非结构化数据分析

SAP Sybase IQ 16.0 SP03

文档 ID： DC00958-01-1603-01
最后修订日期： 2013 年 12 月

目录

非结构化数据分析简介

了解 SAP® Sybase® IQ 中的非结构化数据分析以及与大对象数据的兼容性和符合性。

读者

本指南的目标读者是需要有关在 SAP Sybase IQ 中处理非结构化数据的参考资料的用户。

了解与非结构化数据分析功能相关的可用语法、参数、函数、存储过程、索引和选项。使用该指南作为参考资料，以了解非结构化数据在数据库中的存储和检索。

非结构化数据分析选件

非结构化数据分析选项将 SAP Sybase IQ 的功能扩展到允许在数据库中存储、检索和全文本搜索二进制大对象 (BLOB) 和字符大对象 (CLOB)。

注意： 用户必须获得专门许可才能使用本产品文档中介绍的非结构化数据分析功能。

随着数据量的增大，在关系数据库中存储大对象 (LOB) 数据的需求也随之增长。LOB 数据可以是以下形式之一：

- 非结构化 – 数据库只用于存储和检索数据，或者
- 半结构化（如文本） – 数据库支持数据结构并会提供支持函数（例如，字符串函数）。

典型的 LOB 数据源包括图像、地图、文档（如 PDF 文件、文字处理文件和演示文稿）、音频、视频和 XML 文件。SAP Sybase IQ 可以管理包含千兆字节 (GB)、千吉字节 (TB)，甚至是千万亿字节 (PB) 数据的各个 LOB 对象。

通过允许关系数据和非结构化数据存在于同一位置，SAP Sybase IQ 能让组织使用同样的应用程序和同样的界面访问这两种类型的数据。SAP Sybase IQ 全文本搜索功能支持在处理非结构化和半结构化数据时使用文本档案应用程序（文本分析）。

全文本搜索

全文本搜索使用 **TEXT** 索引在数据库中搜索术语和短语，而不必扫描表行。

TEXT 索引存储带索引的列中各个术语的位置信息。文本配置对象控制 **TEXT** 索引在生成或刷新时其中存放的术语，以及全文本查询的解释方式。

使用 **TEXT** 索引查找包含某个术语或短语的行通常要比扫描表中的每一行快。

兼容性

SAP Sybase SQL Anywhere® (SA) 和 SAP Adaptive Server® Enterprise (ASE) 用于存储大文本和二进制对象。

SQL Anywhere 可在数据类型为 `LONG VARCHAR` 或 `LONG BINARY` 的列中存储大对象（最大长度为 2GB）。SQL Anywhere 对这些数据类型的支持符合 SQL/2003 标准。SQL Anywhere 不支持 **BYTE_LENGTH64**、**BYTE_SUBSTR64**、**BFILE**、**BIT_LENGTH**、**OCTET_LENGTH**、**CHAR_LENGTH64** 和 **SUBSTRING64** 函数。

Adaptive Server 可在数据类型为 `TEXT` 或 `IMAGE` 的列中分别存储大文本对象（最大长度为 2GB）和二进制大对象（最大长度为 2GB）。Adaptive Server 对这些数据类型的支持是 ANSI SQL Transact-SQL® 扩展。

代理表中的 `LONG BINARY` 列映射到 Microsoft SQL Server 表中的 `VARBINARY(max)` 列。

与标准的一致性

SAP Sybase IQ `LONG BINARY` 和 `LONG VARCHAR` 功能符合 ISO/ANSI SQL 标准的核心级别。

TEXT 索引和文本配置对象

了解如何处理 **TEXT** 索引和文本配置对象。

TEXT 索引存储带索引的列中各个术语的位置信息。**TEXT** 索引是使用文本配置对象中存储的设置创建的。文本配置对象控制 **TEXT** 索引数据的特性，例如要忽略的术语，以及要在索引中包含的术语的最小长度和最大长度。

TEXT 索引

在全文本搜索中，系统会搜索 **TEXT** 索引，而不是表行。

必须先在要搜索的列上创建 **TEXT** 索引，然后才能执行全文本搜索。**TEXT** 索引存储带索引的列中各个术语的位置信息。与必须扫描表中所有值的查询相比，使用 **TEXT** 索引的查询通常更快。

创建 **TEXT** 索引时，可以指定创建和刷新 **TEXT** 索引时要使用的文本配置对象。文本配置对象中包含影响索引生成方式的设置。如果不指定文本配置对象，数据库服务器就会使用缺省配置对象。

可以在以下类型的列上创建 **TEXT** 索引：CHAR、VARCHAR 和 LONG VARCHAR 以及 BINARY、VARBINARY 和 LONG BINARY。BINARY、VARBINARY 和 LONG BINARY 列要求 **TEXT** 索引使用具有外部前置过滤器库的文本配置。

WD 和 TEXT 索引的比较

WD 和 **TEXT** 索引在语法和功能方面的比较。

表 1. WD 和 TEXT 索引

功能	是否被 WD 索引支持？	是否被 TEXT 索引支持？
结合术语	是，采用如下形式： `tbl.col CON-TAINS('great','whi-te' ,'whale')`	是，采用如下形式： `CONTAINS(tbl.col,'great white whale')`
常规布尔表达式	是，采用如下形式： `tbl.col CONTAINS ('great') AND ( tbl.col CON-TAINS('white) OR tbl.col CONTAINS('whale') AND NOT tbl.col CONTAINS('ship'))`	是，采用如下形式： `CONTAINS(tbl.col, 'great AND ( white OR whale AND NOT ship )')`

功能	是否被 WD 索引支持？	是否被 TEXT 索引支持？
搜索前缀匹配的术语	否	是，例如： `CONTAINS (tbl.col,'whale*')`
对 **LIKE** 谓词进行加速	是，例如： `tbl.col LIKE 'whale%'`	是，条件是 TEXT 索引是不含外部文档前置过滤器的 NGRAM TEXT 索引。例如： `c LIKE '%apple%fruit'`

搜索邻近的术语	否	是，例如：
		`CONTAINS(tbl.col, 'white BEFORE whale')`
		`CONTAINS(tbl.col, 'whale NEAR white')`
		`CONTAINS(tbl.col, ' "white whale" ')`

根据搜索得分对结果进行排序	否	是

在 **TEXT** 索引中，搜索前缀匹配的术语和搜索 **LIKE** 表达式有不同的语义，并且可能会根据文本配置返回极为不同的结果。最小长度、最大长度和停止列表的规范将控制前缀处理，但不会影响 **LIKE** 语义。

注意： 当发生术语删除时，布尔表达式的含义在 **WD** 索引和 **TEXT** 索引之间将有所不同，因为 **TEXT** 索引处理中被删除术语的影响在 **WD** 索引中没有等效项。

使用 Interactive SQL 创建 TEXT 索引

必须先在要搜索的列上创建 **TEXT** 索引，然后才能执行全文本搜索。

TEXT 索引存储带索引的列中各个术语的位置信息。

1. 以具有 CREATE ANY INDEX 或 CREATE ANY OBJECT 系统特权的用户身份或以基础表的所有者身份连接到数据库。
2. 执行 **CREATE TEXT INDEX** 语句。

以下示例在 `iqdemo` 数据库中 `Customers` 表的 `CompanyName` 列上创建一个 **TEXT** 索引，`myTxtIdx`。该示例使用 `default_char` 文本配置对象。

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

TEXT 索引大小估计准则

公式估计 **TEXT** 索引的 main 存储大小。

*Number of bytes = (15+L)*U + U*PAGESIZE * R + T*

其中：

- L = 词汇表的平均术语长度
- U = 词汇表中的不重复术语数
- R = 文档数（以百万计）
- T = 所有文档中所有术语的总数

TEXT 索引需要的临时空间（以字节为单位）为 *(M+20)* T*，其中：

- M = 文本配置的最大术语长度（以字节为单位）

注意： 需要的临时空间受限于排序数据的可压缩性。

TEXT 索引限制

文本配置对象和 **TEXT** 索引在设计上有一些限制。

- SAP Sybase IQ 不支持跨多个列的 **TEXT** 索引。
- 不支持 **TEXT** 索引手动刷新或自动刷新选项。
- **sp_iqrebuildindex** 不能用于生成 **TEXT** 索引。
- 不能在 **BEGIN PARALLEL IQ…END PARALLEL IQ** 中创建 **TEXT** 索引。
- 由于 **NGRAM** 术语断开器是基于 **TEXT** 索引而生成的，因此，请使用文本配置对象设置来定义是使用 **NGRAM** 还是使用 **GENERIC TEXT** 索引。
- **NGRAM TEXT** 索引搜索主要适用于单词拼写错误的情况。SAP Sybase IQ 不支持同义词和反义词之类的搜索。

使用 Interactive SQL 显示 TEXT 索引列表

查看数据库中所有 **TEXT** 索引的列表。

1. 以具有以下任一系统特权的用户身份连接到数据库：
   - CREATE ANY INDEX
   - ALTER ANY INDEX
   - DROP ANY INDEX
   - CREATE ANY OBJECT
   - ALTER ANY OBJECT
   - DROP ANY OBJECT
   - MANAGE ANY DBSPACE
2. 执行 **SELECT** 语句。

列出所有 **TEXT** 索引：

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

列出所有 **TEXT** 索引（包括目录表上的索引）：

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

使用 Interactive SQL 编辑 TEXT 索引

更改 **TEXT** 索引的设置，包括数据库空间和 **TEXT** 索引名称。

1. 以具有 ALTER ANY INDEX 或 ALTER ANY OBJECT 系统特权的用户身份或以基础表的所有者身份连接到数据库。
2. 执行 **ALTER TEXT INDEX** 语句。

将 **TEXT** 索引 `myTxtIdx` 重命名为 `MyTextIndex`：

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

使用 Interactive SQL 修改 TEXT 索引位置

更改用于存储 **TEXT** 索引的数据库空间。

1. 以具有 MANAGE ANY DBSPACE 系统特权的用户身份连接到数据库。
2. 执行带有 **MOVE TO** 子句的 **ALTER TEXT INDEX** 语句。

将 **TEXT** 索引 `MyTextIndex` 移动到名为 `tispace` 的 dbspace：

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

使用 Interactive SQL 删除 TEXT 索引

从数据库中删除 **TEXT** 索引。

1. 以具有 DROP ANY INDEX 或 DROP ANY OBJECT 系统特权的用户身份或以基础表的所有者身份连接到数据库。
2. 执行 **DROP TEXT INDEX** 语句。

删除 `MyTextIndex` **TEXT** 索引：

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

TEXT 索引刷新

SAP Sybase IQ 表中的 **TEXT** 索引仅支持刷新类型“立即刷新”，此刷新会在基础表中数据变化时发生。

表中的立即刷新 **TEXT** 索引支持隔离级别 3。它们会在创建时以及每次使用 **INSERT**、**UPDATE** 或 **DELETE** 语句对列中数据进行更改时被填充。在初始刷新期间，表中持有独占锁。

TEXT_DELETE_METHOD 数据库选项

指定在 **TEXT** 索引中进行删除时所使用的算法。

允许值
0 – 2

0 - 开销模型选择删除方法。

1 - 强制小型删除方法。当要删除的行数在表的总行数中所占的百分比非常小时，小型删除方法非常有用。少量删除算法可以随机访问索引，导致带有大型数据集的高速缓存抖动。

2 - 强制大型删除方法。该算法将扫描整个索引搜索要删除的行。当要删除的行数在表的总行数中所占的百分比非常大时，大型删除方法非常有用。

缺省值
0

范围
必须具备 SET ANY PUBLIC OPTION 系统特权才能为 PUBLIC 或者其他用户或角色设置此选项。只能为 PUBLIC 角色临时设置。设置立即生效。

描述
TEXT_DELETE_METHOD 指定在 **TEXT** 索引中进行删除操作时所使用的算法。当该选项未设置或设置为 0 时，由开销模型选择删除方法。开销模型在选择相应的删除算法时会考虑 CPU 相关开销以及 I/O 相关开销。开销模型会考虑以下因素：

- 删除的行数
- 索引大小
- 索引数据类型的宽度
- 索引数据的基数
- 可用临时高速缓存
- 与计算机相关的 I/O 和 CPU 特性
- 可用 CPU 和线程

示例
在 **TEXT** 索引中强制大型删除方法：

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

NGRAM TEXT 索引

通过将文本拆分成文本值为 N 的 n-gram（其中 N 是用户给定的值），NGRAM TEXT 索引存储列中的文本。

将查询的 **CONTAINS** 子句中文本值的 n-gram 与索引中存储的 n-gram 进行匹配，可以通过 **NGRAMTEXT** 索引执行搜索。

NGRAM TEXT 索引提供了针对欧洲语言文本和非欧洲语言文本的模糊搜索功能。

注意： **NGRAM TEXT** 索引搜索主要适用于单词拼写错误的情况。SAP Sybase IQ 不支持同义词和反义词之类的搜索。

由于 **NGRAM** 分词符是基于 **TEXT** 索引而建立的，因此，请使用文本配置对象设置来定义是使用 **NGRAM** 还是使用 **GENERICTEXT** 索引。

文本配置对象

文本配置对象控制着在生成或刷新 **TEXT** 索引时放置其中的术语以及如何对全文本查询进行解释。

数据库服务器创建或刷新 **TEXT** 索引时，将使用创建 **TEXT** 索引时指定的文本配置对象的设置。如果未指定文本配置对象，数据库服务器将根据要建立索引的列的数据类型来选择其中一个缺省文本配置对象。在 SAP Sybase IQ 数据库中，始终使用 `default_char` 文本配置对象。

文本配置对象指定使用哪个预过滤器库和分词符从要编制索引的文本中生成术语。它们指定要在 **TEXT** 索引中存储的术语的最小和最大长度，以及不应包含的术语的列表。文本配置对象由以下参数组成：

- 文档前置过滤器 - 删除文档中的不必要信息，如格式和图像信息。然后，被过滤的文档将由其它模块获取以进行进一步的处理。文档前置过滤器由第三方供应商提供。
- 文档术语断开器 - 将传入字节流分解成由术语分隔符分隔的术语，或者根据指定的规则进行分解。文档术语断开器由服务器或第三方供应商提供。
- 非索引字表处理器 - 指定在生成 **TEXT** 索引时将忽略的术语列表。

缺省文本配置对象

SAP Sybase IQ 提供缺省文本配置对象。

缺省文本配置对象 `default_char` 与非 `NCHAR` 数据一起使用。此配置是在您首次创建文本配置对象或 **TEXT** 索引时创建的。

系统支持文本配置对象 default_nchar 用于 **IN SYSTEM** 表中 **TEXT** 索引的 NCHAR；不能将 default_nchar 文本配置用于表中的 **TEXT** 索引。

“缺省文本配置设置”表显示了 default_char 和 default_nchar 的缺省设置，对于大多数基于字符的语言，这是最适合的设置。我们强烈建议您不要更改缺省文本配置对象中的设置。

表 2. 缺省文本配置设置

设置	安装值
TERM BREAKER	GENERIC
MINIMUM TERM LENGTH	1
MAXIMUM TERM LENGTH	20
STOPLIST	（空）

如果您删除缺省文本配置对象，则在下次您创建 **TEXT** 索引或文本配置对象时，系统会自动使用缺省值重新创建它。

使用 Interactive SQL 创建文本配置

创建文本配置来指定取决于文本配置进程的 **TEXT** 索引如何处理数据中的术语。

1. 以具有 CREATE TEXT CONFIGURATION 或 CREATE ANY TEXT CONFIGURATION 或 CREATE ANY OBJECT 系统特权的用户身份连接到数据库。
2. 执行 **CREATE TEXT CONFIGURATION** 语句。

使用 default_char 文本配置对象为模板，创建名为 myTxtConfig 的文本配置对象：

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

文本配置对象设置

了解文本配置对象设置、它们如何影响编制索引的内容，以及如何解释全文本搜索查询。

另请参见

- 文本配置对象设置解释 （第 13 页）

分词符算法 (TERM BREAKER)

TERM BREAKER 设置指定用于将字符串拆分成术语的算法。

SAP Sybase IQ **GENERIC**（缺省值）或 **NGRAM** 用于存储术语。

注意： **NGRAM** 分词符存储 n-gram。 n-gram 是一组长度为 *n* 的字符，其中 *n* 是 **MAXIMUM TERM LENGTH** 的值。

无论指定何种术语断开器，当术语插入到 **TEXT** 索引时，数据库服务器会在 **TEXT** 索引中记录这些术语的原始位置信息。如果是 n 元语法词，则会存储 n 元语法词的位置信息，而不是原始术语的位置信息。

表 3. TERM BREAKER 影响

对于 TEXT 索引	对于查询术语
• **GENERIC TEXT 索引** – 当生成 **GENERIC TEXT** 索引（缺省设置）时，数据库服务器将出现在非字母数字字符之间的字母数字字符组视为术语进行处理。在定义术语后，会对超过术语长度设置的术语和位于非索引字表中的术语进行计数，但不会将其插入 **TEXT** 索引中。 **GENERIC TEXT** 索引上的性能可能高于 **NGRAM TEXT** 索引上的性能。但无法对 **GENERIC TEXT** 索引执行模糊搜索。	• **GENERIC TEXT 索引** – 查询 **GENERIC TEXT** 索引时，将按照术语已建立索引的方式来处理查询字符串中的术语。通过比较查询术语和 **TEXT** 索引中的术语来执行匹配。
• **NGRAM TEXT 索引** – 构建 **NGRAM TEXT** 索引时，数据库服务器会将非字母数字字符之间的任意一组字母数字字符处理为术语。定义术语后，数据库服务器将术语分解为 n 元语法词。这样便可放弃长度小于 n 的术语以及非索引字表中的 n 元语法词。 例如，对于 **MAXIMUM TERM LENGTH** 为 3 的 **NGRAM TEXT** 索引，字符串 'my red table' 在 **TEXT** 索引中表示为以下 n 元语法词：red tab abl ble。	• **NGRAM TEXT 索引** – 查询 **NGRAM TEXT** 索引时，将按照术语已建立索引的方式来处理查询字符串中的术语。通过比较查询术语中的 n 元语法词和索引术语中的 n 元语法词来执行匹配。

