

Tibero

TEXT 참조 안내서

Tibero 7



Copyright © 2022 TmaxTibero Co., Ltd. All Rights Reserved.

Copyright Notice

Copyright © 2022 TmaxTibero Co., Ltd. All Rights Reserved.

대한민국 경기도 성남시 분당구 황새울로258번길 29, BS 타워 9층 우)13595

Website

<http://www.tmaxtibero.com>

기술서비스센터

Tel : +82-1544-8629

E-Mail : info@tmax.co.kr

Restricted Rights Legend

All TmaxTibero Software (Tibero®) and documents are protected by copyright laws and international convention. TmaxTibero software and documents are made available under the terms of the TmaxTibero License Agreement and this document may only be distributed or copied in accordance with the terms of this agreement. No part of this document may be transmitted, copied, deployed, or reproduced in any form or by any means, electronic, mechanical, or optical, without the prior written consent of TmaxTibero Co., Ltd. Nothing in this software document and agreement constitutes a transfer of intellectual property rights regardless of whether or not such rights are registered) or any rights to TmaxTibero trademarks, logos, or any other brand features.

This document is for information purposes only. The company assumes no direct or indirect responsibilities for the contents of this document, and does not guarantee that the information contained in this document satisfies certain legal or commercial conditions. The information contained in this document is subject to change without prior notice due to product upgrades or updates. The company assumes no liability for any errors in this document.

이 소프트웨어(Tibero®) 사용설명서의 내용과 프로그램은 저작권법과 국제 조약에 의해서 보호받고 있습니다. 사용설명서의 내용과 여기에 설명된 프로그램은 TmaxTibero Co., Ltd.와의 사용권 계약 하에서만 사용이 가능하며, 사용설명서는 사용권 계약의 범위 내에서만 배포 또는 복제할 수 있습니다. 이 사용설명서의 전부 또는 일부분을 TmaxTibero의 사전 서면 동의 없이 전자, 기계, 녹음 등의 수단을 사용하여 전송, 복제, 배포, 2차적 저작물작성 등의 행위를 하여서는 안 됩니다.

이 소프트웨어 사용설명서와 프로그램의 사용권 계약은 어떠한 경우에도 사용설명서 및 프로그램과 관련된 지적 재산권(등록 여부를 불문)을 양도하는 것으로 해석되지 아니하며, 브랜드나 로고, 상표 등을 사용할 권한을 부여하지 않습니다. 사용설명서는 오로지 정보의 제공만을 목적으로 하고, 이로 인한 계약상의 직접적 또는 간접적 책임을 지지 아니하며, 사용설명서 상의 내용은 법적 또는 상업적인 특정한 조건을 만족시키는 것을 보장하지는 않습니다. 사용설명서의 내용은 제품의 업그레이드나 수정에 따라 그 내용이 예고 없이 변경될 수 있으며, 내용상의 오류가 없음을 보장하지 아니합니다.

Trademarks

Tibero® is a registered trademark of TmaxTibero Co., Ltd. Other products, titles or services may be registered trademarks of their respective companies.

Tibero®는 TmaxTibero Co., Ltd.의 등록 상표입니다. 기타 모든 제품들과 회사 이름은 각각 해당 소유주의 상표로서 참조용으로만 사용됩니다.

Open Source Software Notice

Some modules or files of this product are subject to the terms of the following licenses. : OpenSSL, RSA Data Security, Inc., Apache Foundation, Jean-loup Gailly and Mark Adler, Paul Hsieh's hash

Detailed Information related to the license can be found in the following directory : `${INSTALL_PATH}/license/oss_licenses`

본 제품의 일부 파일 또는 모듈은 다음의 라이선스를 준수합니다. : OpenSSL, RSA Data Security, Inc., Apache Foundation, Jean-loup Gailly and Mark Adler, Paul Hsieh's hash

관련 상세한 정보는 제품의 다음의 디렉터리에 기재된 사항을 참고해 주십시오. : `${INSTALL_PATH}/license/oss_licenses`

안내서 정보

안내서 제목: Tibero TEXT 참조 안내서

발행일: 2024-08-22

소프트웨어 버전: Tibero 7.2.2

안내서 버전: v7.2.2

내용 목차

안내서에 대하여	vii
제1장 TEXT 인덱스	1
1.1. 개요	1
1.2. CTXCAT 인덱스	1
1.2.1. 인덱스 생성	1
1.2.2. 인덱스 제거	2
1.2.3. 인덱스 관련 객체	2
1.2.4. 인덱스 제약	3
1.3. CONTEXT 인덱스	3
1.3.1. 인덱스 생성	3
1.3.2. 인덱스 제거	5
1.3.3. 인덱스 관련 객체	5
1.3.4. 인덱스 제약	5
제2장 TEXT 질의	7
2.1. CATSEARCH	7
2.2. CONTAINS	8
제3장 TEXT Indexing Elements	13
3.1. 개요	13
3.2. STORAGE	13
3.3. LEXER	14
3.3.1. BASIC_LEXER	14
3.3.2. JAPANESE_LEXER	14
3.3.3. CHINESE_LEXER	15
3.3.4. CHINESE_VGRAM_LEXER	15
3.3.5. WORLD_LEXER	15
3.4. WORDLIST	16
3.5. STOPLIST	16
3.6. DATASTORE	16
3.6.1. DIRECT_DATASTORE	16
3.6.2. MULTI_COLUMN_DATASTORE	16
3.7. FILTER	17
3.7.1. NULL_FILTER	17
3.8. SECTION GROUP	17
3.9. TEXT_DDL 패키지	17
3.9.1. ADD_SPECIAL_SECTION	18
3.9.2. ADD_STOPWORD	18
3.9.3. CREATE_PREFERENCE	19
3.9.4. CREATE_SECTION_GROUP	19
3.9.5. CREATE_STOPLIST	20

