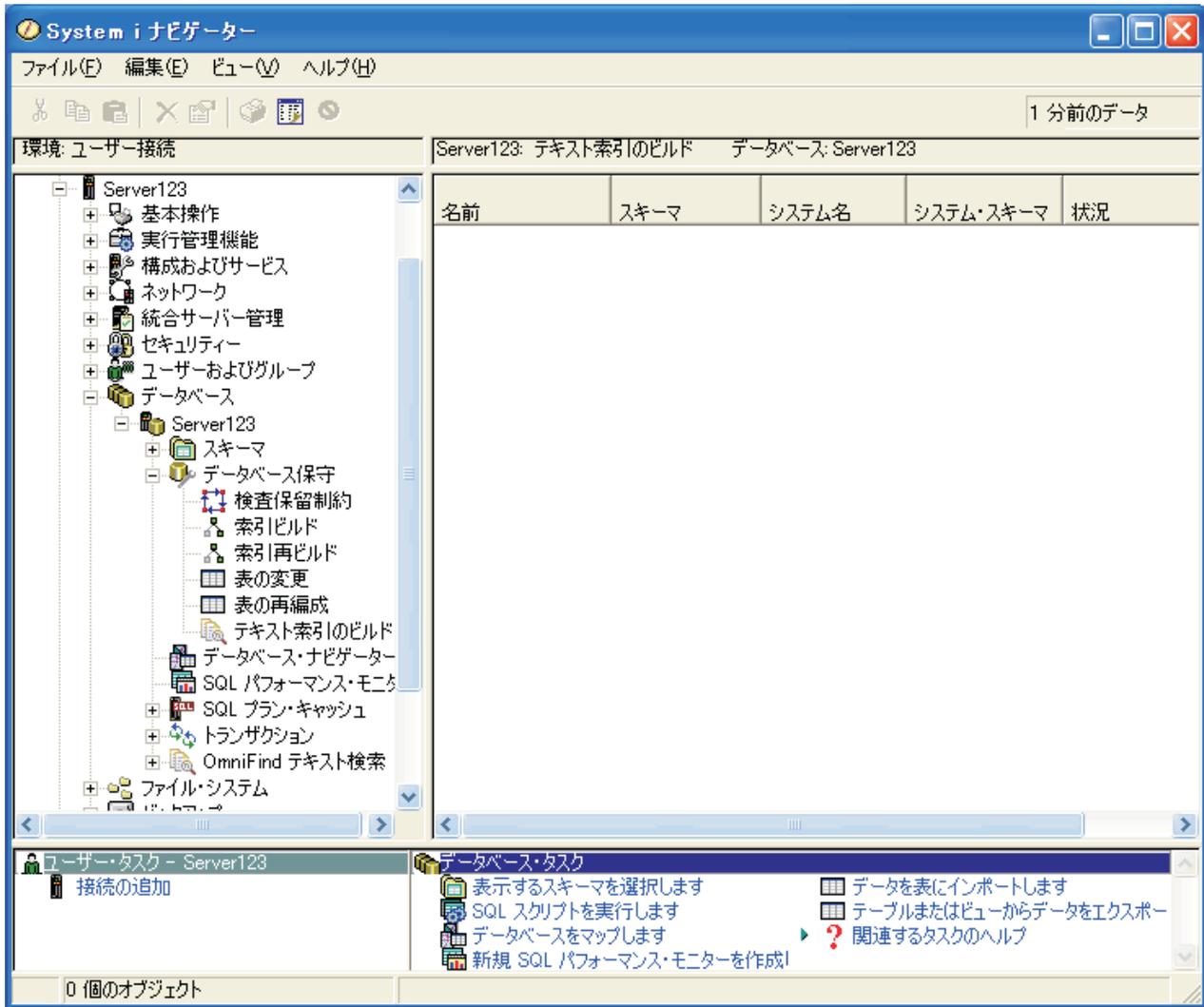


テキスト検索索引の作成状況の表示:

System i ナビゲーターを使用して、データベースによってテキスト索引が作成されている状況を表示できます。このビューは、アプリケーションでテキスト検索索引を使用できるようになる時点を確認するために役立ちます。

- 作成中のテキスト検索索引を表示するには、以下の手順を実行します。
1. 「System i ナビゲーター」ウィンドウで、使用するシステムを展開します。
 2. 「データベース」を展開します。
 3. 処理対象のデータベースを展開します。
 4. 「データベース保守」を展開します。
 5. 「テキスト索引のビルド」を選択します。

- このパネルには、進行中のテキスト検索索引の更新処理だけが表示されます。システムで現在実行中の更新処理や再作成処理がない場合は、空になります。



テキスト検索の管理表

- QSYS2 の管理表を使用して、テキスト検索サーバーとテキスト検索索引をサポートできます。

QSYS2.SYSTEXTDEFAULTS 管理表

- QSYS2.SYSTEXTDEFAULTS 管理表では、デフォルト・パラメーターとその値を確認できます。この表は、DB2 for i OmniFind テキスト検索サーバー のインストール時に作成されます。