最小术语长度设置 (MINIMUM TERM LENGTH)

MINIMUM TERM LENGTH 设置指定插入到索引中的术语或在全文本查询中搜索的术语的最小长度（以字符为单位）。

MINIMUM TERM LENGTH 与 **NGRAM TEXT** 索引不相关。

MINIMUM TERM LENGTH 对于前缀搜索有特殊含义。**MINIMUM TERM LENGTH** 的值必须大于 0。如果将其设置为大于 **MAXIMUM TERM LENGTH**，则 **MAXIMUM TERM LENGTH** 会自动调整为等于 **MINIMUM TERM LENGTH**。

MINIMUM TERM LENGTH 的缺省值是根据缺省文本配置对象中的设置得到的，通常为 1。

表 4. MINIMUM TERM LENGTH 影响

对于 TEXT 索引	对于查询术语
• **GENERIC TEXT 索引** – 对于 **GENERIC TEXT** 索引，**TEXT** 索引将不包含长度小于 **MINIMUM TERM LENGTH** 的单词。	• **GENERIC TEXT 索引** – 查询 **GENERIC TEXT** 索引时，将忽略长度小于 **MINIMUM TERM LENGTH** 的查询术语，因为它们不会存在于 **TEXT** 索引中。
• **NGRAM TEXT 索引** – 对于 **NGRAM TEXT** 索引，将忽略此设置。	• **NGRAM TEXT 索引** – **MINIMUM TERM LENGTH** 设置对 **NGRAM TEXT** 索引上的全文本查询没有影响。

最大术语长度设置 (MAXIMUM TERM LENGTH)

MAXIMUM TERM LENGTH 设置指定插入到索引中的术语或在全文本查询中搜索的术语的最大长度（以字符为单位）。

根据分词符算法的不同，**MAXIMUM TERM LENGTH** 设置有不同的用法。**MAXIMUM TERM LENGTH** 的值必须小于或等于 60。如果将 **MAXIMUM TERM LENGTH** 设置为小于 **MINIMUM TERM LENGTH**，则 **MINIMUM TERM LENGTH** 会自动调整为等于 **MAXIMUM TERM LENGTH**。

此设置的缺省值是根据缺省文本配置对象中的设置得到的，通常为 20。

表 5. MAXIMUM TERM LENGTH 影响

对于 TEXT 索引	对于查询术语
• **GENERIC TEXT 索引** – 对于 **GENERIC TEXT** 索引，**MAXIMUM TERM LENGTH** 可指定插入到 **TEXT** 索引中的术语的最大长度（以字符为单位）。	• **GENERIC TEXT 索引** – 对于 **GENERIC TEXT** 索引，将忽略长度大于 **MAXIMUM TERM LENGTH** 的查询术语，因为它们不会存在于 **TEXT** 索引中。
• **NGRAM TEXT 索引** – 对于 **NGRAM TEXT** 索引，**MAXIMUM TERM LENGTH** 用于确定术语分解后形成的 n 元语法词的长度。如何选择适当的 **MAXIMUM TERM LENGTH** 长度取决于相应的语言。对于英文，典型值为 4 个或 5 个字符；对于中文，典型值为 2 个或 3 个字符。	• **NGRAM TEXT 索引** – 对于 **NGRAM TEXT** 索引，查询术语将分解成长度为 n 的 n 元语法词，其中 n 和 **MAXIMUM TERM LENGTH** 相同。数据库服务器使用 n 元语法词搜索 **TEXT** 索引。将忽略长度小于 **MAXIMUM TERM LENGTH** 的术语，因为它们与 **TEXT** 索引中的 n 元语法词不匹配。

停止列表设置 (STOPLIST)

停止列表设置指定不编制索引的术语。

停止列表设置的缺省值是根据缺省文本配置对象中的设置得到的，通常会是一个空的停止列表。

表 6. STOPLIST 影响

对于 TEXT 索引	对于查询术语
• **GENERIC TEXT 索引** – 对于 **GENERIC TEXT** 索引，非索引字表中的术语不会插入到 **TEXT** 索引中。	• **GENERIC TEXT 索引** – 对于 **GENERIC TEXT** 索引，将忽略非索引字表中的查询术语，因为它们不会存在于 **TEXT** 索引中。
• **NGRAM TEXT 索引** – 对于 **NGRAM TEXT** 索引，由非索引字表中的术语形成的 n 元语法词不会包含在 **TEXT** 索引之中。	• **NGRAM TEXT 索引** – 非索引字表中的术语分解为 n 元语法词，然后将 n 元语法词用于非索引字表。类似地，查询术语将分解为 n 元语法词，然后删除任何与非索引字表中的 n 元语法词相匹配的项目，因为它们不会存在于 **TEXT** 索引中。

请仔细考虑是否将术语放入停止列表中。特别是，不要包含其中含有非字母数字字符（如撇号和横线）的单词。这些字符充当分词符。例如，单词 you'll（必须指定为“you'll”）拆分成 you 和 ll，并以这两个术语的形式存储在停止列表中。以后对“you”或“they'll”进行全文本搜索时会受到负面影响。

NGRAM TEXT 索引中的非索引字表可能导致意外的结果，因为所存储的非索引字表实际上采用的是 n 元语法词形式，而不是您指定的实际非索引字表术语形式。例如，对于 **MAXIMUM TERM LENGTH** 为 3 的 **NGRAM TEXT** 索引，如果指定 **STOPLIST** 'there'，则以下 n 元语法词将存储为非索引字表：the her ere。这会影响查询任何包含 n 元语法词 the、her 和 ere 的术语的能力。

使用 Interactive SQL 显示文本配置列表

查看数据库中所有文本配置的列表。

1. 以具有 CREATE TEXT CONFIGURATION 或 CREATE ANY TEXT CONFIGURATION 或 CREATE ANY OBJECT 系统特权的用户身份连接到数据库。
2. 执行 **SELECT** 语句。

列出所有文本配置对象：

```
SELECT * FROM SYSTEXTCONFIG;
```

使用 Interactive SQL 更改文本配置

更改文本配置对象的设置，包括数据库空间和准许的术语长度范围。

只能更改不被 **TEXT** 索引使用的文本配置对象。

1. 以具有 ALTER ANY TEXT CONFIGURATION 或 ALTER ANY OBJECT 系统特权的用户身份或以文本配置对象所有者身份连接到数据库。
2. 执行 **ALTER TEXT CONFIGURATION** 语句。

更改 `myTxtConfig` 文本配置对象的最小术语长度：

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

使用 Interactive SQL 修改停止列表

修改停止列表，该停止列表包含在使用此文本配置生成 **TEXT** 索引时要忽略的术语的列表。

只能更改不被 **TEXT** 索引使用的文本配置对象。

1. 以具有 ALTER ANY TEXT CONFIGURATION 或 ALTER ANY OBJECT 系统特权的用户身份或以文本配置对象所有者身份连接到数据库。
2. 执行带有 **STOPLIST** 子句的 **ALTER TEXT CONFIGURATION** 语句。

向 myTxtConfig 配置对象添加非索引字表：

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

使用 Interactive SQL 删除文本配置

从数据库中删除不必要的文本配置。

只能删除不被 **TEXT** 索引使用的文本配置。

1. 以具有 DROP ANY TEXT CONFIGURATION 或 DROP ANY OBJECT 系统特权的用户身份连接到数据库。
2. 执行 **DROP TEXT CONFIGURATION** 语句。

删除文本配置对象 myTxtConfig：

```
DROP TEXT CONFIGURATION myTxtConfig;
```

文本配置对象示例

查看这些示例以了解文本配置设置如何影响 **TEXT** 索引，以及索引是如何被解释的。

文本配置对象设置解释

显示不同文本配置对象的设置、这些设置如何影响编制索引的内容以及全文本查询字符串如何被解释的示例。

所有示例都使用字符串“I'm not sure I understand”。

表 7. 文本配置设置解释

配置设置	编制索引的术语	查询解释
TERM BREAKER: GENERIC MINIMUM TERM LENGTH: 1 MAXIMUM TERM LENGTH: 20 STOPLIST: "	`I m not sure I understand`	`'("I m" AND not sure) AND I AND understand'`
TERM BREAKER: GENERIC MINIMUM TERM LENGTH: 2 MAXIMUM TERM LENGTH: 20 STOPLIST: 'not and'	`sure understand`	`'understand'`
TERM BREAKER: GENERIC MINIMUM TERM LENGTH: 1 MAXIMUM TERM LENGTH: 20 STOPLIST: 'not and'	`I m sure I understand`	`'"I m" AND sure AND I AND understand'`

文本配置对象的 CONTAINS 查询字符串解释

系统如何为 CONTAINS 查询解释文本配置对象字符串设置的示例。

表“CONTAINS 字符串解释”的“解释的字符串”列中括号内的数字表示为各个术语存储的位置信息。文档中这些数字用于说明目的。实际存储的术语不包括括号内数字。

注意： 文本文档的最大位置数是 4294967295。

此表中的解释仅适用于 **CONTAINS** 查询。分析数据时，**AND**、**NOT** 和 **NEAR** 被认为是常规标识；*****、**|** 等符号以及其它符号会被删除，因为它们不是字母数字字符。

表 8. CONTAINS 字符串解释

配置设置	字符串	解释成的字符串
TERM BREAKER: GENERIC MINIMUM TERM LENGTH: 3 MAXIMUM TERM LENGTH: 20	`'w*'`	`'"w*(1)"'`
	`'we*'`	`'"we*(1)"'`
	`'wea*'`	`'"wea*(1)"'`
	`'we* -the'`	`'"we*(1)" -"the(1)"'`
	`'for* | wonderl*'`	`'"for*(1)" | "wonderl*(1)"'`

配置设置	字符串	解释成的字符串
	'wonderlandwonder-landwonderland*'	''
	'"tr* weather"'	'"weather(1)"'
	'"tr* the weather"'	'"the(1) weath-er(2)"'
	'"wonderlandwonder-landwonderland* wonderland"'	'"wonderland(1)"'
	'"wonderlandwonder-landwonderland* weather"'	'"weather(1)"'
	'"the_wonderland-wonderlandwonder-land* weather"'	'"the(1) weath-er(3)"'
	'the_wonderlandwon-derlandwonderland* weather'	'"the(1)" & "weath-er(1)"'
	'"light_a* the end" & tunnel'	'"light(1) the(3) end(4)" & "tun-nel(1)"'
	light_b* the end" & tunnel'	'"light(1) the(3) end(4)" & "tun-nel(1)"'
	'"light_at_b* end"'	'"light(1) end(4)"'
	'and-te*'	'"and(1) te*(2)"'
	'a_long_and_t* & journey'	'"long(2) and(3) t*(4)" & "jour-ney(1)"'

MAX_PREFIX_PER_CONTAINS_PHRASE 数据库选项

指定在文本搜索表达式中允许的前缀术语数目。

允许值

0 – 300

0 - 对搜索短语中的前缀术语没有限制

300 - 上限（这是对短语中允许的术语总数的总限制）

缺省值
1

范围
必须具有 SET ANY PUBLIC OPTION 系统特权才能设置此选项。可针对单个连接或 PUBLIC 角色进行临时设置。设置立即生效。

描述
MAX_PREFIX_PER_CONTAINS_PHRASE 指定用于禁止文本搜索表达式中存在多个前缀的阈值。

当此选项设置为 0 时，表示任意数量都可以。如果查询中有任何 **CONTAINS** 表达式具有所含前缀术语数量超过此选项所指定数量的短语，则 SAP Sybase IQ 会检测并报告此错误。

示例
使用缺省的 MAX_PREFIX_PER_CONTAINS_PHRASE 设置：

```
SET MAX_PREFIX_PER_CONTAINS_PHRASE = 1
```

以下 **CONTAINS** 子句就是有效的：

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

使用缺省的 MAX_PREFIX_PER_CONTAINS_PHRASE 设置 1，此 **CONTAINS** 子句将返回语法错误：

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

如果将 MAX_PREFIX_PER_CONTAINS_PHRASE 设置为等于 0（没有限制）或 2，此 **CONTAINS** 子句有效。

外部库

了解如何使用外部库来为文档提供预过滤和分词。

预过滤器和分词符外部库

SAP Sybase IQ 可以在索引创建或查询处理期间使用以 C 或 C++ 编写的外部前置过滤器和术语断开器库来对文档进行前置过滤和标记化。这些库可以动态装载到数据库服务器的进程空间中。

注意： 外部前置过滤器和术语断开器库必须由 SAP Sybase 认证的合作伙伴提供。有关认证供应商的解决方案的信息，请参见“合作伙伴认证报告 Web 站点”，然后过滤认证报告以显示 SAP Sybase IQ 认证。

外部的动态装载的预过滤器和分词符库是在文本配置中指定的，需要由数据库服务器装载。每个库中都包含一个导出的符号，用以实现在文本配置对象中指定的外部函数。此函数返回一组函数描述符，调用者会使用这些函数描述符来执行必要的任务。

外部预过滤器和分词符库是由数据库服务器在处理第一个 **CREATE TEXT INDEX** 请求时装载的，这种情况发生在收到的针对给定列的查询要求装载库时或者 **TEXT** 索引需要更新时。

在进行 **ALTER TEXT CONFIGURATION** 调用时不会装载这些库，在进行 **DROP TEXT CONFIGURATION** 调用时也不会自动卸载它们。如果服务器在启动时使用了禁止装载外部库的启动选项，则不会装载外部预过滤器和分词符库。

由于这些外部 C/C++ 库涉及将非服务器库代码装载到服务器的进程空间中，因此，编写错误或恶意编写的函数会给数据完整性、数据安全性和服务器稳健性带来潜在风险。为管理这些风险，每个服务器均可显式启用或禁用此功能。

`ISYSTEXTCONFIG` 系统视图存储与文本配置对象关联的外部库的信息。

另请参见

- 在启动时启用和禁用外部库 （第 18 页）

外部库限制

使用外部库的文本配置对象和 **TEXT** 索引在设计上有一些限制。

- 对于二进制列上的 **TEXT** 索引，必须使用由外部供应商提供的外部库来进行文档转换。SAP Sybase IQ 不会隐式转换二进制列中存储的文档。
- 如果使用外部术语断开器对文档进行标记化，则不支持基于 n 元语法词的文本搜索。

Multiplex 服务器上的外部库

所有 Multiplex 服务器都必须具有访问预过滤器和分词符外部库的权限。

用户必须确保每个外部预过滤器和分词符库都复制到承载 Multiplex 服务器的计算机上，并且放入服务器能够装载库的位置。

每个 Multiplex 服务器都在调用外部预过滤器和分词符库时独立于其它服务器而工作。每个进程空间都可以让外部库装载进来并完成自己的执行。假定预过滤器和分词符函数的实现在每个服务器上都一样，而且将返回相同的结果。

从一个服务器进程空间中卸载某个外部库并不会将该库从其它服务器进程空间中卸载。

在启动时启用和禁用外部库

SAP Sybase IQ 提供 **-sf** 启动开关来启用或禁用外部第三方库的装载。

此开关可以在服务器启动命令行中指定，也可以在服务器配置文件中指定。

允许装载外部第三方库：

```
-sf -external_library_full_text
```

禁止装载外部第三方库：

```
-sf external_library_full_text
```

要查看服务器中当前装载的库的列表，请使用 **sa_list_external_library** 存储过程。

卸载外部库

未使用外部库时，使用系统过程 **dbo.sa_external_library_unload** 可以卸载该库。

dbo.sa_external_library_unload 采用一个类型为 LONG VARCHAR 的可选参数。该参数指定要卸载的库的名称。如果不指定参数，则所有不在使用中的外部库都会被卸载。

卸载外部函数库：

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

非结构化数据查询

了解如何查询大对象数据，包括用于处理非结构化和半结构化数据的全文本搜索功能。

全文本搜索

全文本搜索使用 **TEXT** 索引快速在数据库中查找某个术语（单词）的所有实例，而不必扫描表行。

TEXT 索引存储带索引的列中各个术语的位置信息。使用 **TEXT** 索引查找包含某个术语的行要比扫描表中的每一行快。

全文本搜索使用 **CONTAINS** 搜索条件，这不同于使用 **LIKE**、**REGEXP** 和 **SIMILAR TO** 等谓词进行的搜索，因为这种匹配是基于术语的，而不是基于模式的。

全文搜索中的字符串比较使用数据库的所有常规归类设置。例如，将数据库配置为不区分大小写，则全文本搜索也不区分大小写。

另请参见

- CONTAINS 条件 （第 34 页）

全文本搜索的类型

通过使用全文本搜索，可以搜索术语、前缀或短语（术语序列）。另外，还可以将多个术语、短语或前缀组合到布尔型表达式中，或者使用临近搜索要求表达式看起来彼此接近。

在 **SELECT** 语句的 **WHERE** 子句或 **FROM** 子句中使用 **CONTAINS** 子句，执行全文本搜索。

术语和短语搜索

执行术语列表的全文搜索时，除非多个术语处于一个短语之中，否则术语的顺序不重要。如果将术语放在一个短语之中，数据库服务器将严格按照指定术语时的相同顺序和相对位置来查找这些术语。