3.9.6.	DROP_PREFERENCE	20
3.9.7.	DROP_SECTION_GROUP	21
3.9.8.	DROP_STOPLIST	21
3.9.9.	REMOVE_SECTION	22
3.9.10.	REMOVE_STOPWORD	22
3.9.11.	SET_ATTRIBUTE	23
3.10.	CTX_DDL 패키지	23
3.10.1.	OPTIMIZE_INDEX	23
3.10.2.	SYNC_INDEX	24
3.11.	CTX_OUTPUT 패키지	25
3.11.1.	END_LOG	25
3.11.2.	START_LOG	25
Appendix A.	STOPWORDS	27
A.1.	영어	27
색인	29

안내서에 대하여

안내서의 대상

본 안내서는 Tibero[®](이하 Tibero)를 사용하여 TEXT 애플리케이션 프로그램 작성하는 개발자나 TEXT 시스템을 유지하는 관리자를 대상으로 기술한다.

안내서의 전제 조건

본 안내서는 Tibero에서 TEXT 기능을 사용하고 유지하는 데 필요한 SQL 문장을 설명하는 안내서이다. 따라서 본 안내서를 원활히 이해하기 위해서는 다음과 같은 사항을 미리 알고 있어야 한다.

- 데이터베이스의 이해
- RDBMS의 이해
- SQL의 이해

안내서의 제한 조건

본 안내서는 Tibero를 실무에 적용하거나 운용하는 데 필요한 모든 사항을 포함하고 있지 않다. 따라서 설치, 환경설정 등 운용 및 관리에 대해서는 각 제품 안내서를 참고하기 바란다.

참고

Tibero의 설치 및 환경설정에 관한 내용은 "Tibero 설치 안내서"를 참고한다.

안내서 구성

Tibero TEXT 참조 안내서는 총 3개의 장과 Appendix로 구성되어 있다.

각 장의 주요 내용은 다음과 같다.

- 제1장: TEXT 인덱스

TEXT 인덱스를 생성/유지관리하는 방법에 대하여 기술한다.

- 제2장: TEXT 질의

Tibero TEXT 기능을 이용한 질의에 대해 기술한다.

- 제3장: TEXT Indexing Elements

Tibero TEXT 인덱스와 관련된 Elements에 대해 기술한다.

- Appendix A: STOPWORDS

Tibero TEXT에서 기본으로 제공되는 STOPWORD에 대하여 기술한다.

안내서 규약

표기	의미
<<AaBbCc123>>	프로그램 소스 코드의 파일명
<Ctrl>+C	Ctrl과 C를 동시에 누름
[Button]	GUI의 버튼 또는 메뉴 이름
진하게	강조
" "(따옴표)	다른 관련 안내서 또는 안내서 내의 다른 장 및 절 언급
'입력항목'	화면 UI에서 입력 항목에 대한 설명
하이퍼링크	메일 계정, 웹 사이트
>	메뉴의 진행 순서
+----	하위 디렉터리 또는 파일 있음
----	하위 디렉터리 또는 파일 없음
<u>참고</u>	참고 또는 주의사항
<u>주의</u>	주의할 사항
[그림 1.1]	그림 이름
[예 1.1]	예제 이름
AaBbCc123	Java 코드, XML 문서
[<i>command argument</i>]	옵션 파라미터
< xyz >	'<'와 '>' 사이의 내용이 실제 값으로 변경됨
	선택 사항. 예) A B: A나 B 중 하나
...	파라미터 등이 반복되어서 나옴
\${ }	환경변수

시스템 사용 환경

	요구 사항
Platform	HP-UX 11i v3(11.31)
	Solaris (Solaris 11)
	AIX (AIX 7.1/AIX 7.2/AIX 7.3)
	GNU (X86, 64, IA64)
	Red Hat Enterprise Linux 7 kernel 3.10.0 이상
	Windows(x86) 64bit
Hardware	최소 2.5GB 하드디스크 공간
	1GB 이상 메모리 공간
Compiler	PSM (C99 지원 필요)
	tbESQL/C (C99 지원 필요)

관련 안내서

안내서	설명
Tibero 설치 안내서	설치 시 필요한 시스템 요구사항과 설치 및 제거 방법을 기술한 안내서이다.
Tibero 관리자 안내서	Tibero의 동작과 주요 기능의 원활한 수행을 보장하기 위해 DBA가 알아야 할 관리 방법을 논리적 또는 물리적 측면에서 설명하고, 관리를 지원하는 각종 도구를 기술한 안내서이다.
Tibero 애플리케이션 개발자 안내서	각종 애플리케이션 라이브러리를 이용하여 애플리케이션 프로그램을 개발하는 방법을 기술한 안내서이다.
Tibero tbCLI 안내서	Call Level Interface인 tbCLI의 개념과 구성요소, 프로그램 구조를 소개하고 tbCLI 프로그램을 작성하는 데 필요한 데이터 타입, 함수, 에러 메시지를 기술한 안내서이다.
Tibero External Procedure 안내서	External Procedure를 소개하고 이를 생성하고 사용하는 방법을 기술한 안내서이다.
Tibero tbESQL/C 안내서	C 프로그래밍 언어를 사용해 데이터베이스 작업을 수행하는 각종 애플리케이션 프로그램을 작성하는 방법을 기술한 안내서이다.
Tibero tbPSM 안내서	저장 프러시저 모듈인 tbPSM의 개념과 문법, 구성요소를 소개하고, 프로그램을 작성하는 데 필요한 제어 구조, 복합 타입, 서브 프로그램, 패키지 및 SQL 문장을 실행하고 에러를 처리하는 방법을 기술한 안내서이다.
Tibero tbPSM 참조 안내서	저장 프러시저 모듈인 tbPSM의 패키지를 소개하고, 이러한 패키지에 포함된 각 프러시저와 함수의 프로토타입, 파라미터, 예제 등을 기술한 참조 안내서이다.
Tibero 유틸리티 안내서	데이터베이스와 관련된 작업을 수행하기 위해 필요한 유틸리티의 설치 및 환경설정, 사용 방법을 기술한 안내서이다.
Tibero TAS 안내서	Tibero Active Storage(TAS)를 사용해서 Tibero의 파일을 관리하고자 하는 관리자를 대상으로 기술한 안내서이다.
Tibero 에러 안내서	Tibero를 사용하는 도중에 발생할 수 있는 각종 에러의 원인과 해결 방법을 기술한 안내서이다.
Tibero	Tibero에서 Geometry 타입에 대한 설명과 Spatial 기능 관련 프러시저 함수 목록 및 사용 방법 등을 기술한 안내서이다.