- QSYS2.SYSTEXTDEFAULTS 管理表の内容を以下の表にまとめます。

表 31. QSYS2.SYSTEXTDEFAULTS 管理表の内容

列名	データ・タイプ	ヌル可能?	説明
NAME	VARCHAR(30)	いいえ	テキスト検索のデータベースのデフォルト・パラメーターの名前。

表 31. QSYS2.SYSTEXTDEFAULTS 管理表の内容 (続き)

列名	データ・タイプ	ヌル可能?	説明
VALUE	VARCHAR(512)	いいえ	テキスト検索のデフォルト・パラメーターの値。
TYPE	INTEGER	いいえ	予約済み。

QSYS2.SYSTEXTINDEXES 管理表

QSYS2.SYSTEXTINDEXES 管理表では、各テキスト検索索引の情報を確認できます。それぞれのテキスト検索索引ごとに、名前、スキーマ名、テキスト検索サーバーにある関連コレクション名が記述されています。

QSYS2.SYSTEXTINDEXES 管理表の内容を以下の表にまとめます。この表のユニーク・キーは、INDEXSCHEMA 列と INDEXNAME 列です。主キーは INDEXID 列です。

表 32. QSYS2.SYSTEXTINDEXES 管理表の内容

列名	データ・タイプ	ヌル可能?	説明
INDEXID	INTEGER	いいえ	テキスト検索索引に対して一意的に生成された索引 ID。
INDEXSCHEMA	VARCHAR(128)	いいえ	テキスト検索索引のスキーマ名。
INDEXNAME	VARCHAR(128)	いいえ	テキスト検索索引の非修飾名。
TABLESCHEMA	VARCHAR(128)	いいえ	基本表のスキーマ名。
TABLENAME	VARCHAR(128)	いいえ	基本表の非修飾名。
TABLEIASP	SMALLINT	いいえ	基本表の独立 ASP。
COLLECTIONNAME	VARCHAR(255)	いいえ	テキスト検索サーバーにある関連コレクションの名前。
SERVERID	INTEGER	いいえ	テキスト検索索引のサーバーの ID。
TAKEOVERSERVERID	INTEGER	はい	将来使用するために予約済み。
TAKEOVERSERVERPULSE	TIMESTAMP	はい	将来使用するために予約済み。
SEARCHARGS	VARBINARY(1024)	はい	将来使用するために予約済み。
ALIASSCHEMA	VARCHAR(128)	いいえ	SYSPROC.SYSTS_CREATE ストアード・プロシージャでを使用した基本表のスキーマの別名。別名を使用しなかった場合は、TABLESCHEMA と同じ値になります。
ALIASNAME	VARCHAR(128)	いいえ	SYSPROC.SYSTS_CREATE ストアード・プロシージャでを使用した基本表の名前の別名。別名を使用しなかった場合は、TABLENAME と同じ値になります。
STAGINGTABLENAME	VARCHAR(128)	はい	テキスト検索索引のログ表の名前。
EVENTTABLENAME	VARCHAR(128)	いいえ	テキスト検索索引のイベント表の名前。
OFINDEXTABLENAME	VARCHAR(128)	いいえ	DB2 for i OmniFind テキスト検索サーバーにあるテキスト検索索引の表の名前。

表 32. QSYS2.SYSTEXTINDEXES 管理表の内容 (続き)

列名	データ・タイプ	ヌル可能?	説明
UPDATEMINIMUM	INTEGER	いいえ	テキスト検索索引のインクリメンタル更新の実行前にログ表に存在していなければならない項目の最小数。
UPDATEFREQUENCY	VARCHAR(512)	いいえ	SYSPROC.SYSTS_CREATE ストアード・プロシージャでテキスト検索索引を作成したときに指定した更新頻度。
UPDATEMODE	INTEGER	いいえ	テキスト検索索引の更新モードの指標。整数 0 (ゼロ) は、テキスト検索索引の初期更新です。値 1 は、その後のインクリメンタル更新です。
REORGANIZATIONMODE	INTEGER	いいえ	テキスト検索索引の再編成モードの指標。
CREATETIME	TIMESTAMP	いいえ	テキスト検索索引の作成時刻。
LASTUPDATETIME	TIMESTAMP	はい	テキスト検索索引の最終更新時刻。
LASTUPDATESTATUS	CHAR	はい	<p>テキスト検索索引の初期更新またはインクリメンタル更新の後に実行するクリーンアップ・プロセスの最適化のための内部状況の指標。標準的な値は、以下のとおりです。</p> <ul style="list-style-type: none"> • (ヌル) は、索引がまだ更新されていない、という意味です。 • 「C」は、初期更新が開始された、という意味です。更新モードがインクリメンタルでない場合は、初期更新が進行中であるか、完了していません。 • 「N」は、インクリメンタル更新が正常に完了した、という意味です。 <p>その他のコードは、更新処理中に内部で使用されます。更新処理では、それらのコードに基づいて、更新を完了できなかった場合に実行できるリカバリー・アクションを判別します。</p>
SCHEDULERTASKID	INTEGER	はい	将来使用するために予約済み。
EXPRESSIONLISTS	CLOB (32 K)	はい	将来使用するために予約済み。
EXPRESSIONNUMBERS	VARBINARY(32)	はい	将来使用するために予約済み。
USEREXITFUNCTION	VARCHAR(18)	はい	将来使用するために予約済み。
REMARKS	VARCHAR(2000)	はい	<p>SYSPROC.SYSTS_CREATE ストアード・プロシージャの index-configuration-options パラメータの COMMENTS オプションで記述した注釈。</p>