执行术语或短语搜索时，如果术语因为超出术语长度设置或位于非索引字表之中而从查询中删除，则返回的行数可能与预期不符。这是因为从查询中删除术语相当于更改搜索条件。例如，如果搜索短语 `['"grown cotton"']` 并且 grown 位于非索引字表中，将得到所有包含 cotton 的索引行。

只要视作 CONTAINS 子句语法关键字的术语在短语之中，即可搜索它们。

术语搜索
在示例数据库中，MarketingInformation 表的 Description 列中创建了名为 MarketingTextIndex 的文本索引。以下语句查询 MarketingInformation.Description 列，并返回 Description 列中包含术语 `cotton` 的值所在的行。

```
SELECT ID, Description
   FROM MarketingInformation
   WHERE CONTAINS ( Description, 'cotton' );
```

ID	Description
906	<html><head><meta http-equiv=Content-Type content="text/html; charset=windows-1252"> <title>Visor</title></head><body lang=EN-US> Lightweight 100% organically grown **cotton** construction. Shields against sun and precipitation.cotton Metallic ions in the fibers inhibit bacterial growth, and help neutralize odor. </body></html>
908	<html><head><meta http-equiv=Content-Type content="text/html; charset=windows-1252"> <title>Sweatshirt</title></head><body lang=EN-US> Lightweight 100% organically grown **cotton** hooded sweatshirt with taped neck seams. Comes pre-washed for softness and to lessen shrinkage. </body></html>
909	<html><head><meta http-equiv=Content-Type content="text/html; charset=windows-1252"> <title>Sweatshirt</title></head><body lang=EN-US> Top-notch construction includes durable topstitched seams for strength with low-bulk, resilient rib-knit collar, cuffs and bottom. An 80% **cotton**/20% polyester blend makes it easy to keep them clean. </body></html>

ID	Description
910	<html><head><meta http-equiv=Content-Type content="text/html; charset=windows-1252"> <title>Shorts</title></head><body lang=EN-US> These quick-drying **cotton** shorts provide all day comfort on or off the trails. Now with a more comfortable and stretchy fabric and an adjustable drawstring waist. </body></html>

以下示例查询 MarketingInformation 表，并为每一行返回一个指示 Description 列中的值是否包含术语 `cotton` 的值。

```
SELECT ID, IF CONTAINS ( Description, 'cotton' )
      THEN 1
      ELSE 0
      ENDIF AS Results
   FROM MarketingInformation;
```

ID	Results
901	0
902	0
903	0
904	0
905	0
906	1
907	0
908	1
909	1
910	1

下一示例在 MarketingInformation 表中查询 Description 列中含有术语 `cotton` 的项目，然后显示各个匹配结果的分数。

```
SELECT ID, ct.score, Description
   FROM MarketingInformation CONTAINS
( MarketingInformation.Description, 'cotton' ) as ct
   ORDER BY ct.score DESC;
```

ID	score	Description
908	0.9461597363521859	<html><head><meta http-equiv=Content-Type content="text/html; charset=windows-1252"> <title>Sweatshirt</title></head><body lang=EN-US> Lightweight 100% organically grown **cotton** hooded sweatshirt with taped neck seams. Comes pre-washed for softness and to lessen shrinkage. </body></html>
910	0.9244136988525732	<html><head><meta http-equiv=Content-Type content="text/html; charset=windows-1252"> <title>Shorts</title></head><body lang=EN-US> These quick-drying **cotton** shorts provide all day comfort on or off the trails. Now with a more comfortable and stretchy fabric and an adjustable drawstring waist. </body></html>
906	0.9134171046194403	<html><head><meta http-equiv=Content-Type content="text/html; charset=windows-1252"> <title>Visor</title></head><body lang=EN-US> Lightweight 100% organically grown **cotton** construction. Shields against sun and precipitation. Metallic ions in the fibers inhibit bacterial growth, and help neutralize odor. </body></html>

ID	score	Description
909	0.8856420222728282	<html><head><meta http-equiv=Content-Type content="text/html; charset=windows-1252"> <title>Sweatshirt</title></head><body lang=EN-US> Top-notch construction includes durable top-stitched seams for strength with low-bulk, resilient rib-knit collar, cuffs and bottom. An 80% **cotton**/20% polyester blend makes it easy to keep them clean. </body></html>

短语搜索
对短语执行全文搜索时，将该短语括在双引号中。如果某一列包含具有指定顺序和相对位置的术语，则该列是匹配的。

不能指定 CONTAINS 关键字（如 AND 或 FUZZY ）作为要搜索的术语，除非将它们放在短语之内（允许只包含一个术语的短语）。例如，即使 NOT 是 CONTAINS 关键字，以下语句也是可接受的。

```
SELECT * FROM table-name CONTAINS ( Remarks, '"NOT"' );
```

如果特殊字符位于短语之中，则不会将其解释为特殊字符，但星号例外。

短语不能用作邻近搜索的参数。

以下语句在 MarketingInformation.Description 中查询短语 ["grown cotton"]，然后显示各个匹配结果的分数：

```
SELECT ID, ct.score, Description
   FROM MarketingInformation CONTAINS
( MarketingInformation.Description, '"grown cotton"' ) as ct
   ORDER BY ct.score DESC;
```

ID	score	Description
908	1.6619019465461564	<html><head><meta http-equiv=Content-Type content="text/html; charset=windows-1252"> <title>Sweatshirt</title></head><body lang=EN-US> Lightweight 100% organically grown **cotton** hooded sweatshirt with taped neck seams. Comes pre-washed for softness and to lessen shrinkage. </body></html>
906	1.6043904700786786	<html><head><meta http-equiv=Content-Type content="text/html; charset=windows-1252"> <title>Visor</title></head><body lang=EN-US> Lightweight 100% organically grown **cotton** construction. Shields against sun and precipitation. Metallic ions in the fibers inhibit bacterial growth, and help neutralize odor. </body></html>

前缀搜索

全文搜索功能允许搜索某一术语的开头部分。这称为**前缀搜索**。要执行前缀搜索，应指定要搜索的前缀，后跟一个星号。这称为**前缀术语**。

CONTAINS 子句的关键字不能用于前缀搜索，除非它们位于短语之中。

也可以在查询字符串中指定多个前缀术语，并将其包括在短语中（例如 `['"shi* fab"']`。）

以下示例在 MarketingInformation 表中查询以前缀 `shi` 开头的项目：

```
SELECT ID, ct.score, Description
   FROM MarketingInformation CONTAINS
( MarketingInformation.Description, 'shi*' ) AS ct
   ORDER BY ct.score DESC;
```

ID	score	Description
906	2.295363835537917	<html><head><meta http-equiv=Content-Type content="text/html; charset=windows-1252"> <title>Visor</title></head><body lang=EN-US> Lightweight 100% organically grown cotton construction. **Shi**elds against sun and precipitation. Metallic ions in the fibers inhibit bacterial growth, and help neutralize odor. </body></html>
901	1.6883275743936228	<html><head><meta http-equiv=Content-Type content="text/html; charset=windows-1252"> <title>Tee **Shi**rt</title></head><body lang=EN-US> We've improved the design of this perennial favorite. A sleek and technical **shi**rt built for the trail, track, or sidewalk. UPF rating of 50+. </body></html>
903	1.6336529491832605	<html><head><meta http-equiv=Content-Type content="text/html; charset=windows-1252"> <title>Tee **Shi**rt</title></head><body lang=EN-US> A sporty, casual **shi**rt made of recycled water bottles. It will serve you equally well on trails or around town. The fabric has a wicking finish to pull perspiration away from your skin. </body></html>

ID	score	Description
902	1.6181703448678983	<html><head><meta http-equiv=Content-Type content="text/html; charset=windows-1252"> <title>Tee **Shi**rt</title></head><body lang=EN-US> This simple, sleek, and lightweight technical **shi**rt is designed for high-intensity workouts in hot and humid weather. The recycled polyester fabric is gentle on the earth and soft against your skin. </body></html>

ID 906 的分数最高，因为术语 shield 在文本索引中的出现频率低于 shirt。

对 GENERIC 文本索引执行前缀搜索
对于 GENERIC 文本索引，前缀搜索的行为如下所示：

- 如果前缀术语的长度大于 MAXIMUM TERM LENGTH，则将其从查询字符串中删除，因为文本索引中不能存在长度超过 MAXIMUM TERM LENGTH 的术语。因此，对于 MAXIMUM TERM LENGTH 为 3 的文本索引，搜索 ['red appl*'] 相当于搜索 ['red']。
- 如果前缀术语的长度小于 MINIMUM TERM LENGTH，并且该术语不是短语搜索的一部分，则前缀搜索将正常进行。因此，对于 MINIMUM TERM LENGTH 为 5 的 GENERIC 文本索引，搜索 ['macintosh a*'] 将返回包含 macintosh 以及任何以 a 开头且长度大于等于 5 的术语的索引行。
- 如果前缀术语的长度小于 MINIMUM TERM LENGTH，但术语是短语搜索的一部分，则前缀术语将从查询中删除。因此，对于 MINIMUM TERM LENGTH 为 5 的 GENERIC 文本索引，搜索 ['"macintosh appl* turnover"'] 相当于搜索其后为任何后跟 turnover 的术语的 macintosh。不会查找包含 ["macintosh turnover"] 的行；macintosh 和 turnover 之间必须存在术语。

对 NGRAM 文本索引执行前缀搜索
对于 NGRAM 文本索引，前缀搜索可能返回意外的结果，因为 NGRAM 文本索引仅包含 n 元语法词，而不包含任何有关术语开头的信息。查询术语也将分解为 n 元语法词，然后使用 n 元语法词而不是查询术语执行搜索。因此，应注意以下行为：

- 如果前缀术语的长度小于 n 元语法词的长度 (MAXIMUM TERM LENGTH)，则查询将返回所有包含以该前缀术语开头的 n 元语法词的索引行。例如，对于 3 元语法词文本索引，搜索 ['ea*'] 将返回所有包含以 ea 开头的 n 元语法词的索引行。因此，如果对术语 weather 和 fear 建立索引，则行将被视为匹配，因为它们各自的 n 元语法词分别包含 eat 和 ear。
- 如果前缀术语的长度大于 n 元语法词长度，并且不是短语的一部分，也不是邻近搜索中的参数，则前缀术语将转换为 n 元语法词短语，并将星号删除。例如，对于 3

元语法词文本索引，搜索 ['purple blac*'] 相当于搜索 ['"pur urp rpl ple" AND "bla lac"']。

- 对于短语，还会发生以下行为：
  - 如果前缀术语是短语中的唯一术语，则将其转换为 n 元语法词短语，并删除星号。例如，对于 3 元语法词文本索引，搜索 ['"purpl*"'] 相当于搜索 ['"pur urp rpl"']。
  - 如果前缀术语在短语中处于最后位置，则删除星号并将术语转换为 n 元语法词短语。例如，对于 3 元语法词文本索引，搜索 ['"purple blac*"'] 相当于搜索 ['"pur urp rpl ple bla lac"']。
  - 如果前缀术语不在短语的最后位置，则短语分解为由 AND 连接的多个短语。例如，对于 3 元语法词文本索引，搜索 ['"purp* blac*"'] 相当于搜索 ['"pur urp" AND "bla lac"']。
- 如果前缀术语是邻近搜索中的参数，则邻近搜索转换为 AND。例如，对于 3 元语法词文本索引，搜索 ['red NEAR[1] appl*'] 相当于搜索 ['red AND "app ppl"']。

邻近搜索

全文搜索功能允许搜索某一列中相互邻近的多个术语。这称为**邻近搜索**。要执行邻近搜索，应指定关键字 NEAR 或 BEFORE 或者代字号 (~) 在其中间的两个术语。

可使用 proximity-expression 搜索彼此邻近的术语。例如，*b* NEAR[2,5] *c* 会搜索彼此间距离最多为 5 个术语，最少为 2 个术语的 *b* 和 *c* 的实例。术语的顺序并不重要；*b* NEAR *c* 等同于 *c* NEAR *b*。如果指定 NEAR 时未指定 distance，则应用缺省值十个术语。可指定代字号 (~) 代替 NEAR。这相当于指定 NEAR 时未指定距离，所以应用了缺省值 10 个术语。NEAR 表达式不可以链接在一起（例如，*a* NEAR[1] *b* NEAR[1] *c*）。

BEFORE 与 NEAR 相似，唯一的区别在于，其术语顺序比较重要。*b* BEFORE *c* 不同于 *c* BEFORE *b*；在前者中，术语 *b* 必须在 *c* 之前，而在后者中，术语 *b* 必须在 *c* 之后。BEFORE 既接受最小距离，也接受最大距离（与 NEAR 相同）。缺省的最小距离为 1。如给定最小距离，则其必须小于等于最大距离，否则将返回错误。

如果不指定距离，数据库服务器将使用 10 作为缺省距离。

还可以指定代字号 (~) 取代 NEAR 关键字。例如，'*term1 ~ term2*'。但是，在使用代字号格式时无法指定距离；此时应用十个术语的缺省值。

不能将短语指定为邻近搜索中的参数。

在使用 NGRAM 文本索引的邻近搜索中，如果指定前缀术语作为参数，则邻近搜索转换为 AND 表达式。例如，对于 3 元语法词文本索引，搜索 ['red NEAR[1] appl*'] 相当于搜索 ['red AND "app ppl"']。由于这不再是邻近搜索，所以在 CONTAINS 子句中指定了多列的情况下搜索不再限于单一列。

示例

假定要在 MarketingInformation.Description 中搜索含有术语 skin 的 10 个术语中的术语 fabric。此时可以执行以下语句。

```
SELECT ID, "contains".score, Description
   FROM MarketingInformation CONTAINS ( Description, 'fabric ~
skin' );
```

ID	score	Description
902	1.5572371866083279	<html><head><meta http-equiv=Content-Type content="text/html; charset=windows-1252"> <title>Tee Shirt</title></head><body lang=EN-US> This simple, sleek, and lightweight technical shirt is designed for high-intensity workouts in hot and humid weather. The recycled polyester **fabric** is gentle on the earth and soft against your **skin**. </body></html>

由于缺省距离即为 10 个术语，因此不必指定距离。但如果将距离增加一个术语，将返回另一行：

```
SELECT ID, "contains".score, Description
   FROM MarketingInformation CONTAINS ( Description, 'fabric NEAR[11]
skin' );
```

ID	score	Description
903	1.5787803210404958	<html><head><meta http-equiv=Content-Type content="text/html; charset=windows-1252"> <title>Tee Shirt</title></head><body lang=EN-US> A sporty, casual shirt made of recycled water bottles. It will serve you equally well on trails or around town. The **fabric** has a wicking finish to pull perspiration away from your **skin**. </body></html>

ID	score	Description
902	2.163125855043747	<html><head><meta http-equiv=Content-Type content="text/html; charset=windows-1252"> <title>Tee Shirt</title></head><body lang=EN-US> This simple, sleek, and lightweight technical shirt is designed for high-intensity workouts in hot and humid weather. The recycled polyester **fabric** is gentle on the earth and soft against your **skin**. </body></html>

ID 903 的分数较高，因为其术语间的距离较近。

布尔搜索

在执行全文搜索时，可以指定由布尔运算符分隔的多个术语。在执行全文搜索时，SQL Anywhere 支持以下布尔运算符：AND、OR 和 AND NOT。

在全文搜索中使用 AND 运算符

使用 AND 运算符得到的匹配行是包含在 AND 两侧指定的两个术语的行。对于 AND 运算符，还可以使用和符号 (&)。如果指定的术语之间没有任何运算符，则暗指采用 AND 运算符。

术语的列出顺序并不重要。

例如，以下各个语句都可在 MarketingInformation.Description 中查找包含术语 `fabric` 和以 `ski` 开头的术语的行：

```
SELECT *
   FROM MarketingInformation
   WHERE CONTAINS ( MarketingInformation.Description, 'ski* AND
fabric' );
SELECT *
   FROM MarketingInformation
   WHERE CONTAINS ( MarketingInformation.Description, 'fabric &
ski*' );
SELECT *
   FROM MarketingInformation
   WHERE CONTAINS ( MarketingInformation.Description, 'ski*
fabric' );
```

在全文搜索中使用 OR 运算符

使用 OR 运算符得到的匹配行是至少包含在 OR 两侧指定的任一搜索术语的行。对于 OR 运算符，还可以使用竖线 (|)；两者是等同的。

术语的列出顺序并不重要。

例如，以下任一语句都可返回 MarketingInformation.Description 中包含术语 fabric 或以 ski 开头的术语的行：

```
SELECT *
   FROM MarketingInformation
   WHERE CONTAINS ( MarketingInformation.Description, 'ski* OR
fabric' );
SELECT *
   FROM MarketingInformation
   WHERE CONTAINS ( MarketingInformation.Description, 'fabric |
ski*' );
```

在全文搜索中使用 AND NOT 运算符
使用 AND NOT 运算符查找的结果匹配左侧参数但不匹配右侧参数。对于 AND NOT 运算符，还可以使用连字符 (-)；两者是等同的。