안내서	설명
Spatial 참조 안내서	
Tibero TDP.NET 안내서	Tibero Data Provider for .NET 기능을 기술하는 안내서이다.
Tibero IMCS 안내서	Tibero에서 제공하는 In-Memory Column Store(이하 IMCS) 기능을 기술하는 안내서이다.

제1장 TEXT 인덱스

Tibero 본 장에서는 CTXCAT 인덱스와 CONTEXT 인덱스의 생성/삭제 방법과 인덱스와 관련된 객체에 대해 설명한다.

1.1. 개요

Tibero TEXT는 텍스트로 구성된 데이터에 대한 질의 성능을 향상시킨 기능이다.

Tibero TEXT에서는 책 제목, 온라인 쇼핑몰의 상품명 등 비교적 간단한 정보를 담고 있는 컬럼에서 검색어 질의를 빠르게 처리하기 위한 **CTXCAT 인덱스**와 책 본문과 같은 큰 일관성있는 문서를 인덱싱하는 **CONTEXT 인덱스**를 제공하고 있다.

1.2. CTXCAT 인덱스

본 절에서는 CTXCAT 인덱스의 생성/제거, 관련 객체 제약 사항에 대해서 설명한다.

1.2.1. 인덱스 생성

CTXCAT 인덱스 생성 방법은 다음과 같다.

- 사용법

```
CREATE INDEX idx_name on [sch_name.]tbl_name (col_name) CTXCATINDEX  
[PARAMETERS('preferences_class_name user_defined_preference_name')]
```

항목	설명
<i>idx_name</i>	생성할 CTXCAT 인덱스의 이름이다.
<i>sch_name</i>	인덱스를 생성할 대상 테이블의 소유자를 설정한다.
<i>tbl_name</i>	인덱스를 생성할 대상 테이블을 설정한다.
<i>col_name</i>	인덱스를 생성할 대상 column VARCHAR2 TYPE이어야 한다.
CTXCATINDEX	CTXCAT 인덱스로 생성함을 의미하는 예약어이다.
parameters	CTXCAT 인덱스를 생성할 때 사용할 설정을 지정하는 예약어이다.
preferences_class_name	CTXCAT 인덱스를 생성할 때 설정을 지정할 클래스이다.
user_defined_preference_name	위에서 지정된 CLASS에 대해 생성한 PREFERENCE이다.

참고

PREFERENCE, CLASS 등 인덱스의 설정과 관련된 자세한 내용은 “[제3장 TEXT Indexing Elements](#)”를 참고한다.

- 예제

```
SQL>create table t(name varchar(400));

Table 'T' created.

SQL>create index t_idx on t(name) ctxcatindex;

Index 'T_IDX' created.
```

주의

TEXT 인덱스 관련 쿼리를 실행할 때 내부적으로 'TEXT'라는 이름의 패키지를 사용하기 때문에 테이블 이름을 'TEXT'로 지정하지 않는다.

1.2.2. 인덱스 제거

CTXCAT 인덱스, CONTEXT 인덱스를 제거하기 위한 SQL은 다른 인덱스의 경우와 동일하다.

- 사용법

```
DROP INDEX index_name
```

항목	설명
index_name	삭제 할 TEXT 인덱스의 이름이다.

- 예제

```
SQL>drop index t_idx;

Index 'T_IDX' dropped.
```

1.2.3. 인덱스 관련 객체

CTXCAT 인덱스를 생성할 때 다음과 같은 객체가 생성된다.

객체 이름	객체 타입	설명
DR\$index_name\$I	테이블	컬럼 텍스트의 토큰이 저장되는 테이블이다.
DR\$index_name\$R	인덱스	토큰 테이블의 rowid 컬럼에 걸리는 인덱스이다.
DR\$index_name\$X	인덱스	토큰 테이블의 토큰 컬럼과 rowid 컬럼에 걸리는 인덱스이다.
DR\$index_name_TRG_I	트리거	CTXCAT 인덱스가 걸린 컬럼에 데이터가 입력될 때 입력 데이터를 토큰 테이블에 입력하기 위한 트리거이다.
DR\$index_name_TRG_D	트리거	CTXCAT 인덱스가 걸린 컬럼에 데이터가 삭제될 때 해당 토큰을 삭제하기 위한 트리거이다.
DR\$index_name_TRG_U	트리거	CTXCAT 인덱스가 걸린 컬럼에 데이터가 수정될 때 해당 토큰을 수정하기 위한 트리거이다.

1.2.4. 인덱스 제약

CTXCAT 인덱스는 다음과 같은 경우에 제약을 가진다.