表 32. QSYS2.SYSTEXTINDEXES 管理表の内容 (続き)

列名	データ・タイプ	ヌル可能?	説明
TABLEMBR	VARCHAR(10)	はい	テキスト索引を作成した表メンバー。マルチメンバー・ファイルの場合に、索引生成の対象になっている特定のメンバーを追跡管理するための値です。値がヌルの場合は、メンバーが表の最初の (唯一の) メンバーである、という意味です。

関連概念

12 ページの『サーバーの別名』

サーバーの別名を使用して、意味のある名前をサーバーに割り当てることができます。

QSYS2.SYSTEXTCOLUMNS 管理表

QSYS2.SYSTEXTCOLUMNS 管理表では、テキスト検索索引に対応するテキスト列の情報を確認できます。それぞれのテキスト検索索引ごとに、索引 ID、テキスト列の名前、基本表のスキーマ名が記述されています。

QSYS2.SYSTEXTCOLUMNS 管理表の内容を以下の表にまとめます。この表の主キーは、INDEXID 列と COLUMNNAME 列です。外部キーは INDEXID 列です。

表 33. QSYS2.SYSTEXTCOLUMNS 管理表の内容

列名	データ・タイプ	ヌル可能?	説明
INDEXID	INTEGER	いいえ	テキスト検索索引に対して一意的に生成された索引 ID。
COLUMNNAME	VARCHAR(128)	いいえ	テキスト列の非修飾名。
TABLESCHEMA	VARCHAR(128)	いいえ	基本表のスキーマ名。
TABLERNAME	VARCHAR(128)	いいえ	基本表の非修飾名。
LANGUAGE	VARCHAR(5)	いいえ	テキスト検索サーバーがテキスト文書の言語処理で使用する言語。デフォルト値は en_US (英語) です。
FUNCTIONSCHEMA	VARCHAR(128)	はい	サポート対象でないデータ・タイプの列に存在するテキスト文書や他の場所に格納されているテキスト文書にアクセスするために OMNIFIND が使用するユーザー定義関数のスキーマ。
FUNCTIONNAME	VARCHAR(18)	はい	サポート対象でないデータ・タイプの列に存在するテキスト文書や他の場所に格納されているテキスト文書にアクセスするために OMNIFIND が使用するユーザー定義関数の名前。
CCSID	INTEGER	いいえ	バイナリー・データ・タイプの列のテキスト検索索引で使用するコード化文字セット ID。
FORMAT	VARCHAR(30)	いいえ	列に存在するテキスト文書の形式。サポート対象の形式の値は、TEXT、HTML、XML、INSO です。

表 33. QSYS2.SYSTEXTCOLUMNS 管理表の内容 (続き)

列名	データ・タイプ	ヌル可能?	説明
KEYCOLUMNCOUNT	INTEGER	いいえ	テキスト検索索引のキー列の数。
KEYCOLUMNNAMES	VARCHAR(1200)	いいえ	テキスト検索索引のキー列の名前。

QSYS2.SYSTEXTSERVERS 管理表

QSYS2.SYSTEXTSERVERS 管理表では、テキスト検索サーバーのインストール先に関する情報を確認できます。

QSYS2.SYSTEXTSERVERS 管理表の内容を以下の表にまとめます。この表のユニーク・キーは、SERVERNAME 列と SERVERPORT 列です。主キーは SERVERID 列です。