例如，以下语句互相等效，都返回包含术语 fabric 但不包含任何以 ski 开头的术语的行。

```
SELECT *
   FROM MarketingInformation
  WHERE CONTAINS ( MarketingInformation.Description, 'fabric AND NOT
ski*' );
SELECT *
   FROM MarketingInformation
   WHERE CONTAINS ( MarketingInformation.Description, 'fabric -
ski*' );
SELECT *
   FROM MarketingInformation
   WHERE CONTAINS ( MarketingInformation.Description, 'fabric & -
ski*' );
```

组合不同的布尔运算符
可在查询字符串中组合布尔运算符。例如，以下语句相互等效，都在 MarketingInformation.Description 列中搜索包含 fabric 和 skin，但不包含 cotton 的项目：

```
SELECT *
   FROM MarketingInformation
   WHERE CONTAINS ( MarketingInformation.Description, 'skin fabric -
cotton' );
SELECT *
   FROM MarketingInformation
  WHERE CONTAINS ( MarketingInformation.Description, 'fabric -cotton
AND skin' );
```

以下语句相互等效，都在 MarketingInformation.Description 列中搜索包含 fabric，或同时包含 cotton 和 skin 的项目：

```
SELECT *
   FROM MarketingInformation
   WHERE CONTAINS ( MarketingInformation.Description, 'fabric |
cotton AND skin' );
SELECT *
```

```
   FROM MarketingInformation
  WHERE CONTAINS ( MarketingInformation.Description, 'cotton skin OR
fabric' );
```

对术语和短语分组
可以使用括号对术语和表达式分组。例如，以下语句在 MarketingInformation.Description 列中搜索包含 `cotton` 或 `fabric`，并包含以 `ski` 开头的术语的项目。

```
SELECT ID, Description FROM MarketingInformation
   WHERE CONTAINS( MarketingInformation.Description, '( cotton OR
fabric ) AND shi*' );
```

ID	Description
902	<html><head><meta http-equiv=Content-Type content="text/html; charset=windows-1252"> <title>Tee Shirt</title></head><body lang=EN-US> This simple, sleek, and light-weight technical **shi**rt is designed for high-intensity workouts in hot and humid weather. The recycled polyester **fabric** is gentle on the earth and soft against your skin. </body></html>
903	<html><head><meta http-equiv=Content-Type content="text/html; charset=windows-1252"> <title>Tee Shirt</title></head><body lang=EN-US> A sporty, casual **shi**rt made of recycled water bottles. It will serve you equally well on trails or around town. The **fabric** has a wicking finish to pull perspiration away from your skin. </body></html>
906	<html><head><meta http-equiv=Content-Type content="text/html; charset=windows-1252"> <title>Visor</title></head><body lang=EN-US> Lightweight 100% organically grown **cotton** construction. **Shi**elds against sun and pre-cipitation. Metallic ions in the fibers inhibit bacterial growth, and help neutralize odor. </body></html>

在多列间执行搜索
可以通过单一查询在多个列之间执行全文搜索，前提是这些列是同一文本索引的一部分。

```
SELECT *
   FROM t
      WHERE CONTAINS ( t.c1, t.c2, 'term1|term2' );
```

```
SELECT *
   FROM t
   WHERE CONTAINS( t.c1, 'term1' )
      OR CONTAINS( t.c2, 'term2' );
```

如果 *t1.c1* 包含 *term1*，或 *t1.c2* 包含 *term2*，则第一个查询匹配。

如果 *t1.c1* 或 *t1.c2* 包含 *term1* 和 *term2* 中的任一个，则第二个查询匹配。以这种方式使用 CONTAINS 子句也会返回匹配分数。

模糊搜索
模糊搜索可用于搜索拼写错误的单词或单词的变体。为此，可在带双引号的字符串前使用 FUZZY 运算符，来查找该字符串的近似匹配。例如，`CONTAINS ( Products.Description, 'FUZZY "cotton"' )` 将返回 `cotton` 及其拼写错误的形式，如 `coton` 或 `cotten`。

注意： 只能对使用 NGRAM 术语断开器构建的文本索引执行模糊搜索。

使用 FUZZY 运算符相当于手动将字符串分解成长度为 *n* 的子字符串并使用 OR 运算符将它们分隔开。例如，假定有一个配置有 NGRAM 术语断开器且最大术语长度为 3 的文本索引。则指定 `'FUZZY "500 main street"'` 相当于指定 `'500 OR mai OR ain OR str OR tre OR ree OR eet'`。

FUZZY 运算符在返回分数的全文搜索中很有用。这是因为可能会返回许多近似匹配项，但通常只有分数最高的匹配项才有意义。

视图搜索
要对视图或派生表使用全文搜索，必须在要执行全文搜索的基表中的列上构建文本索引。以下语句对示例数据库中的 MarketingInformation 表创建一个其中已有文本索引名称的视图，然后对该视图执行全文搜索。

要对 MarketingInformation 基表创建视图，请执行以下语句：

```
CREATE VIEW MarketingInfoView AS
SELECT MI.ProductID AS ProdID,
     MI."Description" AS "Desc"
FROM GROUPO.MarketingInformation AS MI
WHERE MI."ID" > 3
```

利用以下语句，您可以在基础表上用文本索引来查询视图。

```
SELECT *
FROM MarketingInfoView
WHERE CONTAINS ( "Desc", 'Cap OR Tee*' )
```

也可执行以下语句，以在基础表上使用文本索引来查询派生表。

```
SELECT *
FROM (
     SELECT MI.ProductID, MI."Description"
     FROM MarketingInformation AS MI
     WHERE MI."ID" > 4 ) AS dt ( P_ID, "Desc" )
WHERE CONTAINS ( "Desc", 'Base*' )
```

注意： 注释

您想要对其运行全文搜索的列必须包括在视图或派生表的 SELECT 列表中。

在基础性基表上使用文本索引搜索视图有如下限制：

- 视图不能含有 TOP、FIRST、DISTINCT、GROUP BY、ORDER BY、UNION、INTERSECT 和 EXCEPT 子句或窗口函数。
- 视图不能包含集合函数。
- CONTAINS 查询可查询某一视图内的基表，但不能查询另一个视图内的某一视图内的基表。

FROM 子句
指定 **SELECT** 语句中涉及的数据库表或视图。

语法

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

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

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

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

用法
contains-expression - 在表名后使用 **CONTAINS** 子句可对表进行过滤，仅返回那些与使用 *contains-query* 指定的全文本查询匹配的行。

表的每个匹配行与分数列一起返回，可以使用 *score-correlation-name*（如果已指定）引用此分数列。如果未指定 *score-correlation-name*，则可使用缺省相关名 *contains* 来引用分数列。

除了可选的相关名参数外，**CONTAINS** 子句采用与 **CONTAINS** 搜索条件相同的参数。**CONTAINS** 子句中列出的列上必须有 **TEXT** 索引。

另请参见

- CONTAINS 条件 （第 34 页）

CONTAINS 条件

在 **SELECT** 语句的 **FROM** 子句中使用 **CONTAINS** 子句，或者在 **WHERE** 子句中使用 **CONTAINS** 搜索条件（谓词）来执行全文本查询。

这两种方法返回同样的行；但 **CONTAINS** 子句还返回匹配行的分数。

语法

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

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

simple-expression:
   primary-expression | and-expression

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


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

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

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

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

fuzzy-expression:
   term | fuzzy-expression term


term:
   simple-term | prefix-term

prefix-term:
   simple-term*

phrase:
   " phrase-string "
```

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

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

参数

- **simple-term** – 由空格或特殊字符分隔的字符串，表示要搜索的单个索引术语。
- **distance** – 正整数。
- **and-expression** – 指定 *primary-expression* 和 *simple-expression* 均必须出现在 **TEXT** 索引中。缺省情况下，如果在术语或表达式之间未指定任何运算符，则假定为 and-expression。例如，`'a b''` 被解释为 `'a AND b'`。和号 (&) 可用来代替 AND，并且可以紧靠表达式或术语的任何一侧（例如，`'a & b'`）。
- **and-not-expression** – 指定 *primary-expression* 必须出现在 **TEXT** 索引中，但 *basic-expression* 不得出现在 **TEXT** 索引中。这也称为取非。当使用连字符表示取非时，必须在连字符前加一个空格，且连字符必须紧靠后面的术语。例如，`'a -b'` 等同于 `'a AND NOT b'`；而对于 `'a - b'`，连字符被忽略，该字符串等同于 `'a AND b'`。`'a-b'` 等同于短语 `'"a b"'`。
- **or-expression** – 指定 *simple-expression* 或 *contains-query-string* 中至少有一个必须出现在 **TEXT** 索引中。例如，`'a|b'` 被解释为 `'a OR b'`。
- **fuzzy-expression** – 查找与您指定的术语相类似的术语。只有 **NGRAM TEXT** 索引上支持模糊匹配。
- **proximity-expression** – 搜索彼此邻近的术语。例如，`'b NEAR[2,5] c'` 搜索相互间隔最多 5 个术语、最少 2 个术语的 b 和 c 实例。术语的顺序并不重要；`'b NEAR c'` 等效于 `'c NEAR b'`。如果指定 **NEAR** 时未指定 *distance*，则应用缺省值 10 个术语。可指定代字号 (~) 代替 **NEAR**。这相当于指定 **NEAR** 时未指定距离，所以采用缺省值 10 个术语。**NEAR** 表达式无法链接到一起（例如，`'a NEAR[1] b NEAR[1] c'`）。

  BEFORE 与 **NEAR** 类似，只不过术语的顺序很重要。`'b BEFORE c'` 不等效于 `'c BEFORE b'`；在前者中，术语 `'b'` 必须在 `'c'` 的前面，而在后者中，术语 `'b'` 必须在 `'c'` 的后面。**BEFORE** 与 **NEAR** 一样，接受最小和最大距离。缺省的最小距离为 1。如给定最小距离，则其必须小于等于最大距离，否则将返回错误。
- **prefix-term** – 搜索以指定前缀开头的术语。例如，`'datab*'` 搜索以 *datab* 开头的任何术语。这也称为前缀搜索。在前缀搜索中，执行匹配的是星号左侧的术语部分。

用法

CONTAINS 搜索条件采用列列表和 *contains-query-string* 作为参数。

CONTAINS 搜索条件可以用在任何可以指定搜索条件（也称为谓词）的地方，返回 TRUE 或 FALSE。*contains-query-string* 必须是常量字符串，或者变量（其值在查询时是已知的）。

如果指定了多个列，则它们全都必须引用单个基表；一个 **TEXT** 索引不能跨多个基表。可以在 **FROM** 子句中直接引用基表，也可以在视图或派生表中使用它，但前提条件是该视图或派生表不使用 **DISTINCT**、**GROUP BY**、**ORDER BY**、**UNION**、**INTERSECT**、**EXCEPT** 或行限制。

支持使用 ANSI 连接语法（**FULL OUTER JOIN**、**RIGHT OUTER JOIN**、**LEFT OUTER JOIN**）进行查询，但性能可能无法达到最优。仅在需要每个 **CONTAINS** 子句中的 `score` 列时才将外连接用于 **FROM** 子句中的 **CONTAINS**。否则，请将 **CONTAINS** 移动到 **ON** 条件或 **WHERE** 子句中。

不支持以下查询类型：

- 使用 SQL Anywhere 表的远程查询，而且该表具有连接到远程表的完整 **TEXT** 索引。
- 使用 SAP Sybase IQ 和 SQL Anywhere 表的查询，其中，SQL Anywhere 表具有要使用的完整 **TEXT** 索引。
- 使用 TSQL 样式外部连接语法（`*=*`、`=*` 和 `*=`）的查询。

如果使用长度小于 32KB 的 SQL 变量作为搜索术语，且变量类型为 `LONG VARCHAR`，则使用 **CAST** 将该变量转换为 `VARCHAR` 数据类型。例如：

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

以下警告适用于在查询字符串中使用非字母数字字符的情况：

- 在术语中间使用星号会返回错误。
- 避免在 fuzzy-expression 中使用非字母数字字符（包括特殊字符），因为它们会被视为空格并用作术语断开符。
- 尽量避免在查询字符串中使用不是特殊字符的非字母数字字符。任何不是特殊字符的非字母数字字符都会使包含该字符的术语被视为一个短语，从而在字符所在的位置断开术语。例如，`'things we've done'` 被解释为 `'things "we ve" done'`。

在短语中，星号是唯一继续被解释为特殊字符的特殊字符。短语中的所有其它特殊字符都会被视为空格充当分词符。

contains-query-string 的解释分为两个主要步骤：

- 第 1 步：解释运算符和优先级：在执行此步骤期间，将关键字解释为运算符，并应用优先级规则。
- 第 2 步：应用文本配置对象设置：在执行此步骤期间，将文本配置对象设置应用于术语。任何查询术语若超过术语长度设置或位于非索引字表中，都会被删除。

另请参见

- 对 TEXT 索引执行模糊搜索 （第 40 页）

CONTAINS 搜索条件中的运算符优先级
在查询求值期间，会使用优先级顺序对表达式求值。

查询表达式计算的优先级顺序为：

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

星号 (*) 的允许语法
星号用于查询中的前缀搜索。

星号可以出现在查询字符串的结尾，也可以后跟空格、与符号、竖线、右括号或右引号。星号的任何其它用法都会返回错误。

“星号解释”表显示了允许的星号用法：

表 9. 星号解释

查询字符串	等同于	解释为
`'th*&best'`	`'th* AND best'` 和 `'th* best'`	查找所有以 th 开头的术语，并查找术语 best。
`'th*|best'`	`'th* OR best'`	查找所有以 th 开头的术语，或查找术语 best。
`'very&(best|th*)'`	`'very AND (best OR th*)'`	查找术语 very，并查找术语 best 或任何以 th 开头的术语。
`'"fast auto*"'`		查找术语 fast，随后紧跟以 auto 开头的术语。
`'"auto* price"'`		查找以 auto 开头的术语，随后紧跟术语 price。

注意： 包含星号的查询字符串的解释随文本配置对象设置的不同而不同。

连字符 (-) 的允许语法
连字符可以用于查询中的术语或表达式否定，等同于 NOT。

连字符是否被解释为否定取决于其在查询字符串中的位置。例如，当连字符紧挨在某个术语或表达式的前面时，它会被解释为否定。如果连字符内嵌在某个术语内，它就会被解释为连字符。

用于否定的连字符必须前面带一个空格，而且后面紧跟一个表达式。

当用于模糊表达式的短语时，连字符被视为空格并用作分词符。

“连字符解释”表显示了连字符的允许语法：

表 10. 连字符解释

查询字符串	等同于	解释为
`'the -best'`	`'the AND NOT best', 'the AND -best', 'the & -best', 'the NOT best'`	查找术语 the，而不是术语 best。
`'the -(very best)'`	`'the AND NOT (very AND best)'`	查找术语 the，而不是术语 very 和 best。
`'the -"very best"'`	`'the AND NOT "very best"'`	查找术语 the，而不是短语 very best。
`'alpha-numerics'`	'"alpha numerics"'	查找术语 alpha，且其后紧跟术语 numerics。
`'wild - west'`	`'wild west'`和 `'wild AND west'`	查找术语 wild，并查找术语 west。

特殊字符的允许语法
“特殊字符解释”表显示除星号和连字符外的所有特殊字符的允许语法。

星号和连字符如果位于某个短语中，就不会被认为是特殊字符，而且会被删除。

注意： 有关指定字符串文字的限制也适用于查询字符串。例如，撇号必须位于转义序列内。

表 11. 特殊字符解释

字符或语法	用法示例和注释
和号 (&)	和号等同于 **AND**，可指定为以下形式： • `'a & b'` • `'a &b'` • `'a& b'` • `'a&b'`

字符或语法	用法示例和注释
竖线 (|)	竖线等同于 **OR**，可指定为以下形式： • `'a| b'` • `'a |b'` • `'a | b'` • `'a|b'`
双引号 (")	双引号用于包含那些顺序和相对距离都很重要的一连串术语。例如，在查询字符串 `'learn "full text search"'` 中，"full text search" 是一个短语。在此示例中，learn 可以放置在短语的前面或后面，或者位于另一个列中（如果 **TEXT** 索引基于不止一个列构建），但必须在单个列中找到完全一样的短语。
括号 ()	如果表达式的计算顺序与缺省顺序不同，则使用括号指定表达式的计算顺序。例如，`'a AND (b|c)'` 被解释为 a 和 b 或 c。
代字号 (~)	代字号等同于 **NEAR**，没有特殊的语法规则。查询字符串 `'full~text'` 等同于 `'full NEAR text'`，可解释为：术语 full 和术语 text 的距离在十个术语范围之内。
方括号 [ ]	方括号与关键字 **NEAR** 结合使用以包含 *distance*。方括号的其它用法会返回错误。

被删除术语的影响
TEXT 索引可排除满足特定条件的术语。

TEXT 索引根据为用来创建 **TEXT** 索引的文本配置对象而定义的设置进行构建。**TEXT** 索引中不包括满足以下任一条件的术语：

- 术语包括在非索引字表中。
- 术语的长度小于最小术语长度（仅限 **GENERIC**）。
- 术语长度比最大术语长度长。

此规则也适用于查询字符串。删除的术语可以匹配短语末尾或开头处的零个或多个术语。例如，假定术语 'the' 位于非索引字表中：

- 如果该术语出现在 **AND**、**OR** 或 **NEAR** 的任意一侧，则同时删除该运算符和该术语。例如，搜索 `'the AND apple'`、`'the OR apple'` 或 `'the NEAR apple'` 相当于搜索 `'apple'`。
- 如果该术语出现在 **AND NOT** 的右侧，则 **AND NOT** 和该术语均会被删除。例如，搜索 `'apple AND NOT the'` 相当于搜索 `'apple'`。
- 如果该术语出现在 **AND NOT** 的左侧，则整个表达式都会被删除。例如，搜索 `'the AND NOT apple'` 不会返回任何行。另一个示例：`'orange and the AND NOT apple'` 与 `'orange AND (the AND NOT apple)'` 相同，后者在 **AND NOT** 表达式被删除后等同于搜索 `'orange'`。而搜索表达式 `'(orange and the) and not apple'` 则与之相反，它等同于搜索 `'orange and not apple'`。

- 如果该术语出现在短语中，则允许该短语与出现在该删除的术语位置处的任何术语匹配。例如，搜索 'feed the dog' 匹配 'feed the dog'、'feed my dog'、'feed any dog' 等。

注意： 如果您要搜索的所有术语都被删除，则 SAP Sybase IQ 将返回错误 CONTAINS 具有 Null 个搜索术语。SQL Anywhere 不会报告任何错误，也不返回任何行。

查询匹配分数
可使用表示匹配相似程度的分数对查询结果排序。

在查询的 **FROM** 子句中使用 **CONTAINS** 子句时，各个匹配都会有与之相关联的分数。分数表明了匹配的近似程度，因此可以使用分数信息来排序数据。分数主要由两个标准决定：

- 术语在索引行中出现的次数。术语在索引行中出现的次数越多，其分数就越高。
- 术语在 **TEXT** 索引中出现的次数。术语在 **TEXT** 索引中出现的次数越多，其分数就越低。

根据全文搜索类型的不同，其它标准也可能影响计分过程。例如，在邻近搜索中，搜索术语的接近程度会影响计分过程。缺省情况下，**CONTAINS** 子句的结果集具有相关名 contains，其中包含一个名为 score 的列。可以引用 **SELECT** 列表、**ORDER BY** 子句或查询的其它部分中的 "contains".score。但是，由于 contains 是 SQL 保留字，因此一定要将其放在双引号之中。也可以指定另外的相关名，例如 CONTAINS (expression) AS ct。全文本搜索示例以 ct.score 的形式引用 score 列。

以下语句在 MarketingInformation.Description 中搜索以 "stretch" 开头的术语或以 "comfort" 开头的术语：

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

NGRAM TEXT 索引搜索

对于 **NGRAM** 类型的 **TEXT** 索引，可以通过 **TEXT** 索引实现模糊和非模糊搜索功能。

对 TEXT 索引执行模糊搜索

仅当 **TEXT** 索引为 **NGRAM** 类型时，才能通过 **TEXT** 索引实现模糊搜索功能。**GENERICTEXT** 索引无法处理模糊搜索。

模糊搜索可用于搜索拼写错误的单词或单词的变体。为此，可在带双引号的字符串前使用 **FUZZY** 运算符，来查找该字符串的近似匹配。

使用 **FUZZY** 运算符相当于手动将字符串分解成长度为 *n* 的子字符串并使用 **OR** 运算符将它们分隔开。例如，如果您有一个配置有 **NGRAM** 术语断开器且 **MAXIMUM TERM**

LENGTH 为 3 的文本索引，则指定 `'FUZZY "500 main street" '` 相当于指定 `'500 OR mai OR ain OR str OR tre OR ree OR eet'`。

FUZZY 运算符在返回分数的全文本搜索中很有用。系统可能会返回许多近似匹配项，但通常只有分数最高的匹配项才有意义。

注意： 模糊搜索不支持前缀或后缀搜索。例如，搜索子句不能是“v*”或“*vis”。

示例 1：通过 NGRAM TEXT 索引执行模糊搜索

创建一个表和一个 **NGRAMTEXT** 索引：