- 여러 컬럼에 대해 하나의 CTXCAT 인덱스를 걸 수 없다.
- TEXT 인덱스는 파티션이 될 수 없다.
- TEXT 인덱스는 VARCHAR2 타입인 컬럼에 대해서만 생성할 수 있다.

1.3. CONTEXT 인덱스

본 절에서는 CONTEXT 인덱스의 생성/제거, 관련 객체 제약 사항에 대해서 설명한다.

1.3.1. 인덱스 생성

CONTEXT 인덱스 생성 방법은 다음과 같다.

- 사용법

```
CREATE INDEX idx_name on [sch_name.]tbl_name INDEXTYPE IS CTXSYS.CONTEXT
parameters('preferences_class_name user_defined_preference_name sync(option)')
```

항목	설명
idx_name	생성할 CONTEXT 인덱스의 이름이다.
sch_name	인덱스를 생성할 대상 테이블의 소유자를 설정한다.
tbl_name	인덱스를 생성할 대상 테이블을 설정한다.
CTXSYS.CONTEXT	CONTEXT 인덱스로 생성함을 의미하는 예약어이다.

항목	설명
parameters	CONTEXT 인덱스를 생성할 때 사용할 설정을 지정하는 예약어이다.
preferences_class_name	CONTEXT 인덱스를 생성할 때 설정을 지정할 클래스이다.
user_defined_preference_name	위에서 지정된 CLASS에 대해 생성한 PREFERENCE이다.
sync(option)	CONTEXT 인덱스의 자동 동기화를 위해 사용 된다. 다음과 같이 on commit, every 2가지 방법이 있다. <ul style="list-style-type: none"> - ON COMMIT : 커밋 후 인덱스를 동기화한다. 동기화가 끝날 때 까지 커밋이 리턴되지 않는다. - EVERY : interval-string 값으로 지정된 일정한 간격으로 인덱스를 자동으로 동기화 한다. EVERY를 사용하는 동기화는 작성자가 create job 특권이 있어야 한다.

참고

PREFERENCE, CLASS 등 인덱스의 설정과 관련된 자세한 내용은 [“제3장 TEXT Indexing Elements”](#)를 참고한다.

● 예제

```
SQL>create table t(name varchar(400), value varchar(1000));

Table 'T' created.

SQL>create index t_idx on t(name) indextype is ctxsys.context
parameters('sync(on commit)');

Index 'T_IDX' created.

SQL>create index t_idx2 on t(value) indextype is ctxsys.context
parameters('sync(every sysdate+1/24/60)');

Index 'T_IDX2' created.
```

주의

TEXT 인덱스 관련 쿼리를 실행할 때 내부적으로 'TEXT'라는 이름의 패키지를 사용하기 때문에 테이블 이름을 'TEXT'로 지정하지 않는다.

1.3.2. 인덱스 제거

CTXCAT 인덱스, CONTEXT 인덱스를 제거하기 위한 SQL은 다른 인덱스의 경우와 동일하다.

- 사용법

```
DROP INDEX index_name
```

항목	설명
<i>index_name</i>	삭제할 TEXT 인덱스의 이름이다.

- 예제

```
SQL>drop index t_idx;
```

```
Index 'T_IDX' dropped.
```

1.3.3. 인덱스 관련 객체

CONTEXT 인덱스를 생성할 때 다음과 같은 객체가 생성된다.

객체 이름	객체 타입	설명
DR\$ <i>index_name</i> \$I	테이블	컬럼 텍스트의 토큰이 저장되는 테이블이다.
DR\$ <i>index_name</i> \$X	인덱스	토큰 테이블의 토큰 컬럼과 rowid 컬럼에 걸리는 인덱스이다.
DR\$ <i>index_name</i> \$K	테이블	토큰 테이블의 document ID가 저장되는 테이블이다.

1.3.4. 인덱스 제약

CONTEXT 인덱스는 다음과 같은 경우에 제약을 가진다.

- 여러 컬럼에 대해 하나의 CONTEXT 인덱스를 걸 수 없다.
- CONTEXT 인덱스는 글로벌 파티셔닝 될 수 없다. 단, 글로벌 논파티션드 인덱스는 생성할 수 있다.
- CONTEXT 인덱스는 BASIC_LEXER, CHINESE_LEXER의 경우 VARCHAR2 타입과 CLOB 타입인 컬럼에 대해서만 생성할 수 있다.
- CONTEXT 인덱스는 JAPANESE_LEXER의 경우 VARCHAR2 타입인 컬럼에 대해서만 생성할 수 있다.

제2장 TEXT 질의

본 장에서는 Tiberio TEXT 기능을 이용한 질의에 대해 설명한다.

2.1. CATSEARCH

CATSEARCH는 CTXCAT 인덱스를 이용하기 위한 함수로 SELECT 문장의 WHERE 절에 사용된다.

- 사용법

```
CATSEARCH( column, text_query, reserved argument )
```

항목	설명
column	CTXCAT 인덱스가 걸려 있는 컬럼이다.
text_query	컬럼에서 찾을 Text를 정의하는 CATSEARCH 쿼리 연산이다.
reserved argument	추후 추가 기능 구현을 위해 예비된 argument이다.

- CATSEARCH 쿼리 연산

CATSEARCH 쿼리는 다음의 6가지 연산을 지원한다.

연산	문법	설명
Logical AND	a b c	a, b, c 모두를 포함한 row를 찾는다.
Logical OR	a b c	a, b, c 중 하나 이상을 포함한 row를 찾는다.
Logical NOT	a - b	a를 포함하고 b를 포함하지 않는 row를 찾는다. 단독으로 사용될 수 없다.
" "	"a b c"	"a b c"로 구성된 구(phrase)를 포함한 row를 찾는다.
()	(a b) c	연산의 우선순위를 정한다. a b를 먼저 계산하고 c를 후에 계산한다.
와일드카드	ab*, a*b, *ab	*는 0개 이상의 임의의 문자를 나타낸다.