表 34. QSYS2.SYSTEXTSERVERS 管理表の内容

列名	データ・タイプ	ヌル可能?	説明
SERVERID	INTEGER	いいえ	テキスト検索サーバーに対して一意的に生成された ID。
SERVERNAME	VARCHAR(128)	いいえ	テキスト検索サーバーのホスト名または IP アドレス。
SERVERADRINFO	VARBINARY(3000)	はい	SYSPROC.SYSTS_START ストアード・プロシージャで指定した SERVERNAME と SERVERPORT の内部表記。
SERVERPORT	INTEGER	いいえ	テキスト検索サーバーのポート番号。
SERVERPATH	VARCHAR(512)	いいえ	テキスト検索サーバーのサーバー・パス。
SERVERTYPE	INTEGER	いいえ	テキスト検索サーバーのサーバー・タイプ。値 0 (ゼロ) は、IBM i のテキスト検索サーバーです。値 1 は、Linux のテキスト検索サーバーです。値 2 は、Windows のテキスト検索サーバーです。
SERVERAUTHTOKEN	VARCHAR(256)	いいえ	テキスト検索サーバーの認証トークン。
SERVERMASTERKEY	VARCHAR(36)	いいえ	テキスト検索サーバーのサーバー・キー。
SERVERCLASS	INTEGER	いいえ	テキスト検索サーバーのサーバー・クラス。値 0 (ゼロ) は、実動サーバーであり、自動選択が可能です。値 9 は、テスト・サーバーであり、自動割り振りはできません。
SERVERSTATUS	INTEGER	いいえ	このサーバーをテキスト検索サーバーとして使用して、新しいテキスト検索索引を作成できるかどうかを示す指標。デフォルト値は 0 (ゼロ) であり、このサーバーを使用できる、という意味です。
ALIASNAME	VARCHAR(128)	はい	別名は、ヌル以外であれば固有の値であり、大/小文字の区別があります。

関連概念

- 12 ページの『サーバーの別名』
- サーバーの別名を使用して、意味のある名前をサーバーに割り当てることができます。

QSYS2.SYSTEXTCONFIGURATION 管理表

- QSYS2.SYSTEXTCONFIGURATION 管理表では、SYSPROC.SYSTS_CREATE ストアード・プロシージャで渡したテキスト検索索引の構成パラメーターを確認できます。

- QSYS2.SYSTEXTCONFIGURATION 管理表の内容を以下の表にまとめます。主キーは、INDEXID 列と PARAMETER 列です。外部キーは INDEXID 列です。

表 35. QSYS2.SYSTEXTCONFIGURATION 管理表の内容

列名	データ・タイプ	ヌル可能?	説明
INDEXID	INTEGER	いいえ	テキスト検索索引に対して一意的に生成された索引 ID。
PARAMETER	VARCHAR(30)	いいえ	SYSPROC.SYSTS_CREATE ストアード・プロシージャでテキスト検索索引を作成するときに指定したパラメーター。
VALUE	VARCHAR(512)	いいえ	指定したパラメーターの値。

関連資料

- 21 ページの『SYSPROC.SYSTS_CREATE』
- SYSPROC.SYSTS_CREATE ストアード・プロシージャを呼び出すことによって、テキスト検索索引を作成できます。このストアード・プロシージャでは、テキスト列に基づいてテキスト検索索引を作成します。そのテキスト検索索引は、CONTAINS 関数または SCORE 関数を組み込んだ SQL 照会で使用できます。

QSYS2.SYSTEXTSERVERHISTORY 管理表

- QSYS2.SYSTEXTSERVERHISTORY 補助表を表示することによって、SYSPROC.SYSTS_DROP ストアード・プロシージャで使用したサーバーの履歴情報を確認できます。

- QSYS2.SYSTEXTSERVERHISTORY 管理表の内容を以下の表にまとめます。この表のユニーク・キーは、INDEXID 列と SERVERID 列です。外部キーは INDEXID 列です。

表 36. QSYS2.SYSTEXTSERVERHISTORY 管理表の内容

列名	データ・タイプ	ヌル可能?	説明
INDEXID	INTEGER	いいえ	作成したテキスト検索索引の索引 ID。
SERVERID	INTEGER	いいえ	SYSPROC.SYSTS_DROP でテキスト検索索引を削除しなければならなかったサーバーの ID。

関連資料

- 35 ページの『SYSPROC.SYSTS_DROP』
- SYSPROC.SYSTS_DROP ストアド・プロシージャを呼び出すことによって、
- SYSPROC.SYSTS_CREATE ストアド・プロシージャで定義したテキスト検索索引を削除できます。

メッセージとコード

- DB2 for i OmniFind テキスト検索サーバーのメッセージと SQL 戻りコードを確認できます。メッセージのリストは、数字の順序になっています。

OmniFind のメッセージ

- DB2 for i OmniFind テキスト検索サーバーの OmniFind メッセージを表示できます。メッセージのリストは、数字の順序になっています。

- 以下のエラーが発生すると、メッセージが OmniFind メッセージ・ファイル (QDBTSLIB/QOMFMSGF) に追加されます。

表 37. OmniFind のメッセージ

番号	タイプ	メッセージ
OMF0011	通知	テキスト検索索引が別の構成オプションで復元されました。
OMF0012	警告	作成する索引の FORMAT タイプは、XML ではありません。XML 検索はサポートされていません。
OMF0334	エラー	指定したオブジェクト "{0}."{1}" はサポートされていません。テキスト検索索引を作成できるのは、SQL 表または SQL 別名、あるいは物理ファイルの 1 つのメンバーに限られています。
OMF0358	エラー	現在のユーザーには、要求された操作を実行するための十分な権限がありません。
OMF0359	エラー	復元が失敗しました。テキスト索引のテキスト・サーバーが使用不能な状態になっています。