```
CREATE TEXT CONFIGURATION NGRAMTxtcfg
   FROM default_char;
ALTER TEXT CONFIGURATION NGRAMTxtcfg TERM BREAKER    NGRAM;
ALTER TEXT CONFIGURATION NGRAMTxtcfg maximum term    length 3;
CREATE TABLE t_iq(a int, b varchar(100));
CREATE TEXT INDEX TXT_IQ on t_iq(b) CONFIGURATION    NGRAMTxtcfg
```

将以下数据插入到该表中：

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

插入数据以后，执行以下查询来通过 **NGRAMTEXT** 索引执行模糊搜索：

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

查询结果如下：

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

示例 2：模糊搜索子句中的附加字母

以下查询说明模糊搜索子句中的附加字母：

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

查询结果如下：

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

示例 3：从模糊搜索子句中删除的字母

在以下查询中，系统从模糊搜索子句中删除了一个字母：

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

查询结果如下：

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

对 TEXT 索引执行非模糊搜索

基于 **NGRAM** 的非模糊搜索可以将术语拆分成相应的 n-gram 并搜索 **NGRAMTEXT** 索引中的 n-gram。

查询 `CONTAINS ( M.Description, 'ams' ) ct;` 说明了针对 2GRAM 索引的非模糊 **NGRAM** 搜索，该搜索在语义上等同于搜索查询 `CONTAINS( M.Description, '"am ms"' ) ct;`

如果您基于 2GRAM 索引搜索“v*”术语，则后面跟有任意字母的 v 被视为搜索术语的匹配 2GRAM，并且会输出为结果。

查询 `CONTAINS (M.Description, ‘white whale’) ct;` 说明了针对 3GRAM 索引的非模糊 **NGRAM** 搜索，该搜索在语义上等同于搜索查询 `CONTAINS (M.Description, ‘” whi hit ite wha hal ale”’);`

NGRAM 模糊和非模糊搜索之间的区别是，模糊搜索对单个 GRAMS 进行了分离。而非模糊搜索则对单个 GRAMS 进行了结合。在同一列上创建 **GENERIC** 和 **NGRAMTEXT** 索引时，**GENERICTEXT** 索引用于带有非模糊搜索的查询，而 **NGRAMTEXT** 索引则用于模糊搜索。

示例 1：在同一列上创建 GENERICTEXT 索引以后执行非模糊搜索

以下查询说明在同一列上创建 **GENERICTEXT** 索引以后执行非模糊搜索：

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

查询结果如下：

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

示例 2：在同一列上同时使用 NGRAM 和 GENERICTEXT 索引执行模糊搜索

以下查询说明在同一列上同时使用 **NGRAM** 和 **GENERICTEXT** 索引执行模糊搜索：

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

查询结果如下：

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

示例 3：非模糊搜索子句中的模糊搜索短语

以下查询说明非模糊搜索子句中的模糊搜索短语行为：

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

系统没有为此查询返回任何结果。

查询 LONG BINARY 列

在 **SELECT** 语句的 **WHERE** 子句中，除 **BYTE_LENGTH64**、**BYTE_SUBSTR64**、**BYTE_SUBSTR**、**BIT_LENGTH**、**OCTET_LENGTH**、**CHARINDEX** 和 **LOCATE** 函数外，LONG BINARY 列只能用于 IS NULL 和 IS NOT NULL 表达式中。

LONG BINARY 列不能用在 **SELECT** 语句的 **ORDER BY**、**GROUP BY** 和 **HAVING** 子句中，也不能与 **DISTINCT** 关键字一起使用。

SAP Sybase IQ 不支持在 **LONG BINARY** (BLOB) 列或变量上使用 **LIKE** 谓词。使用 **LIKE** 谓词在 LONG BINARY 列中搜索模式，会返回错误 `Invalid data type comparison in predicate`。

另请参见

- 函数支持 （第 77 页）

查询 LONG VARCHAR 列

在 **SELECT** 语句的 **WHERE** 子句中，除 **BIT_LENGTH**、**CHAR_LENGTH**、**CHAR_LENGTH64**、**CHARINDEX**、**LOCATE**、**OCTET_LENGTH**、**PATINDEX**、**SUBSTRING64** 和 **SUBSTRING** 函数外，LONG VARCHAR 列只能用于 IS NULL 和 IS NOT NULL 表达式中。

可使用 **LIKE** 谓词在 LONG VARCHAR 列上搜索模式。支持所有长度小于等于 126 个字符的模式。不支持长度大于 254 个字符的模式。根据模式的内容，支持长度在 127 和 254 个字符之间的某些模式。

LIKE 谓词支持任意数据大小的 **LONG VARCHAR** (CLOB) 变量。目前，一个 SQL 变量可以容纳长度最大为 2GB - 1 的数据。

LONG VARCHAR 列不能用在 **SELECT** 语句的 **ORDER BY**、**GROUP BY** 和 **HAVING** 子句中，也不能与 **DISTINCT** 关键字（**SELECT DISTINCT** 和 **COUNT DISTINCT**）一起使用。

另请参见

- 函数支持 （第 77 页）

CONTAINS 谓词支持

可以在 LONG VARCHAR (CLOB) 列上创建 **WORD** (**WD**) 索引，并使用 **CONTAINS** 谓词在该列中搜索最大长度为 255 个字符的字符串常量。

不支持在使用 **WD** 索引的 **LONG BINARY** (BLOB) 列中使用 **CONTAINS** 谓词。如果尝试通过 **CONTAINS** 谓词在使用 **WD** 索引的 LONG BINARY 列中搜索字符串，则会返回错误。使用外部库的 **TEXT** 索引支持对二进制数据使用 **CONTAINS**。

LONG BINARY 和 LONG VARCHAR 列的性能监控

性能监控器显示 LONG BINARY 和 LONG VARCHAR 列的性能数据。

存储过程支持

了解数据类型为 LONG BINARY (BLOB) 和 LONG VARCHAR (CLOB) 的列以及全文本搜索的存储过程支持。

TEXT 索引中的术语管理

可使用存储过程将字符串拆分成术语，查找 **TEXT** 索引中有多少术语以及各个术语的位置，显示有关 **TEXT** 索引的统计信息。

sa_char_terms 系统过程

将 CHAR 字符串拆分为多个术语，并以行返回每个术语及其位置。

语法

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

参数

- **char-string** – 所分析的 CHAR 字符串。
- **text-config-name** – 处理字符串时应用的文本配置对象。缺省值为 'default_char'。
- **owner** – 指定的文本配置对象的所有者。缺省值为 DBA。

特权
无

注释
可以使用 **sa_char_terms** 来查明在应用文本配置对象的设置时如何解释字符串。这可以帮助您了解哪些术语将在编制索引期间被删除或者将从查询字符串中被删除。

示例
返回 CHAR 字符串中的术语 'the quick brown fox jumped over the fence':

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

表 12. CHAR 字符串解释

术语	位置
the	1

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

sa_nchar_terms 系统过程

将 NCHAR 字符串分解为多个术语，并以行返回每个术语及其位置。

语法

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

参数

- **char-string** – 所分析的 NCHAR 字符串。
- **text-config-name** – 处理字符串时应用的文本配置对象。缺省值为 'default_nchar'。
- **owner** – 指定的文本配置对象的所有者。缺省值为 DBA。

特权
您必须具有系统过程的 EXECUTE 特权。

注释
可以使用 **sa_nchar_terms** 来了解在应用文本配置对象的设置时如何解释字符串。如果想知道索引过程中哪些术语会被删除或者会从查询字符串中删除哪些术语，这一点很有用。

sa_nchar_terms 的语法与 **sa_char_terms** 系统过程的语法类似。

注意： 仅 **IN SYSTEM** 表支持 NCHAR 数据类型。

sa_text_index_stats 系统过程

返回有关数据库中 TEXT 索引的统计信息。

语法

```
sa_text_index_stats( )
```

特权
您必须具有系统过程的 EXECUTE 特权。 必须具有以下一种系统特权：

- MANAGE ANY STATISTICS
- CREATE ANY INDEX
- ALTER ANY INDEX
- DROP ANY INDEX
- CREATE ANY OBJECT
- ALTER ANY OBJECT
- DROP ANY OBJECT

注释
使用 **sa_text_index_stats** 可以查看数据库中的每个 TEXT 索引的统计信息。

表 13. sa_text_index_stats 返回的 TEXT 索引的统计信息

列名	类型	描述
owner_id	UNSIGNED INT	表所有者的 ID
table_id	UNSIGNED INT	表 ID
index_id	UNSIGNED INT	TEXT 索引的 ID
text_config_id	UNSIGNED BIGINT	TEXT 索引引用的文本配置的 ID
owner_name	CHAR(128)	所有者的名称
table_name	CHAR(128)	表的名称
index_name	CHAR(128)	TEXT 索引的名称
text_config_name	CHAR(128)	文本配置对象的名称
doc_count	UNSIGNED BIGINT	TEXT 索引中的索引列值总数
doc_length	UNSIGNED BIGINT	TEXT 索引中数据的总长度
pending_length	UNSIGNED BIGINT	待执行更改的总长度
deleted_length	UNSIGNED BIGINT	待执行删除的总长度
last_refresh	TIMESTAMP	最后一次刷新的日期和时间

对于 IMMEDIATE REFRESH TEXT 索引，`pending_length`、`deleted_length` 和 `last_refresh` 的值为 NULL。

示例
返回数据库中每个 **TEXT** 索引的统计信息：

```
CALL sa_text_index_stats( );
```

sa_text_index_vocab 系统过程

列出所有在 **TEXT** 索引中出现的术语以及每个术语在其中出现的索引值的总数。

语法

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

参数

- **text-index-name** – 此 CHAR(128) 参数用于指定 **TEXT** 索引的名称。
- **table-name** – 此 CHAR(128) 参数用于指定构建 **TEXT** 索引时所基于的表的名称。
- **table-owner** – 此 CHAR(128) 参数用于指定表的所有者。

特权

您必须具有系统过程的 EXECUTE 特权。 必须具有以下一种:

- SELECT ANY TABLE 系统特权
- 对索引表的 SELECT 特权

注释

sa_text_index_vocab 返回 **TEXT** 索引中出现的所有术语，以及显示每个术语的带索引值的总数（如果术语在某些带索引值中出现多次，则该值小于总出现次数）。

参数值不能是宿主变量或表达式。参数 *text-index-name*、*table-name* 和 *table-owner* 必须是约束或变量。

示例

执行 **sa_text_index_vocab** 以返回 GROUPO 拥有的表 Customers 上 **TEXT** 索引 MyTextIndex 中出现的所有术语:

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

表 14. 索引中的术语

术语	次数
a	1
Able	1
Acres	1

术语	次数
Active	5
Advertising	1
Again	1
...	...

外部库标识

sa_list_external_library 存储过程列出服务器中当前装载的库。在标识库后，可以使用 **sa_external_library_unload** 将库从服务器中卸载。

sa_external_library_unload 系统过程

卸载外部库。

语法

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

参数

- **external-library** – 此可选 LONG VARCHAR 参数用于指定要卸载的库的名称。如果未指定任何库，则将卸载所有未使用的外部库。

特权

您必须具有系统过程的 EXECUTE 特权。 您还必须具有 MANAGE ANY EXTERNAL OBJECT 系统特权。

注释

如果指定了外部库，但它在使用中或还未装载，则会返回错误。如果未指定任何参数，则在找不到任何已装载的外部库时，会返回错误。

示例

卸载名为 myextlib.dll 的外部库：

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

卸载当前未使用的所有库：

```
CALL sa_external_library_unload();
```

sa_list_external_library 系统过程

列出服务器上当前装载的外部库。

语法

```
sa_list_external_library ( )
```

特权

您必须具有系统过程的 EXECUTE 特权。 您还必须具有 MANAGE ANY EXTERNAL OBJECT 系统特权。

注释

返回引擎中装载的外部库的列表以及各个库的引用计数。

引用计数是引擎中库的实例数。可以通过执行过程 **sa_external_library_unload** 来卸载外部库（只要其引用计数是 0）。

示例

列出外部库及其引用计数：