- 반환값

NUMBER 타입으로 의미를 가지지 않는다.

- 예제

- 다음은 테이블 및 CTXCAT 인덱스를 생성하는 예제이다.

```
SQL> create table book(id number, name varchar(4000));
SQL> create index book_ctxcatindex on book(name) ctxcatindex;
```

- 다음은 테이블에 데이터를 입력하는 예제이다.

```
SQL> insert into book values(1,'The little boat. ');
SQL> insert into book values(2,'The little yellow digger. ');
SQL> insert into book values(3,'The magic pasta pot : an old tale. ');
SQL> insert into book values(4,'The mousehole cat. ');
SQL> insert into book values(5,'The pear in the pear tree. ');
SQL> insert into book values(6,'The rainbow fish. ');
SQL> insert into book values(7,'The story about Ping. ');
```

- 다음은 데이터를 검색하는 예제이다.

```
SQL> select * from book where catsearch(name, 'story ping', null) = 0;
      ID NAME
-----
       7 The story about Ping.

1 row selected.

SQL> select * from book where catsearch(name, 'li*', null) = 0;
      ID NAME
-----
       1 The little boat.
       2 The little yellow digger.

2 rows selected.

SQL> select * from book where catsearch(name, '"little boat"', null) = 0;
      ID NAME
-----
       1 The little boat.

1 row selected.
```

2.2. CONTAINS

CONTAINS는 CONTEXT 인덱스를 이용하기 위한 함수로 SELECT 문장의 WHERE 절에 사용된다.

- 사용법

```
CONTAINS( column, text_query, reserved argument)
```

항목	설명
column	CONTEXT 인덱스가 걸려 있는 컬럼이다.
text_query	column에서 찾을 Text를 정의하는 CONTAINS 쿼리 연산이다.
reserved argument	추후 추가 기능 구현을 위해 예비된 argument이다.

- CONTAINS 쿼리 연산

CONTAINS 쿼리는 다음의 6가지 연산을 지원한다.

연산	문법	설명
Logical AND	a AND b	a, b 모두를 포함한 row를 찾는다.
Logical OR	a OR b, a b	a, b 중 하나 이상을 포함한 row를 찾는다.
Logical NOT	a ~ b	a를 포함하고 b를 포함하지 않는 row를 찾는다.
SEQUENCE	a b	a b로 구성된 구(phrase)를 포함한 row를 찾는다.
와일드카드	ab%, %ab	%는 0개 이상의 임의의 문자를 나타낸다.
WITHIN	a WITHIN SENTENCE, a WITHIN PARAGRAPH	a를 포함하고 있는 SENTENCE 또는 PARAGRAPH가 있는 row를 찾는다.

- 반환값

NUMBER 타입으로 의미를 가지지 않는다.

- 예제

- 다음은 테이블 및 CONTEXT 인덱스 생성 및 입력하는 예제이다.

```
SQL> create table t(c1 clob);
SQL> insert into t values('Paris Rome Warsav');
SQL> insert into t values('Seoul Incheon');
SQL> insert into t values('Paris Rome Seoul');
SQL> commit;
SQL> create index t_idx on t(c1) indextype is ctxsys.context;
```

- 다음은 데이터를 검색하는 예제이다.

```
SQL> select * from t where contains(c1, 'Paris', NULL) > 0;
C1
-----
Paris Rome Warsav
Paris Rome Seoul
```

```
2 rows selected.
```

```
SQL> select * from t where contains(c1, 'Paris and Warsav', NULL) > 0;  
C1
```

```
-----  
Paris Rome Warsav
```

```
1 row selected.
```

```
SQL> select * from t where contains(C1, 'Warsav | Seoul', null) = 0;  
C1
```

```
-----  
Paris Rome Warsav
```

```
Seoul Incheon
```

```
Paris Rome Seoul
```

```
3 rows selected.
```

```
SQL> select * from t where contains(c1, 'Paris - Seoul', NULL) > 0;  
C1
```

```
-----  
Paris Rome Warsav
```

```
Paris Rome Seoul
```

```
2 rows selected.
```

- 다음은 WITHIN 연산자를 사용해서 데이터를 검색하는 예제이다. LEXER 속성에 WHITESPACE 설정이 필요하며 해당 내용은 "[제3장 TEXT Indexing Elements](#)"를 참고한다.

```
SQL> drop table t;
```

```

Table 'T' dropped.

SQL> create table t (c1 clob);

Table 'T' created.

SQL> insert into t values ('Paris.-Rome');

1 row inserted.

SQL> commit;

Commit completed.

SQL> select * from t;

C1
-----
Paris.-Rome

1 row selected.

SQL> begin
  2 text_ddl.create_preference('mylex', 'BASIC_LEXER');
  3 text_ddl.set_attribute('mylex', 'WHITESPACE', '-');
  4 end;
  5 /

PSM completed.

SQL> begin
  2 text_ddl.create_section_group('nullgroup', 'NULL_SECTION_GROUP');
  3 text_ddl.add_special_section('nullgroup', 'SENTENCE');
  4 end;
  5 /

PSM completed.

SQL> create index t_idx on t (a) indextype is ctxsys.context
  2 parameters ('LEXER mylex sync (on commit) section group nullgroup');

Index 'T_IDX' created.

SQL> select * from t where contains(a, 'Paris Rome within sentence') > 0;

0 row selected.