SQLCODE -0196 &2 の &1 のカラム &3 を除去することはできない。

- 説明: 列 &3 を削除しようとした。ビュー、制約、トリガー、索引のいずれかがその列に依存しているか、パーティション・キーにその列が含まれているので、その列を削除することはできません。

- ユーザーの処置: ALTER TABLE ステートメントで CASCADE を指定して、その列と一緒に、その列に依存するビュー、制約、トリガー、索引も削除してください。パーティション・キーにその列が含まれている場合は、ALTER TABLE ステートメントで DROP PARTITIONING を指定して、表のパーティショニングを解除してください。要求を再試行してください。

SQLSTATE: 42817

SQLCODE -5003 コミットメント制御の下ではこの操作を実行することはできない。

- 説明: COMMIT(*CHG)、COMMIT(*CS)、COMMIT(*ALL) のいずれかを指定したコミットメント制御の下で以下の操作を実行することはできません。

- DROP SCHEMA ステートメント。
- 権限所有者が存在するオブジェクトに対する GRANT ステートメントまたは REVOKE ステートメント。
- 権限所有者が存在するオブジェクトに対する SQL 命名モードでの CREATE ステートメント。
- テキスト検索索引の DROP。

- これらの操作をコミット/ロールバックすることはできません。

- ユーザーの処置: COMMIT(*NONE) を指定して、ステートメントの実行をやり直してください。

SQLSTATE: 42922

SQLCODE -20423 テキスト検索処理中にエラーが発生しました。

説明: CONTAINS 関数または SCORE 関数のテキスト検索の処理中にエラーが発生しました。テキスト検索索引 *index-name* の使用中にサーバー *server* でエラーが発生しました (理由コード: *reason-code*)。この問題の説明テキスト: *text*。

server: エラーが発生したテキスト検索サーバーのホスト名または IP アドレスとポート。

index-name: テキスト検索の処理で使用していた索引の名前。

注: 1 つのトークンにスキーマとピリオドと索引名を組み込みます。

reason-code: DB2 for i OmniFind テキスト検索サーバーから返された理由コード。

text: DB2 for i OmniFind テキスト検索サーバーから返されたテキスト。

システムの処置: ステートメントは処理できません。

ユーザーの処置: システム管理者に連絡して、DB2 for i OmniFind テキスト検索サーバーが正常にインストールされているかどうかを確認してください。

SQLSTATE: 38H10

関連資料

40 ページの『CONTAINS』

CONTAINS 関数では、検索索引として指定する基準を使用して、テキスト検索索引を検索できます。この関数は、一致項目が見つかったかどうかを示す結果を返します。

43 ページの『SCORE』

SCORE 関数では、検索索引として指定する基準を使用して、テキスト検索索引を検索できます。この関数は、文書が照会内容に一致する比率を測定した関連度スコアを返します。

SQLCODE -20424 テキスト検索サポートが理由 *reason-code* のため使用できません。

説明: テキスト検索の管理表の 1 つで問題が検出されました。理由コードは *reason-code* です。

1 テキスト検索の管理表 (QSYS2.SYSTEXTINDEXES、QSYS2.SYSTEXTCOLUMNS、QSYS2.SYSTEXTSERVERS) のいずれかが検出されませんでした。

3 テキスト検索サポートがインストールされていません。

4 テキスト検索サポートが開始されていません。

QSYS2.SYSTEXTSERVERS 表の STATUS 列の値が 1 になっています。この値は、テキスト検索のサポートが停止されたという意味です。

7 DB2 for i OmniFind テキスト検索サーバーが定義されていません。

システムの処置: ステートメントは処理できません。

ユーザーの処置: システム管理者に連絡して、テキスト検索のサポートがシステムで正常にセットアップされていることを確認してください。

SQLSTATE: 38H11

関連資料

110 ページの『QSYS2.SYSTEXTSERVERS 管理表』
QSYS2.SYSTEXTSERVERS 管理表では、テキスト検索サーバーのインストール先に関する情報を確認できません。

107 ページの『QSYS2.SYSTEXTINDEXES 管理表』
QSYS2.SYSTEXTINDEXES 管理表では、各テキスト検索索引の情報を確認できます。それぞれのテキスト検索索引ごとに、名前、スキーマ名、テキスト検索サーバーにある関連コレクション名が記述されています。

109 ページの『QSYS2.SYSTEXTCOLUMNS 管理表』
QSYS2.SYSTEXTCOLUMNS 管理表では、テキスト検索索引に対応するテキスト列の情報を確認できます。それぞれのテキスト検索索引ごとに、索引 ID、テキスト列の名前、基本表のスキーマ名が記述されています。

SQLCODE -20425 カラム *column-name* ではテキスト検索ができません。

説明: テキスト検索関数の CONTAINS または SCORE で、*table-schema* の表 *table-name* の列 *column-name* が指定されています。その列にはテキスト索引が存在しないので、テキスト検索の処理を実行できません。