```
CALL sa_list_external_library()
```

大对象数据压缩

sp_iqsetcompression 存储过程控制大对象列的压缩设置。

在将数据库缓冲区写入磁盘时，**sp_iqsetcompression** 控制数据类型为 LONG BINARY 和 LONG VARCHAR 的列的压缩。另外，还可以使用 **sp_iqsetcompression** 来禁用压缩。此功能可节省 CPU 周期，因为 LONG BINARY 或 LONG VARCHAR 列中存储的某些数据格式（例如，JPG 文件）已经压缩，即使再进行压缩也不会取得任何效果。

sp_iqshowcompression 存储过程显示大对象列的压缩设置。

sp_iqsetcompression 过程

设置数据类型为 LONG BINARY (BLOB) 和 LONG VARCHAR (CLOB) 的列中的数据压缩。

语法

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

权限

您必须具有系统过程的 EXECUTE 特权。 必须具有以下一种系统特权：

- ALTER ANY TABLE

- ALTER ANY OBJECT

注释
sp_iqsetcompression 用于控制数据类型为 LONG BINARY (BLOB) 和 LONG VARCHAR (CLOB) 的列的压缩。压缩设置仅适用于基表。

sp_iqsetcompression 的副作用是更改压缩设置之后将会发生 **COMMIT**。

表 15. sp_iqsetcompression 的参数

名称	描述
owner	要设置压缩的表的所有者
table	要设置压缩的表
column	要设置压缩的列
on_off_flag	压缩设置： ON 表示启用压缩，OFF 表示禁用压缩

示例
假定以下表定义：

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

要禁用对 LOB 列 picJPG 的压缩，请调用 **sp_iqsetcompression**：

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

此命令不返回任何行。

sp_iqshowcompression 过程

显示数据类型为 LONG BINARY (BLOB) 和 LONG VARCHAR (CLOB) 的列的压缩设置。

语法

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

特权
您必须具有系统过程的 EXECUTE 特权。 必须具有以下一种系统特权：

- ALTER ANY TABLE
- ALTER ANY OBJECT

注释
返回列名和压缩设置。压缩设置值为“ON”（启用压缩）和“OFF”（禁用压缩）。

表 16. sp_iqshowcompression 的参数

名称	描述
owner	要设置压缩的表的所有者
table	要设置压缩的表
column	要设置压缩的列

示例
假定以下表定义：

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

要检查 pixTable 表中列的压缩状态，请调用 **sp_iqshowcompression**：

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

此命令返回一行：

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

有关大对象列的信息

存储过程 **sp_iqindexsize** 显示单个 LONG BINARY 和 LONG VARCHAR 列的大小。

LONG BINARY 列的大小

显示了一个具有约 42GB 数据的 LONG BINARY 列的 **sp_iqindexsize** 输出。

页大小为 128KB。largelob Info 类型在最后一行中。

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

在此示例中，压缩比为 42953952/(623009*128) = 53.9%。

LONG VARCHAR 列的大小

显示了一个具有约 42GB 数据的 `LONG VARCHAR` 列的 **sp_iqindexsize** 输出。

页大小为 128KB。`largelob` Info 类型在最后一行中。

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

在此示例中，压缩比为 42953952/(623009*128) = 53.9%。

大对象数据装载和卸载

了解如何导出和装载大对象数据。

大对象数据导出

SAP Sybase IQ 数据提取工具包含 **BFILE** 函数，可用于将各个 LONG BINARY 和 LONG VARCHAR 单元提取到服务器上的各个操作系统文件中。

BFILE 不一定要与数据抽取工具一起使用。

BFILE 函数

将各个 LONG BINARY 和 LONG VARCHAR 单元提取到服务器上的各个操作系统文件中。

语法

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

参数

- **file-name-expression** – LONG BINARY 或 LONG VARCHAR 数据将写入到的输出文件的名称。此文件名最大长度可以为 (32K -1) 字节，但必须是文件系统支持的有效路径名。
- **large-object-column** – LONG BINARY 或 LONG VARCHAR 列的名称。

用法

BFILE 返回：

- 如果文件成功写入，则为 1
- 如果文件未成功打开或写入，则为 0
- 如果 LONG BINARY 或 LONG VARCHAR 单元值为 NULL，则为 NULL

如果 LONG BINARY 或 LONG VARCHAR 单元值为 NULL，则不会打开任何文件，也不会写入任何数据。

文件路径相对于服务器的启动位置，且打开和写入操作使用服务器进程的权限执行。**BFILE** 输出文件不支持磁带设备。

使用除 **BFILE** 函数外的其它函数检索到的 LONG BINARY 和 LONG VARCHAR 单元（即，之后通过客户端/服务器数据库连接检索到的单元）受到最大长度为 2GB 的大小限制。使用 **SUBSTRING64** 或 **BYTE_SUBSTR64** 可以通过 **SELECT**（**SELECT**、**OPEN CURSOR**）检索大于 2GB 的 LONG BINARY 单元。使用 **SUBSTRING64** 可以通过 **SELECT**（**SELECT**、**OPEN CURSOR**）检索大于 2GB 的 LONG VARCHAR 单元。某些

连接驱动程序（例如 ODBC、JDBC™ 和 Open Client™）不允许在一个 **SELECT** 中返回超过 2GB 的数据。

BFILE 不一定要与数据抽取工具一起使用。

BFILE 函数示例

使用 **BFILE** 提取和重新装载 LOB 数据。

创建表 LobA：

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

假定 LobA 有两行数据。

提取非 LOB 数据以及 LOB 数据所提取到的文件的路径：

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

系统将创建 LobA_data.txt 文件，该文件包含此非 LOB 数据和以下文件名：

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

执行 LOB 数据提取：

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

提取后，所提取的 LOB 数据的每个单元都有一个文件。例如，如果表 LobA 包含 rowid 值分别为 1 和 2 的两行数据，您会有以下文件：

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

重新装载已提取的数据：

```
LOAD TABLE LobA
(rowid,
  col1 ASCII FILE (',') NULL('NULL'),
  col2 BINARY FILE (',') NULL('NULL'))
FROM LobA_data.txt'
```

```
DELIMITED BY ','
ROW DELIMITED BY '\n'
ESCAPES OFF;
```

大对象数据装载

使用 **LOAD TABLE** 语句的扩展语法来装载 LONG BINARY 和 LONG VARCHAR 数据。

可以从 ASCII 或 BCP 格式的主文件装载的大对象数据没有大小限制，除非操作系统具有相关限制。从主文件加载到大对象列的固定宽度数据的最大长度为 32K - 1。

还可以在主装载文件中指定辅助装载文件。每个单独辅助数据文件正好包含一个 LONG BINARY 或 LONG VARCHAR 单元值。

扩展 LOAD TABLE 语法

为了装载大对象数据，**LOAD TABLE** 对语法进行了扩展。

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

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

关键字 **BINARY FILE**（对于 LONG BINARY）或 **ASCII FILE**（对于 LONG VARCHAR）对装载进行指定：列的主输入文件包含辅助文件（其中包含 LONG BINARY 或 LONG VARCHAR 单元值）的路径，而不包含 LONG BINARY 或 LONG VARCHAR 数据本身。辅助文件路径名可以是完全限定的，也可以是相对的。如果辅助文件路径名不是完全限定的，则该路径相对于服务器的启动目录。辅助文件不支持磁带设备。

SAP Sybase IQ 支持在主装载文件中装载无长度限制的 LONG BINARY 和 LONG VARCHAR 值（受操作系统限制）。在将十六进制格式的二进制数据从主文件装载到 LONG BINARY 列中时，SAP Sybase IQ 要求十六进制数字的总数为偶数。如果单元值所含的十六进制数字的数量为奇数，将报告错误“在列上检测到二进制数据值的奇数长度”。用于对 LONG BINARY 进行装载的输入文件所含的十六进制数字的数量应始终为偶数。

SAP Sybase IQ 不支持使用 **LOAD TABLE…FORMAT BINARY** 从主文件装载大对象列。可以从辅助文件中装载二进制格式的大对象数据。

对于 **LOAD TABLE FORMAT BCP**，装载规范只能包含列名、**NULL** 和 **ENCRYPTED**。这表示在使用 **LOAD TABLE FORMAT BCP** 选项装载 LONG BINARY 和 LONG VARCHAR 列时，不能使用辅助文件。

大对象数据装载示例

使用 LONG BINARY 数据创建并装载表。

```
CREATE TABLE ltab (c1 INT, filename CHAR(64),
    ext CHAR(6), lobcol LONG BINARY NULL);
```

```
LOAD TABLE ltab (
    c1,
    filename,
    ext NULL( 'NULL' ),
    lobcol BINARY FILE ( ',' ) NULL( 'NULL' )
)
FROM 'abc.inp'
QUOTES OFF ESCAPES OFF;
```

主文件 abc.inp 包含以下数据：

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

将 LONG BINARY 数据装载到表 tab 中后，列 lobcol 的第一行和第二行分别包含文件 boston.jpg 和 concord.bmp 的内容。第三行和第四行分别包含零长度值和 NULL。

装载错误的控制

使用数据库选项 SECONDARY_FILE_ERROR 可以指定，在从辅助 **BINARY FILE** 或 **ASCII FILE** 进行打开或读取时出现错误的情况下要执行的装载操作。

如果 SECONDARY_FILE_ERROR 为 ON，则在从辅助 **BINARY FILE** 或 **ASCII FILE** 进行打开或读取时如果出现错误，将回退装载。

如果 SECONDARY_FILE_ERROR 为 OFF（缺省设置），则无论在从辅助 **BINARY FILE** 或 **ASCII FILE** 进行打开或读取时是否出现错误，均将继续装载。LONG BINARY 或 LONG VARCHAR 单元将获得以下任一值：

- 如果列允许空值，则为 NULL
- 如果列不允许空值，则为零长度值

任何用户都可以为 PUBLIC 角色或临时角色设置 SECONDARY_FILE_ERROR；选项设置会立即生效。

在将完整性约束违规记录到装载错误 **ROW LOG** 文件时，为 LONG BINARY 列或 LONG VARCHAR 列记录的信息是：

- 如果记录发生在第一道装载操作内，则为从主数据文件读取的实际文本
- 如果记录发生在第二道装载操作内，则为零长度值

装载包含尾随空白的大对象数据

LOAD TABLE...STRIP 选项对于 LONG VARCHAR 数据没有影响。

不从 LONG VARCHAR 数据去除尾随空白，即使 **STRIP** 选项为 ON，也是如此。

装载带引号的大对象数据

LOAD TABLE...QUOTES 选项不适用于从辅助文件装载 LONG BINARY (BLOB) 或 LONG VARCHAR (CLOB) 数据，无论其设置如何。

前导或尾随引号将作为 CLOB 数据的一部分来装载。使用 **QUOTESON** 选项，位于引号之间的两个连续引号将作为两个连续引号进行装载。

截断部分多字节字符数据

在装载期间，将根据 TRIM_PARTIAL_MBC 数据库选项值对部分多字节 LONG VARCHAR 数据进行截断。

- 如果 TRIM_PARTIAL_MBC 为 ON，对于主数据和带 **ASCII FILE** 选项的 **LOAD**，均会截断部分多字节字符。
- 如果 TRIM_PARTIAL_MBC 为 OFF，则带 **ASCII FILE** 选项的 **LOAD** 将根据 SECONDARY_FILE_ERROR 数据库选项值来处理部分多字节字符。

“使用 ASCII FILE 选项装载 LONG VARCHAR 时的部分多字节字符”表列出了尾随的多字节字符是如何根据 TRIM_PARTIAL_MBC 和 SECONDARY_FILE_ERROR 值进行装载的。

表 17. 使用 ASCII FILE 选项装载 LONG VARCHAR 时的部分多字节字符

TRIM_PARTIAL_MBC	SECONDARY_FILE_ERROR	找到尾随的部分多字节字符
ON	ON/OFF	截断尾随的部分多字节字符
OFF	ON	单元 - 如果允许空值，则为空值 LOAD 错误 - 如果不允许为空值，则回退
OFF	OFF	单元 - 如果允许空值，则为空值 单元 - 如果不允许为空值，则为零长度

大对象变量的装载支持

LOAD TABLE、**INSERT…VALUES**、**INSERT…SELECT**、**INSERT…LOCATION**、**SELECT…INTO** 和 **UPDATE** SQL 语句支持大对象变量。

另请参见

- 大对象数据类型 （第 61 页）
- 大对象变量 （第 63 页）

大对象数据类型

了解大对象 LONG BINARY 和 LONG VARCHAR 数据类型列的特性以及大对象数据的索引支持。

大对象数据类型 LONG BINARY 和 BLOB

SAP Sybase IQ 中的二进制大对象 (BLOB) 数据存储在数据类型为 LONG BINARY 或 BLOB 的列中。

对于大小为 128KB 的 IQ 页，单个 LONG BINARY 数据值的长度范围可以是零 (0) 到 512TB（千吉字节）；对于大小为 512KB 的 IQ 页，可以是零 (0) 到 2PB（千万亿字节）。（最大长度为 4GB 乘以数据库页面大小。）要容纳含有 LONG BINARY 数据的表，必须使用大小至少为 128KB（131072 字节）的 IQ 页面来创建 IQ 数据库。

表或数据库可以包含任意数量的 LONG BINARY 列，只要其分别小于各个表和各个数据库所支持的该列最大数即可。

LONG BINARY 列可以是 NULL，也可以是 NOT NULL，且可以存储零长度的值。域 BLOB 为允许 NULL 的 LONG BINARY 数据类型。

无法在 LONG BINARY 列上构造非 FP 索引。

如果结果集包含 BLOB 列，则会禁用预取。

使用 **UPDATE**、**INSERT**、**LOAD TABLE**、**DELETE**、**TRUNCATE**、**SELECT...INTO** 和 **INSERT...LOCATION** SQL 语句修改 LONG BINARY 列。LONG BINARY 列不支持定位更新和删除。

可以使用 **INSERT...LOCATION** 命令向 LONG BINARY 列插入 Adaptive Server IMAGE 列。插入的所有 IMAGE 数据均以无提示方式正好在 2147483648 字节 (2GB) 处右截断。

LONG BINARY 数据类型转换

其它数据类型与 LONG BINARY 数据类型和非 LONG BINARY 数据类型之间存在有限的隐式数据类型转换。

除用于 **INSERT** 和 **UPDATE** 的 BINARY 和 VARBINARY 数据类型外，不存在从 LONG BINARY 数据类型到其它非 LONG BINARY 数据类型的任何隐式数据类型转换。存在从 TINYINT、SMALLINT、INTEGER、UNSIGNED INTEGER、BIGINT、UNSIGNED BIGINT、CHAR 和 VARCHAR 数据类型到 LONG BINARY 数据类型的隐式转换。不存在从 BIT、REAL、DOUBLE 或 NUMERIC 数据类型到 LONG BINARY 数据类型的隐式转换。可以使用 CONVERSION_MODE 数据库选项控制隐式转换。

LONG BINARY 数据类型当前支持的字节子字符串函数作为 **INSERT** 和 **UPDATE** 语句的隐式转换的输入被接受。

LONG BINARY 数据类型可以显式转换为 BINARY 或 VARBINARY。不存在任何其它以 LONG BINARY 数据类型为源类型或目标类型的显式数据类型转换（例如，使用 **CAST** 或 **CONVERT** 函数）。

在 LONG BINARY 转换到 BINARY 或 VARBINARY 期间对 LONG BINARY 数据截断的处理方式与对 BINARY 和 VARBINARY 数据截断的处理方式相同。如果 STRING_RTRUNCATION 选项为 ON，则对二进制列执行 **INSERT** 或 **UPDATE** 操作时，对任何值（不仅是非空格字符）的任何右截断均会导致截断错误，并回退事务。

另请参见

- 函数支持 （第 77 页）

大对象数据类型 LONG VARCHAR 和 CLOB

SAP Sybase IQ 中的字符大对象 (CLOB) 数据存储在数据类型为 LONG VARCHAR 或 CLOB 的列中。

对于大小为 128KB 的 IQ 页，单个 LONG VARCHAR 数据值的长度范围可以是零 (0) 到 512TB（千吉字节）；对于大小为 512KB 的 IQ 页，可以是零 (0) 到 2PB（千万亿字节）。（最大长度为 4GB 乘以数据库页面大小。）要容纳含有 LONG VARCHAR 数据的表，必须使用大小至少为 64KB（65536 字节）的 IQ 页面来创建 IQ 数据库。

表或数据库可以包含任意数量的 LONG VARCHAR 列，只要其分别小于各个表和各个数据库所支持的该列最大数即可。

SAP Sybase IQ 同时支持单字节和多字节 LONG VARCHAR 数据。

LONG VARCHAR 列可以是 NULL，也可以是 NOT NULL，且可以存储零长度的值。域 CLOB 为允许 NULL 的 LONG VARCHAR 数据类型。要创建非空 LONG VARCHAR 列，请在列定义中显式指定 NOT NULL。

在创建表或向现有表添加列时，您可以使用 CLOB 域创建 LONG VARCHAR 列。例如：

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

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

可以在 LONG VARCHAR 列上创建 **WORD** (**WD**) 索引。但无法在 LONG VARCHAR 列上构造其它非 **FP** 索引类型和连接索引。

可以使用 **UPDATE**、**INSERT...VALUES**、**INSERT...SELECT**、**LOAD TABLE**、**DELETE**、**TRUNCATE**、**SELECT...INTO** 和 **INSERT...LOCATION** SQL 语句修改 LONG VARCHAR 列。LONG VARCHAR 列不支持定位更新和删除。

可以使用 **INSERT...LOCATION** 命令向 LONG VARCHAR 列插入 Adaptive Server TEXT 列。插入的所有 TEXT 数据均以无提示方式正好在 2147483648 字节 (2GB) 处右截断。

LONG VARCHAR 数据类型转换

其它数据类型与 LONG VARCHAR 数据类型和非 LONG VARCHAR 数据类型之间存在有限的隐式数据类型转换。

除用于 **INSERT** 和 **UPDATE** 的 LONG BINARY、CHAR 和 VARCHAR 数据类型外，不存在从 LONG VARCHAR 数据类型到其它非 LONG VARCHAR 数据类型的任何隐式数据类型转换。存在从 CHAR 和 VARCHAR 数据类型到 LONG VARCHAR 数据类型的隐式转换。不存在从 BIT、REAL、DOUBLE、NUMERIC、TINYINT、SMALLINT、INT、UNSIGNED INT、BIGINT、UNSIGNED BIGINT、BINARY、VARBINARY 或 LONG BINARY 数据类型到 LONG VARCHAR 数据类型的隐式转换。可以使用 CONVERSION_MODE 数据库选项控制隐式转换。

LONG VARCHAR 数据类型当前支持的字符串函数作为 **INSERT** 和 **UPDATE** 语句的隐式转换的输入被接受。

LONG VARCHAR 数据类型可以显式转换为 CHAR 和 VARCHAR。不存在任何其它以 LONG VARCHAR 数据类型为源类型或目标类型的显式数据类型转换（例如，使用 **CAST** 或 **CONVERT** 函数）。

在 LONG VARCHAR 转换到 CHAR 期间对 LONG VARCHAR 数据截断的处理方式与对 CHAR 数据截断的处理方式相同。如果 STRING_RTRUNCATION 选项为 ON 并对非空格进行字符串右截断，则会报告截断错误并回退事务。转换时，尾随的部分多字节字符会被替换为空格。

在 LONG VARCHAR 转换到 VARCHAR 期间对 LONG VARCHAR 数据截断的处理方式与对 VARCHAR 数据截断的处理方式相同。如果 STRING_RTRUNCATION 选项为 ON 并对非空格进行字符串右截断，则会报告截断错误并回退事务。转换时，尾随的部分多字节字符会被截断。

另请参见

- 函数支持 （第 77 页）

大对象变量

SAP Sybase IQ 支持大对象变量。

入站 LONG BINARY 和 LONG VARCHAR 变量（由 IQ 使用的主机变量或 SQL 变量）没有最大长度。

出站 LONG BINARY 和 LONG VARCHAR 变量（由 IQ 设置的变量）具有最大长度，为 2GB - 1。

LOAD TABLE、**INSERT…VALUES**、**INSERT…SELECT**、**INSERT…LOCATION**、**SELECT…INTO** 和 **UPDATE** SQL 语句接受任意数据大小的 LONG BINARY 和 LONG VARCHAR 变量。目前，一个 SQL 变量可以容纳长度最大为 2GB - 1 的数据。

BIT_LENGTH、**BYTE_LENGTH**、**BYTE_LENGTH64**、**BYTE_SUBSTR**、**BYTE_SUBSTR64**、**CHARINDEX**、**LOCATE**、 **OCTET_LENGTH** 和 **SUBSTRING64** 函数支持 SQL 变量可容纳的任意数据大小的 LONG BINARY 和 LONG VARCHAR 变量。此外，**CHAR_LENGTH**、**CHAR_LENGTH64**、**PATINDEX**、**SUBSTR** 和 **SUBSTRING** 函数还支持 SQL 变量可容纳的任意数据大小的 LONG VARCHAR 变量。

大对象变量数据类型转换

数据库选项 ENABLE_LOB_VARIABLES 控制大对象变量的数据类型转换。

ENABLE_LOB_VARIABLES 选项

控制大对象变量的数据类型转换。

允许值

ON、OFF

缺省值

OFF

范围

可在数据库 (PUBLIC) 或用户级别设置选项。在数据库级别进行设置时，值将变为任何新用户的缺省值，但不会对现有用户产生任何影响。在用户级别进行设置时，仅替换该用户的 PUBLIC 值。为自身设置选项无需任何系统特权。在数据库或用户级别为任何其他用户设置选项都需要系统特权。

必须具有 SET ANY PUBLIC OPTION 系统特权才能设置此选项。可针对个别连接或 PUBLIC 角色进行临时设置。设置立即生效。

注释

ENABLE_LOB_VARIABLES 控制大对象变量的数据类型转换。

当 **ENABLE_LOB_VARIABLES** 为 OFF 时，会隐式转换小于 32K 的大对象变量；如果大对象变量大于或等于 32K，会报告错误。LONG VARCHAR 变量会隐式转换为 VARCHAR 数据类型，并在 32K 处截断。LONG BINARY 变量会隐式转换为 VARBINARY 数据类型，并在 32K 处截断。

当 **ENABLE_LOB_VARIABLES** 为 ON 时，任何大小的大对象变量都会保留其原始数据类型和大小。

示例

对于大于 32K 的大对象变量，保持其数据类型和大小：

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

大对象列的索引支持

SAP Sybase IQ 支持 LONG BINARY 和 LONG VARCHAR 列的 **TEXT** 索引以及 LONG VARCHAR 列的 **WORD** (**WD**) 索引。

大对象列的 TEXT 索引支持

TEXT 索引支持 LONG BINARY 和 LONG VARCHAR 列。

另请参见

- SQL 语句支持 （第 67 页）
- TEXT 索引和文本配置对象 （第 3 页）

LONG VARCHAR (CLOB) 列的 WD 索引支持

SAP Sybase IQ 对 LONG VARCHAR (CLOB) 列的 **WORD** (**WD**) 索引的支持有限。

- **WD** 索引支持的最宽列为 LOB 列的最大宽度。（最大长度为 4GB 乘以数据库页面大小。）SAP Sybase IQ 支持的最大字长度为 255 字节。
- LONG VARCHAR (CLOB) 列还支持针对 CHAR 和 VARCHAR 列 **WD** 索引的所有 **sp_iqcheckdb** 选项，包括分配、检查和检验模式。
- 可使用 **sp_iqrebuildindex** 针对 LONG VARCHAR (CLOB) 列重建 **WD** 索引。

采用二进制格式的中文文本或文档要求 ETL 预处理，以便定位单词并将其转换为 **WD** 索引可分析的格式。

SQL 语句支持

了解支持对 **TEXT** 索引和文本配置进行处理的 SQL 语句和语法。

ALTER TEXT CONFIGURATION 语句

改变文本配置对象。

注意： 该语句需要非结构化数据分析 (IQ_UDA) 许可证。

快速链接:

转至参数 （第 67 页）

转至示例 （第 68 页）

转至用法 （第 69 页）

转至权限 （第 69 页）

语法

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

参数

(返回顶部) （第 67 页）

- **非索引字表** – string-expression 用于创建或替换建立 TEXT 索引时要忽略的术语列表。查询中也将忽略此列表中指定的术语。使用空格将非索引字表隔离。

  非索引字表术语不得包含空格，也不应包含非字母数字字符。非字母数字字符被解释为空格，并且将术语分解成多个术语。例如，"and/or" 被解释为两个术语 "and" 和 "or"。非索引字表术语的最大数量为 7999。

- **DROP STOPLIST** – 用于删除文本配置对象的非索引字表。

- **MINIMUM TERM LENGTH** – 指定 TEXT 索引中包含的术语的最小长度（以字符数表示）。使用 NGRAM TEXT 索引时将忽略在 MINIMUM TERM LENGTH 子句中指定的值。

构建或刷新 TEXT 索引时，将忽略长度短于该设置的术语。该选项的值必须大于 0。如果将此选项设置为大于 MAXIMUM TERM LENGTH，则 MAXIMUM TERM LENGTH 的值会自动进行调整到与新的 MINIMUM TERM LENGTH 值相等。

- **MAXIMUM TERM LENGTH** – 通过 GENERIC TEXT 索引，指定 TEXT 索引中包含的术语的最大长度（以字符数表示）。构建或刷新 TEXT 索引时，将忽略长度大于该设置的术语。

  MAXIMUM TERM LENGTH 的值必须小于或等于 60。如果将此项设置为小于 MINIMUM TERM LENGTH，则 MINIMUM TERM LENGTH 的值会自动进行调整到与新的 MAXIMUM TERM LENGTH 的值相等。

- **TERM BREAKER** – 对用于将列值分隔为术语时所用算法的名称进行指定。可以为 IN SYSTEM 表选择 GENERIC（缺省值）或 NGRAM。GENERIC 算法将任意由一个或多个字母数字构成并由非字母数字分隔的字符串视为一个术语。

  NGRAM 算法将字符串分成 n 元语法词。一个 n 元语法词就是由较大字符串的 n 个字符组成的子串。对于模糊（近似）匹配或不使用空格或非字母数字字符来分隔术语的文档，NGRAM 术语断开器是必需的。IN SYSTEM 表支持 NGRAM。

  由于 NGRAM 术语断开器是基于 TEXT 索引而生成的，因此，请使用文本配置对象设置来定义是使用 NGRAM 还是 GENERIC TEXT 索引。

  TERM BREAKER 可以包括使用 EXTERNAL NAME 和库入口点来指定外部术语断开器库。

- **library-and-entry-point-name-string** – *[operating-system:]function-name@library*
- **PREFILTER EXTERNAL NAME** – 指定 entry_point 以及由外部供应商提供的外部前置过滤器库的库名。
- **DROP PREFILTER** – 删除外部前置过滤器并将 `ISYSTEXTCONFIG` 表中的前置过滤器列设置为 NULL。

示例

(返回顶部) (第 67 页)

- **示例 1** – 创建文本配置对象 `maxTerm16`，然后将术语的最大长度更改为 16：