```

```
SQL> select * from t where contains(a, 'Paris within sentence') > 0;
```

```
C1
```

```
-----  
Paris.-Rome
```

```
1 row selected.
```


제3장 TEXT Indexing Elements

본 장에서는 Tibero TEXT 인덱스와 관련된 Elements에 대해 설명한다.

3.1. 개요

Tibero TEXT 인덱스를 생성하는 저장공간, STOPLIST, LEXER 등 각종 설정에 대한 파라미터를 설정할 수 있다.

PREFERENCE 클래스	설명
STORAGE	TEXT 인덱스를 생성하는 경우 생성되는 인덱스와 테이블의 저장 공간을 지정한다.
LEXER	TEXT 인덱스를 생성하는 경우 사용되는 LEXER의 설정을 한다.
WORDLIST	TEXT 인덱스를 생성하는 경우 생성되는 토큰에 관련된 설정을 한다.
STOPLIST	TEXT 인덱스에서 제외한 단어를 지정한다.
DASTORE	TEXT 인덱스를 생성하는 경우 문서 저장 방법을 지정한다.
FILTER	문서를 색인하기 위해 필터링되는 방법을 지정한다.
SECTION GROUP	문서를 여러 섹션으로 나누는 기준을 정의한다.

PREFERENCE를 생성하기 위해서 TEXT_DDL.CREATE_PREFERENCE를 사용한다. Tibero TEXT에서는 앞에서 설명한 7가지 PREFERENCE 클래스에 대하여 각각 이미 정의된 PREFERENCE를 제공한다. 사용자는 정의된 PREFERENCE를 사용하여 새로운 PREFERENCE를 생성하여 사용할 수 있다. PREFERENCE 생성 후 TEXT_DDL.SET_ATTRIBUTE를 사용하여 속성값을 변경할 수 있다.

3.2. STORAGE

TEXT 인덱스를 생성하는 경우 생성되는 인덱스와 테이블의 저장 공간을 지정한다.

STORAGE 클래스에는 BASIC_STORAGE가 정의되어 있으며 속성은 다음과 같다.

속성	설명
I_TABLE_CLAUSE	TEXT 인덱스 토큰이 저장되는 테이블의 테이블 스페이스이다.
I_ROWID_INDEX_CLAUSE	토큰 테이블의 ROWID 컬럼에 대한 인덱스의 테이블 스페이스이다.
I_INDEX_CLAUSE	토큰 테이블의 토큰 컬럼에 대한 인덱스의 테이블 스페이스이다.
K_TABLE_CLAUSE	추후 추가될 기능을 위해 예약된 속성이다.
P_TABLE_CLAUSE	추후 추가될 기능을 위해 예약된 속성이다.

속성	설명
N_TABLE_CLAUSE	추후 추가될 기능을 위해 예약된 속성이다.
R_TABLE_CLAUSE	매핑 테이블의 ROWID 컬럼에 대한 인덱스의 테이블 스페이스이다.
S_TABLE_CLAUSE	추후 추가될 기능을 위해 예약된 속성이다.

3.3. LEXER

TEXT 인덱스를 생성하는 경우 사용할 LEXER에 대한 설정을 하기 위한 클래스이다.

3.3.1. BASIC_LEXER

공백/특수문자로 구분하는 LEXER이다.

다음은 BASIC_LEXER에 적용할 수 있는 속성에 대한 예제이다.

속성	설명
MIXED_CASE	대소문자 구별하여 인덱싱할지 여부를 결정한다. TRUE인 경우 구별하여 인덱스한다. (속성값: TRUE/FALSE, 기본값: FALSE)
BASE_LETTER	Umlaut, acute accent와 같은 base letter들을 구분하여 인덱싱할지 여부를 검사한다. TRUE인 경우 구분하지 않고 인덱스한다. (속성값: TRUE/FALSE, 기본값: FALSE)
WHITESPACE	whitespace로 지정한 문자들은 punctuation 및 newline 문자와 결합해 sentence 및 paragraph의 구분자 역할을 하여 sentence 및 paragraph 검색에 사용된다. 기본값은 공백 또는 탭 문자이고 사용자가 whitespace를 지정했을 경우 기본값 외에도 사용자가 지정한 값으로 whitespace가 인식된다. (속성값: 문자열, 기본값: 공백 또는 탭 문자)

3.3.2. JAPANESE_LEXER

일본어 텍스트에서 토큰을 만들기 위한 LEXER이다. JAPANESE_LEXER는 일본어 형태학(morphological) 분석기이다.

다음은 JAPANESE_LEXER에 적용할 수 있는 속성에 대한 예제이다.

속성	설명
ASCII_MIXED_CASE	ASCII 영역의 문자들에 대하여 대소문자 구별하여 인덱싱할지 여부를 결정한다. TRUE인 경우 구별하여 인덱스한다.

속성	설명
	(속성값: TRUE/FALSE, 기본값: FALSE)
DELIMITER	특수문자(/, -, .)와 같은 일본어의 <code>delimiter</code> 를 구분자로 사용할 것인지 여부를 결정한다. TRUE인 경우 구분하지 않고 인덱스한다. (속성값: TRUE/FALSE, 기본값: FALSE)

3.3.3. CHINESE_LEXER

중국어 텍스트에서 토큰을 만들기 위한 LEXER이다.

CHINESE_LEXER는 다음과 같은 제약을 가진다.

- UTF8, GBK character set에서만 CHINESE_LEXER를 지원한다.
- CTXCAT 인덱스는 사용할 수 없고, CONTEXT 인덱스만 사용 가능하다.

3.3.4. CHINESE_VGRAM_LEXER

중국어 텍스트에서 토큰을 만들기 위한 LEXER이다. CHINESE_LEXER와는 다르게 의미 기반이 아닌 글자 수 기반으로 토큰을 생성한다.