システムの処置: ステートメントは処理できません。

ユーザーの処置: その列とその表が DB2 for i OmniFind テキスト検索サーバーに登録されていることを確認してください。

SQLSTATE: 38H12

| 関連資料

| 40 ページの『CONTAINS』

| CONTAINS 関数では、検索引数として指定する基準を使用して、テキスト検索索引を検索できます。この関数は、一致項目が見つかったかどうかを示す結果を返します。

| 43 ページの『SCORE』

| SCORE 関数では、検索引数として指定する基準を使用して、テキスト検索索引を検索できます。この関数は、文書が照会内容に一致する比率を測定した関連度スコアを返します。

| **SQLCODE -20426 競合するテキスト検索管理プロシージャが既に実行中です。**| **説明:** 競合するテキスト検索管理プロシージャ (更新など) がこの索引で既に実行されています。| **システムの処置:** ステートメントは処理できません。| **ユーザーの処置:** 現在実行中のストアード・プロシージャが完了してから、この管理ストアード・プロシージャを再び呼び出してください。| **SQLSTATE:** 38H13| **SQLCODE -20427 テキスト検索管理プロシージャ中にエラーが発生しました。**| **説明:** テキスト検索管理プロシージャの実行中にエラーが発生しました。理由コードは *reason-code* です。返されたテキストは、*text* です。そのエラー・テキストで問題を確認できます。| **システムの処置:** CALL ステートメントがこの SQLCODE で失敗しました。| **ユーザーの処置:** *error* で示されている問題を修正してから、管理ストアード・プロシージャを再び呼び出してください。| **SQLSTATE:** 38H14| **CPF32fa &1 のテキスト検索索引 &2 で操作が許可されませんでした。**| **説明:** テキスト検索索引でサポートされていない操作を実行しようとした。テキスト検索索引では、DB2 の従来の索引とビューでは可能な操作の一部を実行できません。

| 索引を削除しようとした場合は、コミットメント制御がアクティブになっていたためにその操作が失敗した可能性があります。

| **ユーザーの処置:** OmniFind Text Search Server for DB2 for i に組み込まれている管理 SQL ストアード・プロシ

| ュージャを使用して、テキスト検索の管理操作を実行してください。

| テキスト検索索引の詳細と、テキスト検索索引に該当する制約事項については、Information Center (<http://www.ibm.com/systems/i/infocenter/>) の資料を参照してください。| **CPF32fb &1 のテキスト検索索引 &2 で操作を完了できませんでした。**| **説明:** &1 のテキスト検索索引 &2 に対して操作を実行しようとした。要求された操作は、現在無効になっています (理由コードは &3)。それぞれの理由コードと意味を以下にまとめます。

1. 必要な製品がインストールされていません。
2. 要求されたテキスト検索サーバー &4 は、使用不能な状態になっているか、定義されていません。
3. 索引を復元しようとしたが、その索引は既に存在しています。保管されている索引に合わせて既存の索引を変更することはできません。
4. &6 の従属オブジェクト &5 (タイプ &7) は存在しません。
5. &6 のオブジェクト &5 (タイプ &7) は使用不能な状態になっています。
6. 現在のスレッドの補助記憶域プール (ASP) は、テキスト検索索引の ASP と一致しません。
7. ライブラリー &6 の表 &5 の列 &8 のテキスト検索索引は既に存在しています。

| **ユーザーの処置:** 問題点を修正してから、操作をやり直してください。| テキスト検索索引の詳細については、Information Center (<http://www.ibm.com/systems/i/infocenter/>) の資料を参照してください。| **CPI321E ライブラリー &2 のファイル &1 が据え置かれた。**| **説明:** この復元要求の処理中にライブラリー &2 のファイル &1 が据え置かれました (指定の据え置き ID は &4 です)。そのファイルは、理由コード &3 で据え置かれました。それぞれの理由コードを以下にまとめます。

1. &1 を復元のために作成しようとしたときに、ライブラリー &6 の基本ファイル &5 が存在しませんでした。
2. 基本ファイルが欠落しているという理由以外の理由でファイル &1 を作成できませんでした。前のメッセージでファイルの作成が失敗した理由を確認してください。

- | 3. ファイル &1 の 1 つ以上のメンバーを作成できませんでした。前のメッセージでメンバーの作成が失敗した理由を確認してください。
- | 4. このファイルはテキスト検索索引を記述したファイルですが、必要なライセンス・プログラム・オブジェクトがシステムに存在しません。
- | 5. このファイルはテキスト検索索引を記述したファイルですが、索引を再作成できませんでした。
- | **ユーザーの処置:** 理由コード 1 の場合: 欠落している基本ファイルを復元するか、据え置きオブジェクト復元 (RSTDFROBJ) コマンドを使用して、同じ据え置き ID (DFRID パラメーター) &4 を指定します。