```
CREATE TEXT CONFIGURATION maxTerm16 FROM default_char;
ALTER TEXT CONFIGURATION maxTerm16 MAXIMUM TERM LENGTH 16;
```

- **示例 2** – 向 `maxTerm16` 配置对象添加非索引字表术语：

```
ALTER TEXT CONFIGURATION maxTerm16
STOPLIST 'because about therefore only';
```

- **示例 3** – 更新文本配置对象 `my_text_config`，以便在外部库 `mytermbreaker.dll` 中使用入口点 `my_term_breaker` 来断开文本：

```
CREATE TEXT CONFIGURATION my_text_config FROM default_char;
ALTER TEXT CONFIGURATION my_text_config
```

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

- **示例 4** – 更新文本配置对象 my_text_config，以便在外部库 myprefilter.dll 中使用入口点 my_prefilter 来前置过滤文档：

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

用法

(返回顶部) (第 67 页)

TEXT 索引依赖于文本配置对象。SAP Sybase IQ TEXT 索引使用立即刷新，而且无法截断；必须先删除索引，然后才能更改文本配置对象。要查看文本配置对象的设置，请查询 SYSTEXTCONFIG 系统视图。

副作用：

- 自动提交。

权限

(返回顶部) (第 67 页)

TERM BREAKER 或 **PREFILTER EXTERNAL NAME** 子句 - 需要 CREATE ANY EXTERNAL REFERENCE 系统特权，外加以下特权之一：

- ALTER ANY TEXT CONFIGURATION 系统特权。
- ALTER ANY OBJECT 系统特权。
- 您拥有文本配置对象。

所有其它子句 - 不管用户是否为配置对象的所有者，均需要 ALTER ANY TEXT CONFIGURATION 系统特权。

ALTER TEXT INDEX 语句

重命名、移动或更改 TEXT 索引的定义。

注意： 该语句需要非结构化数据分析 (IQ_UDA) 许可证。

快速链接：

转至参数 (第 70 页)

转至示例 (第 70 页)

转至用法 (第 70 页)

转至权限 (第 70 页)

语法

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

alter-clause - (back to Syntax)
    rename-object | move-object

rename-object - (back to alter-clause)
    RENAME { AS | TO } new-name

move-object - (back to alter-clause)
    MOVE TO dbspace-name
```

参数

(返回顶部) (第 69 页)

- **RENAME** – 重命名 TEXT 索引。
- **MOVE** – 将 TEXT 索引移动至指定的 dbspace。

示例

(返回顶部) (第 69 页)

- **示例** – 创建一个 TEXT 索引 `MyTextIndex` 并将其定义为 IMMEDIATE REFRESH，将该 TEXT 索引重命名为 `Text_index_daily`，然后将该 TEXT 索引移动至名为 `tispace` 的 dbspace：

```
CREATE TEXT INDEX MyTextIndex ON Customers ( CompanyName )
IMMEDIATE REFRESH;
ALTER TEXT INDEX MyTextIndex ON Customers RENAME AS
Text_index_daily;
ALTER TEXT INDEX Text_Index_Daily ON Customers MOVE TO tispace;
```

用法

(返回顶部) (第 69 页)

副作用：

- 自动提交。

权限

(返回顶部) (第 69 页)

move-object 子句 - 需要以下特权之一：

- ALTER ANY INDEX 系统特权。

- ALTER ANY OBJECT 系统特权。
- 对基础表的 REFERENCES 特权。
- 您拥有基础表。

rename-object 子句 - 需要以下特权之一：

- ALTER ANY INDEX 系统特权。
- ALTER ANY OBJECT 系统特权。
- MANAGE ANY DBSPACE。
- 以下特权之一：
  - 您拥有要建立索引的基础表。
  - 对表的 REFERENCES 特权，外加以下特权之一：
    - CREATE ANY OBJECT 系统特权。
    - 对目标 dbspace 的 CREATE 特权。

CREATE TEXT CONFIGURATION 语句

创建文本配置对象。

注意： 该语句需要非结构化数据分析 (IQ_UDA) 许可证。

快速链接：

转至参数 （第 71 页）

转至示例 （第 71 页）

转至用法 （第 72 页）

转至权限 （第 72 页）

语法

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

参数

（返回顶部） （第 71 页）

- **FROM** – 指定在创建新文本配置对象时将充当模板的文本配置对象的名称。缺省文本配置对象的名称是 DEFAULT_CHAR 和 DEFAULT_NCHAR。仅 SAP Sybase IQ 表支持 DEFAULT_CHAR；仅 SQL Anywhere 表支持 DEFAULT_NCHAR。

示例

（返回顶部） （第 71 页）

- **示例 1** – 使用 default_char 文本配置对象创建文本配置对象 max_term_sixteen，然后使用 **ALTER TEXT CONFIGURATION** 将 max_term_sixteen 的最大术语长度更改为 16：

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

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

用法

(返回顶部) (第 71 页)

将一个文本配置对象用作模板来创建另一个文本配置对象，然后根据需要使用 **ALTER TEXT CONFIGURATION** 语句改变选项。

要查看数据库中所有文本配置对象及其设置的列表，请查询 SYSTEXTCONFIG 系统视图。

副作用：

- 自动提交。

权限

(返回顶部) (第 71 页)

自身拥有的文本配置对象 -

- 需要 CREATE TEXT CONFIGURATION 系统特权。

由任何用户拥有的文本配置对象 - 需要具备以下特权之一：

- CREATE ANY TEXT CONFIGURATION 系统特权。
- CREATE ANY OBJECT 系统特权。

所有文本配置对象都具有 PUBLIC 访问权限。任何具有创建 **TEXT** 索引权限的用户都可以使用任意文本配置对象。

CREATE TEXT INDEX 语句

创建 TEXT 索引并指定要使用的文本配置对象。

注意： 该语句需要非结构化数据分析 (IQ_UDA) 许可证。

快速链接：

转至参数 (第 73 页)

转至示例 (第 73 页)

转至用法 (第 73 页)

转至权限 （第 73 页）

语法

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

参数

(返回顶部) （第 72 页）

- **ON** – 指定 TEXT 索引要构建于的表和列。
- **IN** – 指定 TEXT 索引所处的 dbspace。如果不指定此子句，就会在基础表所在的 dbspace 中创建 TEXT 索引。
- **CONFIGURATION** – 指定创建 TEXT 索引时要使用的文本配置对象。如果不指定此子句，就会使用 default_char 文本配置对象。
- **IMMEDIATE REFRESH** – （缺省）在每次基础表中的变化影响到 TEXT 索引中的数据时，刷新 TEXT 索引。SAP Sybase IQ 主存储库中的表的唯一允许值。IMMEDIATE REFRESH 子句一经创建便无法更改。

示例

(返回顶部) （第 72 页）

- **示例 1** – 使用 `max_term_sixteen` 文本配置对象，在 `iqdemo` 数据库中 `Customers` 表的 `CompanyName` 列上创建 TEXT 索引 `myTxtIdx`:

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

CONFIGURATION max_term_sixteen;
```

用法

(返回顶部) （第 72 页）

无法在视图、临时表或 IN SYSTEM 实例化视图中创建 TEXT 索引。**BEGIN PARALLEL IQ…END PARALLEL IQ** 语句不支持 **CREATE TEXT INDEX**。

副作用:

- 自动提交。

权限

(返回顶部) （第 72 页）

需要以下特权之一：

- CREATE ANY INDEX 系统特权，以及对索引创建于的 dbspace 的 CREATE 特权。
- CREATE ANY OBJECT 系统特权。

DROP TEXT CONFIGURATION 语句

删除文本配置对象。

注意： 该语句需要非结构化数据分析 (IQ_UDA) 许可证。

快速链接：

转至示例 （第 74 页）

转至用法 （第 74 页）

转至权限 （第 74 页）

语法

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

示例

（返回顶部） （第 74 页）

- **示例 1** – 创建和删除 `mytextconfig` 文本配置对象：

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

用法

（返回顶部） （第 74 页）

使用 **DROP TEXT CONFIGURATION** 删除文本配置对象。

试图删除具有相关 TEXT 索引的文本配置对象将产生错误。必须先删除相关 TEXT 索引，再删除文本配置对象。

文本配置对象存储在 `ISYSTEXTCONFIG` 系统表中。

副作用：

- 自动提交。

权限

（返回顶部） （第 74 页）

自身拥有的文本配置对象 – 无需任何权限。

由任何用户拥有的表配置对象 – 需要以下特权之一:

- DROP ANY TEXT CONFIGURATION 系统特权。
- DROP ANY OBJECT 系统特权。

DROP TEXT INDEX 语句

从数据库中删除 TEXT 索引。

注意: 该语句需要非结构化数据分析 (IQ_UDA) 许可证。

快速链接:

转至参数 (第 75 页)

转至示例 (第 75 页)

转至用法 (第 75 页)

转至权限 (第 76 页)

语法

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

参数

(返回顶部) (第 75 页)

- **ON** – 指定构建 TEXT 索引时所基于的表。

示例

(返回顶部) (第 75 页)

- **示例 1** – 创建和删除 `TextIdx` TEXT 索引:

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

用法

(返回顶部) (第 75 页)

必须先删除相关 TEXT 索引，然后才可以删除文本配置对象。

副作用:

- 自动提交。

权限

(返回顶部) (第 75 页)

需要以下特权之一:

- DROP ANY INDEX 系统特权。
- DROP ANY OBJECT 系统特权。
- 对要建立索引的表的 REFERENCES 特权。
- 您拥有基础表。

函数支持

了解支持 LONG BINARY 和 LONG VARCHAR 数据类型的函数。

大对象数据的函数支持摘要

对 LONG BINARY (BLOB) 和 LONG VARCHAR (CLOB) 数据类型及 LONG BINARY 和 LONG VARCHAR 变量的函数支持进行汇总。

除下表中列出的函数外，还可使用 **BFILE** 函数来提取 LOB 数据。

标量和集合用户定义函数支持使用大对象数据类型作为输入参数。

表 18. LOB 数据类型和变量的函数支持

函数	是否支持 BLOB 数据？	是否支持 BLOB 变量？	是否支持 CLOB 数据？	是否支持 CLOB 变量？
BIT_LENGTH()	是	是	是	是
BYTE_LENGTH()	是*	是*	是*	是*
BYTE_LENGTH64()	是	是	是	是
BYTE_SUBSTR()	是	是	是	是
BYTE_SUBSTR64()	是	是	是	是
CHAR_LENGTH()	否	否	是	是
CHAR_LENGTH64()	否	否	是	是
CHARINDEX()	是	是	是	是
LOCATE()	是	是	是	是
OCTET_LENGTH()	是	是	是	是
PATINDEX()	否	否	是	是
SUBSTR() / SUBSTRING()	否	否	是	是
SUBSTRING64()	是	是	是	是

*只有在查询返回小于 2GB 时，**BYTE_LENGTH** 函数才支持 LONG BINARY 列和变量以及 LONG VARCHAR 列和变量。如果返回的 LONG BINARY 或 LONG VARCHAR 数据的字节长度大于 2GB，**BYTE_LENGTH** 就会返回错误，提示您必须使用 **BYTE_LENGTH64** 函数。

另请参见

- 大对象列的用户定义的函数支持 （第 86 页）
- 大对象数据导出 （第 55 页）

BIT_LENGTH 函数

BIT_LENGTH 函数返回无符号的 64 位值，该值包含大对象列或变量参数的位长度。如果参数为空值，则 **BIT_LENGTH** 返回空值。

语法

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

参数

large-object-column - LONG VARCHAR 或 LONG BINARY 列或变量的名称。

用法

BIT_LENGTH 支持所有的 SAP Sybase IQ 数据类型以及任意数据大小的 LONG BINARY 与 LONG VARCHAR 变量。目前，一个 SQL 变量可以容纳长度最大为 2GB - 1 的数据。

BYTE_LENGTH 函数 [字符串]

返回字符串中的字节数。

语法