CHINESE_VGRAM_LEXER는 다음과 같은 제약을 가진다.

- UTF8 character set에서만 CHINESE_VGRAM_LEXER를 지원한다.

3.3.5. WORLD_LEXER

여러 언어가 포함된 문서에서 토큰을 만들기 위한 LEXER이다. 언어를 자동으로 감지하여 토큰을 만든다. 띄어쓰기가 있는 언어는 띄어쓰기 기준으로 토큰을 생성하고, 띄어쓰기가 없는 일본어나 중국어의 경우에는 2-gram으로 토큰을 만든다.

WORLD_LEXER는 다음과 같은 제약을 가진다.

- UTF8 character set에서만 WORLD_LEXER를 지원한다.
- 2-gram으로 토큰을 생성하는 일본어나 중국어는 와일드카드를 지원하지 않는다.
- CTXCAT 인덱스는 사용할 수 없고, CONTEXT 인덱스만 사용 가능하다.

3.4. WORDLIST

TEXT 인덱스를 생성하는 경우 생성되는 토큰의 설정과 관련된 클래스이다. 현재 BASIC_WORDLIST가 정의되어 있으며 속성은 다음과 같다.

속성	설명
PREFIX_INDEX	YES로 설정하는 경우 단어의 접두어를 별도로 인덱싱하여 접두어 검색 성능을 향상시킨다. (속성값: YES/NO, 기본값: NO)
PREFIX_MIN_LENGTH	PREFIX_INDEX이 YES일 경우 인덱싱하는 접두어의 최소 길이를 지정한다. (범위: 1 ~ 64, 기본값: 1)
PREFIX_MAX_LENGTH	PREFIX_INDEX이 YES일 경우 인덱싱하는 접두어의 최대 길이를 지정한다. (범위: 1 ~ 64, 기본값: 64)

3.5. STOPLIST

STOPLIST는 인덱스에서 제외될 단어를 지정한다. DEFAULT_STOPLIST, EMPTY_STOPLIST가 정의되어 있다. DEFAULT_STOPLIST에는 기본적인 STOPWORD가 입력되어 있다. 자세한 내용은 “[Appendix A. STOPWORDS](#)”를 참고한다. EMPTY_STOPLIST는 STOPWORD가 없는 STOPLIST를 의미한다.

TEXT_DDL.CREATE_STOPLIST를 사용하여 새로운 STOPLIST를 생성하고, TEXT_DDL.ADD_STOPWORD를 사용하여 STOPWORD를 추가할 수 있다.

3.6. DATASTORE

DATASTORE는 문서 저장 방법을 지정 할 수 있다. DATASTORE 클래스에는 DIRECT_DATASTORE와 MULTI_COLUMN_DATASTORE가 정의되어 있다.

3.6.1. DIRECT_DATASTORE

DIRECT_DATASTORE를 사용하면 데이터는 텍스트 컬럼에 내부적으로 저장된다. 각 row는 단일 문서로 인덱싱된다. DIRECT_DATASTORE 유형에는 속성이 없다.

3.6.2. MULTI_COLUMN_DATASTORE

MULTI_COLUMN_DATASTORE는 데이터가 여러 개의 컬럼에 저장되어 있을 때 사용한다. MULTI_COLUMN_DATASTORE로 지정된 컬럼들은 각 row별로 합쳐져서 인덱싱된다.

속성	설명
COLUMNS	인덱싱할 컬럼들을 콤마(,)로 구분하여 지정한다. VARCHAR, CLOB, NUMBER, DATE 타입의 컬럼들만 지정할 수 있으며, 컬럼 이름 외에도 컬럼에 대한 expression 및 PSM 함수를 지정할 수 있다. (속성값: STRING)
DELIMITER	각 컬럼 텍스트를 구분하는 구분자를 지정한다. <ul style="list-style-type: none"> - COLUMN_NAME_TAG : 각 컬럼의 텍스트는 XML 형식의 시작/마침 태그로 구분되며, 각 태그는 컬럼 이름을 가진다. (기본값) - NEWLINE : 각 컬럼의 텍스트는 newline으로 구분된다.

3.7. FILTER

문서를 색인하기 위해 필터링되는 방법을 결정한다. FILTER 클래스에는 NULL_FILTER만 정의되어 있다.

3.7.1. NULL_FILTER

NULL_FILTER는 필터링을 하지 않는다. 일반 문서, HTML, XML을 그대로 인덱싱한다. NULL_FILTER 유형에는 속성이 없다.

3.8. SECTION GROUP

문서구조를 여러 섹션으로 나누는 기능이다. 섹션을 정의하지 않는 NULL_SECTION_GROUP만 정의되어 있다. TEXT_DDL.CREATE_SECTION_GROUP을 사용하여 새로운 SECTION_GROUP을 생성할 수 있다.

3.9. TEXT_DDL 패키지

다음은 PREFERENCE를 생성하고 관리하기 위해 TEXT_DDL 패키지에서 제공하는 프러시저이다.

이름	설명
ADD_SPECIAL_SECTION	SECTION GROUP에 special section인 SENTENCE 또는 PARAGRAPH를 추가하기 위한 프러시저이다.
ADD_STOPWORD	STOPLIST에 STOPWORD를 추가하기 위한 프러시저이다.
CREATE_PREFERENCE	새로운 PREFERENCE를 정의하는 프러시저이다.
CREATE_SECTION_GROUP	새로운 SECTION_GROUP을 정의하는 프러시저이다.
CREATE_STOPLIST	새로운 STOPLIST를 만들기 위한 프러시저이다.
DROP_PREFERENCE	PREFERENCE를 삭제하기 위한 프러시저이다.