- | 理由コード 2 と 3 の場合: 作成処理の失敗理由になっている問題点を修正してから、据え置きオブジェクト復元 (RSTDFROBJ) コマンドを使用して、同じ据え置き ID (DFRID パラメーター) &4 を指定します。
- | 理由コード 5 の場合: ジョブ・ログに含まれている前のメッセージを確認し、エラーを修正してから、据え置きオブジェクト復元 (RSTDFROBJ) コマンドを使用して、同じ据え置き ID (DFRID パラメーター) &4 を指定します。
- | **関連情報**
- | 据え置きオブジェクト復元 (RSTDFROBJ)

コードに関するライセンス情報および特記事項

- | IBM は、お客様に、すべてのプログラム・コードのサンプルを使用することができる非独占的な著作使用権を許諾します。お客様は、このサンプル・コードから、お客様独自の特別のニーズに合わせた類似のプログラムを作成することができます。

- | 強行法規で除外を禁止されている場合を除き、IBM、そのプログラム開発者、および供給者は「プログラム」および「プログラム」に対する技術的サポートがある場合にはその技術的サポートについて、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任を負わないものとしします。

- | いかなる場合においても、IBM および IBM のサプライヤーならびに IBM ビジネス・パートナーは、その予見の有無を問わず発生した以下のものについて賠償責任を負いません。

- | 1. データの喪失、または損傷。
- | 2. 直接損害、特別損害、付随的損害、間接損害、または経済上の結果的損害
- | 3. 逸失した利益、ビジネス上の収益、あるいは節約すべかりし費用

- | 国または地域によっては、法律の強行規定により、上記の責任の制限が適用されない場合があります。

付録. 特記事項

本書は米国 IBM が提供する製品およびサービスについて作成したものです。

本書に記載の製品、サービス、または機能が日本においては提供されていない場合があります。日本で利用可能な製品、サービス、および機能については、日本 IBM の営業担当員にお尋ねください。本書で IBM 製品、プログラム、またはサービスに言及していても、その IBM 製品、プログラム、またはサービスのみが使用可能であることを意味するものではありません。これらに代えて、IBM の知的所有権を侵害することのない、機能的に同等の製品、プログラム、またはサービスを使用することができます。ただし、IBM 以外の製品とプログラムの操作またはサービスの評価および検証は、お客様の責任で行っていただきます。

IBM は、本書に記載されている内容に関して特許権 (特許出願中のものを含む) を保有している場合があります。本書の提供は、お客様にこれらの特許権について実施権を許諾することを意味するものではありません。実施権についてのお問い合わせは、書面にて下記宛先にお送りください。

〒242-8502
神奈川県大和市下鶴間1623番14号
日本アイ・ビー・エム株式会社
法務・知的財産
知的財産権ライセンス渉外

以下の保証は、国または地域の法律に沿わない場合は、適用されません。IBM およびその直接または間接の子会社は、本書を特定物として現存するままの状態を提供し、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任を負わないものとします。国または地域によっては、法律の強行規定により、保証責任の制限が禁じられる場合、強行規定の制限を受けるものとします。

この情報には、技術的に不適切な記述や誤植を含む場合があります。本書は定期的に見直され、必要な変更は本書の次版に組み込まれます。IBM は予告なしに、随時、この文書に記載されている製品またはプログラムに対して、改良または変更を行うことがあります。

本書において IBM 以外の Web サイトに言及している場合がありますが、便宜のため記載しただけであり、決してそれらの Web サイトを推奨するものではありません。それらの Web サイトにある資料は、この IBM 製品の資料の一部ではありません。それらの Web サイトは、お客様の責任でご使用ください。

IBM は、お客様が提供するいかなる情報も、お客様に対してなら義務も負うことのない、自ら適切と信ずる方法で、使用もしくは配布することができるものとします。

本プログラムのライセンス保持者で、(i) 独自に作成したプログラムとその他のプログラム (本プログラムを含む) との間での情報交換、および (ii) 交換された情報の相互利用を可能にすることを目的として、本プログラムに関する情報を必要とする方は、下記に連絡してください。

IBM Corporation
Software Interoperability Coordinator, Department YBWA
3605 Highway 52 N
Rochester, MN 55901
U.S.A.

本プログラムに関する上記の情報は、適切な使用条件の下で使用することができますが、有償の場合もあります。

本書で説明されているライセンス・プログラムまたはその他のライセンス資料は、IBM 所定のプログラム契約の契約条項、IBM プログラムのご使用条件、IBM 機械コードのご使用条件、またはそれと同等の条項に基づいて、IBM より提供されます。

この文書に含まれるいかなるパフォーマンス・データも、管理環境下で決定されたものです。そのため、他の操作環境で得られた結果は、異なる可能性があります。一部の測定が、開発レベルのシステムで行われた可能性があります。その測定値が、一般に利用可能なシステムのものと同じである保証はありません。さらに、一部の測定値が、推定値である可能性があります。実際の結果は、異なる可能性があります。お客様は、お客様の特定の環境に適したデータを確かめる必要があります。