```
BYTE_LENGTH ( string-expression )
```

参数

参数	描述
string-expression	要计算其长度的字符串。

返回

INT

注释

返回的长度中包括尾随空格字符。

空字符串的返回值为空值。

如果字符串使用的是多字节字符集，则 **BYTE_LENGTH** 值不同于 **CHAR_LENGTH** 返回的字符数。

如果您有权使用非结构化数据分析功能，则可以将此函数用于大对象数据：

- 只有在查询返回小于 2GB 时，**BYTE_LENGTH** 函数才支持 LONG BINARY 列和变量以及 LONG VARCHAR 列和变量。如果返回的 LONG BINARY 或 LONG VARCHAR 数据的字节长度大于或等于 2GB，**BYTE_LENGTH** 就会返回错误，提示您必须使用 **BYTE_LENGTH64** 函数。

标准和兼容性

- SQL - ISO/ANSI SQL 语法的服务商扩充。
- SAP Sybase - 不受 Adaptive Server 支持。

示例
返回值 12：

```
SELECT BYTE_LENGTH( 'Test Message' ) FROM iq_dummy
```

BYTE_LENGTH64 函数 [String]

BYTE_LENGTH64 函数返回无符号的 64 位值，该值包含大对象列或变量参数的字节长度。

语法

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

参数
large-object-column - LONG VARCHAR 或 LONG BINARY 列或变量的名称。

用法
BYTE_LENGTH64 函数支持 LONG BINARY 和 LONG VARCHAR 列以及任意数据大小的 LONG BINARY 和 LONG VARCHAR 变量。目前，一个 SQL 变量可以容纳长度最大为 2GB - 1 的数据。

BYTE_SUBSTR64 和 BYTE_SUBSTR 函数 [String]

BYTE_SUBSTR64 和 **BYTE_SUBSTR** 函数返回大对象列或变量参数的字节子字符串。

语法

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

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

参数
large-object-column - LONG VARCHAR 或 LONG BINARY 列或变量的名称。

start - 表示子字符串开始位置的整数表达式。正整数表示从字符串开始处开始，第一个字节位于位置 1。负整数表示子串从字符串结尾处开始，最后一个字节位于位置 -1。

length - 表示子字符串长度的整数表达式。正的 length 表示要返回的字节数且以 *start* 位置为起始位置。负的 length 表示要返回的字节数且以 *start* 位置为结束位置。

用法

- 函数 **BYTE_LENGTH64**、**BYTE_SUBSTR64** 和 **BYTE_SUBSTR** 的嵌套运算不支持大对象列或变量。
- **BYTE_SUBSTR64** 和 **BYTE_SUBSTR** 函数支持 LONG BINARY 和 LONG VARCHAR 列以及任意数据大小的 LONG BINARY 和 LONG VARCHAR 变量。目前，一个 SQL 变量可以容纳长度最大为 2GB - 1 的数据。

CHAR_LENGTH 函数 [String]

CHAR_LENGTH 函数返回带符号的 32 位值，该值包含 LONG VARCHAR 列或变量参数的字符长度（包括尾随空白）。

语法

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

参数

- **long-varchar-object** – LONG VARCHAR 列或 LONG VARCHAR 变量的名称。

用法

- **CHAR_LENGTH** 支持 LONG VARCHAR 列和任意数据大小的 LONG VARCHAR 变量。目前，一个 SQL 变量可以容纳长度最大为 2GB - 1 的数据。
- 如果参数为 NULL，则 **CHAR_LENGTH** 返回 NULL。
- 如果字符长度超过 2GB - 1 (2147483647)，则返回错误。

CHAR_LENGTH64 函数 [String]

CHAR_LENGTH64 函数返回不带符号的 64 位值，该值包含 LONG VARCHAR 列或变量参数的字符长度（包括尾随空白）。

语法

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

参数

long-varchar-object - 表中 LONG VARCHAR 列的名称，或 LONG VARCHAR 变量的名称。

用法

- **CHAR_LENGTH64** 支持 LONG VARCHAR 列和任意数据大小的 LONG VARCHAR 变量。目前，一个 SQL 变量可以容纳长度最大为 2GB - 1 的数据。
- 如果参数为 NULL，则 **CHAR_LENGTH64** 返回 NULL。

CHARINDEX 函数 [String]

CHARINDEX 函数返回带符号的 64 位整数，其中包含指定字符串在大对象列或变量参数中第一次出现的位置。对于 CHAR 和 VARCHAR 列，**CHARINDEX** 返回带符号的 32 位整数位置。

语法

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

参数

string-expression - 您要搜索的最大长度为 255 字节的字符串。

large-object-column - LONG VARCHAR 或 LONG BINARY 列或变量的名称。

用法

- **CHARINDEX** 函数中返回或指定的所有位置或偏移始终为字符偏移，并且可能不同于多字节数据的字节偏移。
- 如果搜索的大对象单元包含多个 *string-expression* 实例，**CHARINDEX** 仅返回第一个实例的位置。
- 如果列不包含字符串，**CHARINDEX** 函数返回零 (0)。
- 搜索长度超过 255 字节的字符串将返回 NULL。
- 搜索零长度的字符串将返回 1。
- 如果任一参数为 NULL，则结果为 NULL。
- **CHARINDEX** 支持搜索 LONG VARCHAR 和 LONG BINARY 列以及任意数据大小的 LONG VARCHAR 和 LONG BINARY 变量。目前，一个 SQL 变量可以容纳长度最大为 2GB - 1 的数据。

LOCATE 函数 [String]

LOCATE 函数返回带符号的 64 位整数，它包含指定字符串在大对象列或变量参数中的位置。对于 CHAR 和 VARCHAR 列，**LOCATE** 返回带符号的 32 位整数位置。

语法

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

参数

large-object-column – 要搜索的 LONG VARCHAR 或 LONG BINARY 列或变量的名称。

string-expression - 您要搜索的最大长度为 255 字节的字符串。

numeric-expression - 字符串中作为搜索起始位置的字符位置或偏移。对于 LONG VARCHAR 和 LONG BINARY 列，*numeric-expression* 为带符号的 64 位整数；对于 CHAR、VARCHAR 和 BINARY 列，该参数为带符号的 32 位整数。第一个字符为位置 1。如果起始偏移是负值，**LOCATE** 返回最后一个匹配字符串偏移，而不是第一个。负的偏移指示从搜索中排除字符串尾的多长一部分。排除的字符数计算公式为（-1 * 偏移）- 1。

用法

- **LOCATE** 函数中返回或指定的所有位置或偏移始终为字符偏移，并且可能不同于多字节数据的字节偏移。
- 如果搜索的大对象单元包含多个字符串实例，则会出现下列情况：
  - 如果指定了 *numeric-expression*，则 **LOCATE** 从字符串中的该偏移处开始搜索。
  - 如果未指定 *numeric-expression*，则 **LOCATE** 仅返回首个实例的位置。
- 如果列不包含字符串，**LOCATE** 将返回零 (0)。
- 搜索长度超过 255 字节的字符串将返回 NULL。
- 搜索零长度的字符串将返回 1。
- 如果任一参数为 NULL，则结果为 NULL。
- **LOCATE** 支持搜索 LONG VARCHAR 和 LONG BINARY 列以及任意数据大小的 LONG VARCHAR 和 LONG BINARY 变量。目前，一个 SQL 变量可以容纳长度最大为 2GB - 1 的数据。

OCTET_LENGTH 函数 [String]

OCTET_LENGTH 函数返回无符号的 64 位值，该值包含大对象列或变量参数的字节长度。

语法

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

参数

large-object-column - LONG VARCHAR 或 LONG BINARY 列或变量的名称。

用法

- 如果参数为 NULL，则 **OCTET_LENGTH** 返回 NULL。
- **OCTET_LENGTH** 支持所有的 SAP Sybase IQ 数据类型以及任意数据大小的 LONG VARCHAR 和 LONG BINARY 变量。目前，一个 SQL 变量可以容纳长度最大为 2GB - 1 的数据。

PATINDEX 函数 [String]

PATINDEX 函数返回不带符号的 64 位整数，它包含指定模式在 LONG VARCHAR 列或变量中第一次出现的位置。对于 CHAR 和 VARCHAR 列，**PATINDEX** 返回带符号的 32 位整数位置。

语法

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

参数

- **pattern** – 要搜索的模式。对于具有通配符的模式，此字符串限制为 126 字节。如果省略前导百分比通配符，当模式出现在列值开头时，**PATINDEX** 会返回一 (1)，当模式没有出现在列值开头时，则会返回零 (0)。类似地，如果省略尾随百分比通配符，模式应出现在列值末尾。模式与 **LIKE** 比较运算使用相同的通配符。

  不带通配符（百分比 (%) 和下划线 (_) ）的模式的长度最大可达 255 字节。
- **long-varchar-column** – LONG VARCHAR 列或变量的名称。

用法

- **PATINDEX** 函数中返回或指定的所有位置或偏移始终为字符偏移，并且可能不同于多字节数据的字节偏移。

- 如果搜索的 LONG VARCHAR 单元包含多个字符串模式实例，**PATINDEX** 仅返回第一个实例的位置。
- 如果列不包含模式，**PATINDEX** 将返回零 (0)。
- 搜索长度超过 126 字节的模式将返回 NULL。
- 搜索零长度的模式将返回 1。
- 如果任一参数为 NULL，则结果为零 (0)。
- **PATINDEX** 支持任意数据大小的 LONG VARCHAR 变量。目前，SQL 变量可以容纳的最大长度为 2GB - 1。**PATINDEX** 不支持 LONG BINARY 变量或搜索 LONG BINARY 列。

SUBSTRING 函数 [String]

SUBSTRING 函数返回 LONG VARCHAR 列或变量参数的可变长度字符串。如果任一参数为 NULL，则 **SUBSTRING** 返回 NULL。

语法

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

参数

long-varchar-column - LONG VARCHAR 列或变量的名称。

start - 表示子字符串开始位置的整数表达式。正整数表示从字符串开始处开始，第一个字符位于位置 1。负整数表示子串从字符串结尾处开始，最后一个字符位于位置 -1。

length - 表示子字符串字符长度的整数表达式。正的 length 表示要返回的字符数且以 *start* 位置为起始位置。负的 length 表示要返回的字符数且以 *start* 位置为结束位置。

用法

SUBSTRING 支持任意数据大小的 LONG VARCHAR 变量。目前，SQL 变量可以容纳的最大长度为 2GB - 1。**SUBSTRING** 不支持 LONG BINARY 变量或搜索 LONG BINARY 列。

ansi_substring 数据库选项设置为 ON（缺省）时，负值无效。

ANSI_SUBSTRING 选项 [TSQL]

控制在为 start 或 length 参数提供负值时 SUBSTRING (SUBSTR) 函数的行为。

允许值

ON、OFF

默认值

ON

范围
可在数据库 (PUBLIC) 或用户级别设置选项。在数据库级别进行设置时，值将变为任何新用户的缺省值，但不会对现有用户产生任何影响。在用户级别进行设置时，仅替换该用户的 PUBLIC 值。为自身设置选项无需任何系统特权。在数据库或用户级别为任何其他用户设置选项都需要系统特权。

必须具有 SET ANY PUBLIC OPTION 系统特权才能设置此选项。可针对个别连接或 PUBLIC 角色进行临时设置。设置立即生效。

注释
ANSI_SUBSTRING 选项设置为 ON 时，**SUBSTRING** 函数的行为相当于 ANSI/ISO SQL/2003 行为。如果起始偏移为负或零，则视为字符串左侧用非字符填补，并且在提供负值长度的情况下会出现错误。

当此选项设置为 OFF 时，**SUBSTRING** 函数的行为与先前版本的 SAP Sybase IQ 的行为相同：负的起始偏移表示从字符串末尾开始的偏移，而负的长度表示所需的子串在起始偏移左侧第 N 个字符处结束，其中 N 为长度。起始偏移 0 等效于起始偏移 1。

避免在 **SUBSTRING** 函数中使用非正起始偏移量或负长度。请尽可能改用 **LEFT** 或 **RIGHT** 函数。

示例
以下示例说明了基于不同 **ANSI_SUBSTRING** 选项设置的不同 **SUBSTRING** 函数返回值：

```
SUBSTRING( 'abcdefgh',-2,4 );
    ansi_substring = Off ==> 'gh'
    // substring starts at second-last character
    ansi_substring = On  ==> 'gh'
    // takes the first 4 characters of
    // ???abcdefgh and discards all ?
```

```
SUBSTRING( 'abcdefgh',4,-2 );
    ansi_substring = Off ==> 'cd'
    ansi_substring = On  ==> value -2 out of range
    for destination
```

```
SUBSTRING( 'abcdefgh',0,4 );
    ansi_substring = Off ==> 'abcd'
    ansi_substring = On  ==> 'abcd'
```

SUBSTRING64 函数 [String]

SUBSTRING64 函数返回大对象列或变量参数的可变长度字符串。

语法

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

参数

large-object-column - LONG VARCHAR 或 LONG BINARY 列或变量的名称。

start - 表示子字符串开始位置的 8 字节整数。**SUBSTRING64** 将值为负或零的 *start* 偏移量解释为用“非字符”对字符串左侧进行了填充。第一个字符从位置 1 开始。

length - 表示子字符串长度的 8 字节整数。如果 *length* 为负，则返回错误。

示例

SUBSTRING64 返回的值（假定有一个名为 col1 的列，它包含字符串 ["ABCDEFG"]）：

SUBSTRING64(col1, 2, 4) 返回字符串 "BCDE"

SUBSTRING64(col1, 1, 3) 返回字符串 "ABC"

SUBSTRING64(col1, 0, 3) 返回字符串 "AB"

SUBSTRING64(col1, -1, 3) 返回字符串 "A"

用法

- 如果任一参数为 NULL，则 **SUBSTRING64** 返回 NULL。
- 函数 **SUBSTRING64**、**SUBSTRING**、**SUBSTR**、**BYTE_SUBSTR** 和 **BYTE_SUBSTR64** 的嵌套运算不支持大对象列或变量。
- **SUBSTRING64** 支持搜索 LONG VARCHAR 和 LONG BINARY 列以及任意数据大小的 LONG VARCHAR 和 LONG BINARY 变量。目前，一个 SQL 变量可以容纳长度最大为 2GB - 1 的数据。

大对象列的集合函数支持

只有集合函数 **COUNT (*)** 在 LONG BINARY 和 LONG VARCHAR 列中受到支持。

不支持 **COUNT DISTINCT** 参数。如果 LONG BINARY 或 LONG VARCHAR 列与 **MIN**、**MAX**、**AVG** 或 **SUM** 集合函数一起使用，则会返回错误。

大对象列的用户定义的函数支持

标量和集合用户定义函数支持最大 4GB（千兆字节）的大对象数据类型 LONG VARCHAR (CLOB) 和 LONG BINARY (BLOB) 作为输入参数。不支持 LOB 数据类型作为输出参数。

您必须获得专门许可才能使用用户定义的函数支持功能。

索引

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

S

T

W

X

Y

Z