IBM 以外の製品に関する情報は、その製品の供給者、出版物、もしくはその他の公に利用可能なソースから入手したものです。IBM は、それらの製品のテストは行っておりません。したがって、他社製品に関する実行性、互換性、またはその他の要求については確認できません。IBM 以外の製品の性能に関する質問は、それらの製品の供給者をお願いします。

IBM の将来の方向または意向に関する記述については、予告なしに変更または撤回される場合があります、単に目標を示しているものです。

表示されている IBM の価格は IBM が小売り価格として提示しているもので、現行価格であり、通知なしに変更されるものです。卸価格は、異なる場合があります。

本書はプランニング目的としてのみ記述されています。記述内容は製品が使用可能になる前に変更になる場合があります。

本書には、日常の業務処理で用いられるデータや報告書の例が含まれています。より具体性を与えるために、それらの例には、個人、企業、ブランド、あるいは製品などの名前が含まれている場合があります。これらの名称はすべて架空のものであり、名称や住所が類似する企業が実在しているとしても、それは偶然にすぎません。

著作権使用許諾:

本書には、様々なオペレーティング・プラットフォームでのプログラミング手法を例示するサンプル・アプリケーション・プログラムがソース言語で掲載されています。お客様は、サンプル・プログラムが書かれているオペレーティング・プラットフォームのアプリケーション・プログラミング・インターフェースに準拠したアプリケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。このサンプル・プログラムは、あらゆる条件下における完全なテストを経ていません。従って IBM は、これらのサンプル・プログラムについて信頼性、利便性もしくは機能性があることをほめかしたり、保証することはできません。このサンプル・プログラムは特定物として現存するままの状態を提供されるものであり、いかなる保証も提供されません。IBM は、このサンプル・プログラムの使用から生ずるいかなる損害に対しても、責任を負いません。

それぞれの複製物、サンプル・プログラムのいかなる部分、またはすべての派生的創作物にも、次のように、著作権表示を入れていただく必要があります。

© (お客様の会社名) (西暦年). このコードの一部は、IBM Corp. のサンプル・プログラムから取られています。 © Copyright IBM Corp. _年を入れる_.

この情報をソフトコピーでご覧になっている場合は、写真やカラーの図表は表示されない場合があります。

プログラミング・インターフェース情報

この「DB2 for i OmniFind テキスト検索サーバー」の資料には、DB2 for i OmniFind テキスト検索サーバーのサービスを利用するためのプログラムを、ユーザーが作成できるようにするためのプログラミング・インターフェースが記述されています。

商標

IBM、IBM ロゴおよび ibm.com は、世界の多くの国で登録された International Business Machines Corp. の商標です。他の製品名およびサービス名等は、それぞれ IBM または各社の商標である場合があります。現時点での IBM の商標リストについては、www.ibm.com/legal/copytrade.shtml をご覧ください。

Adobe、Adobe ロゴ、PostScript、PostScript ロゴは、Adobe Systems Incorporated の米国およびその他の国における登録商標または商標です。

Java およびすべての Java 関連の商標およびロゴは Sun Microsystems, Inc. の米国およびその他の国における商標です。

Linux は、Linus Torvalds の米国およびその他の国における商標です。

Windows は、Microsoft Corporation の米国およびその他の国における商標です。

UNIX は The Open Group の米国およびその他の国における登録商標です。

他の会社名、製品名およびサービス名等はそれぞれ各社の商標です。

使用条件

これらの資料は、以下の条件に同意していただける場合に限りご使用いただけます。

個人使用: これらの資料は、すべての著作権表示その他の所有権表示をしていただくことを条件に、非商業的な個人による使用目的に限り複製することができます。ただし、IBM の明示的な承諾をえずに、これらの資料またはその一部について、二次的著作物を作成したり、配布（頒布、送信を含む）または表示（上映を含む）することはできません。

商業的使用: これらの資料は、すべての著作権表示その他の所有権表示をしていただくことを条件に、お客様の企業内に限り、複製、配布、および表示することができます。ただし、IBM の明示的な承諾をえずにこれらの資料の二次的著作物を作成したり、お客様の企業外で資料またはその一部を複製、配布、または表示することはできません。

ここで明示的に許可されているもの以外に、資料や資料内に含まれる情報、データ、ソフトウェア、またはその他の知的所有権に対するいかなる許可、ライセンス、または権利を明示的にも黙示的にも付与するものではありません。

資料の使用が IBM の利益を損なうと判断された場合や、上記の条件が適切に守られていないと判断された場合、IBM はいつでも自らの判断により、ここで与えた許可を撤回できるものとさせていただきます。

お客様がこの情報をダウンロード、輸出、または再輸出する際には、米国のすべての輸出入関連法規を含む、すべての関連法規を遵守するものとします。

IBM は、これらの資料の内容についていかなる保証もしません。これらの資料は、特定物として現存するままの状態を提供され、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任なしで提供されます。



Printed in Japan