이름	설명
DROP_SECTION_GROUP	SECTION_GROUP을 삭제하기 위한 프러시저이다.
DROP_STOPLIST	STOPLIST를 삭제하기 위한 프러시저이다.
REMOVE_SECTION	SECTION GROUP에서 SECTION을 삭제하기 위한 프로시저이다.
REMOVE_STOPWORD	STOPLIST에서 STOPWORD를 삭제하기 위한 프러시저이다.
SET_ATTRIBUTE	PREFERENCE의 속성값을 설정하기 위한 프러시저이다.

3.9.1. ADD_SPECIAL_SECTION

SECTION GROUP에 Special Section인 SENTENCE 또는 PARAGRAPH을 추가하기 위한 프로시저이다. Special Section을 추가함으로써 WITHIN 연산자를 사용한 SENTENCE 및 PARAGRAPH 검색이 가능해진다.

Special Section을 가진 SECTION GROUP을 지정하여 인덱스를 생성할 때 정해진 구분자를 기준으로 Special Section들을 토큰화한다.

SENTENCE의 구분 기준은 **[단어][punctuation][whitespace]** 또는 **[단어][punctuation][newline]**이고, PARAGRAPH의 구분 기준은 **[단어][punctuation][newline][whitespace]** 또는 **[단어][punctuation][new line][newline]**이다. 이때 punctuation 기본값은 [. / ! ?], newline 기본값은 [\n / \r]이며 사용자가 따로 punctuation 및 newline 값을 변경할 수 없다.

- 프로토타입

```
ADD_SPECIAL_SECTION (
    group_name      IN VARCHAR2,
    section_name    IN VARCHAR2)
```

- 파라미터

파라미터	설명
group_name	추가할 대상의 SECTION GROUP 이름이다.
section_name	추가할 section 이름으로 'SENTENCE' 또는 'PARAGRAPH'를 지정할 수 있다.

- 예제

```
SQL>EXEC TEXT_DDL.ADD_SPECIAL_SECTION('nullgroup','SENTENCE');
```

3.9.2. ADD_STOPWORD

STOPWORD 타입에 새로운 STOPWORD를 추가하기 위한 프러시저이다.

- 프로토타입

```
ADD_STOPWORD (
    stoplist_name      IN VARCHAR2,
    stopword           IN VARCHAR2,
    lang                IN VARCHAR2 DEFAULT 'ALL' )
```

- 파라미터

파라미터	설명
stoplist_name	추가할 대상의 STOPLIST 이름이다.
stopword	추가할 STOPWORD이다.
lang	STOPWORD가 적용될 언어이다. (미지원)

- 예제

```
SQL>EXEC TEXT_DDL.ADD_STOPWORD('DEFAULT_STOPLIST','test_word');
```

3.9.3. CREATE_PREFERENCE

새로운 PREFERENCE를 만들기 위한 프러시저이다.

- 프로토타입

```
CREATE_PREFERENCE (
    udef_pref_name     IN VARCHAR2,
    pdef_pref_name     IN VARCHAR2)
```

- 파라미터

파라미터	설명
udef_pref_name	사용자가 정의할 PREFERENCE 이름이다.
pdef_pref_name	Tibero TEXT에서 정의된 PREFERENCE 이름이다.

- 예제

```
SQL>EXEC TEXT_DDL.CREATE_PREFERENCE('TEST1','BASIC_WORDLIST');
```

3.9.4. CREATE_SECTION_GROUP

새로운 SECTION_GROUP을 만들기 위한 프러시저이다.

- 프로토타입

```
CREATE_SECTION_GROUP (
    udef_pref_name      IN  VARCHAR2,
    pdef_pref_name      IN  VARCHAR2)
```

- 파라미터

파라미터	설명
udef_pref_name	사용자가 정의할 PREFERENCE 이름이다.
pdef_pref_name	Tibero TEXT에서 정의된 PREFERENCE 이름이다.

- 예제

```
SQL>EXEC TEXT_DDL.CREATE_SECTION_GROUP('my_sg', 'NULL_SECTION_GROUP');
```

3.9.5. CREATE_STOPLIST

새로운 STOPLIST을 만들기 위한 프러시저이다.

- 프로토타입

```
CREATE_STOPLIST (
    stoplist_name      IN  VARCHAR2,
    stoplist_type      IN  VARCHAR2 DEFAULT 'DEFAULT_STOPLIST')
```

- 파라미터

파라미터	설명
stoplist_name	생성할 STOPLIST 이름이다.
stoplist_type	생성할 STOPLIST의 타입이다. 현재 DEFAULT_STOPLIST와 EMPTY_STOPLIST가 있다.

- 예제

```
SQL>EXEC TEXT_DDL.CREATE_STOPLIST('new_stoplist', 'DEFAULT_STOPLIST');
```

3.9.6. DROP_PREFERENCE

생성한 PREFERENCE를 삭제하기 위한 프러시저이다.

- 프로토타입

```
DROP_PREFERENCE (udef_pref_name IN VARCHAR2);
```


- 파라미터

파라미터	설명
undef_pref_name	삭제할 PREFERENCE 이름이다.

- 예제

```
SQL>EXEC TEXT_DDL.DROP_PREFERENCE('TEST1');
```

3.9.7. DROP_SECTION_GROUP

생성한 SECTION_GROUP을 삭제하기 위한 프러시저이다.

- 프로토타입

```
DROP_SECTION_GROUP (undef_pref_name IN VARCHAR2);
```

- 파라미터

파라미터	설명
undef_pref_name	삭제할 SECTION_GROUP 이름이다.

- 예제

```
SQL>EXEC TEXT_DDL.DROP_PREFERENCE('my_sg');
```

3.9.8. DROP_STOPLIST

STOPLIST를 삭제하기 위한 프러시저이다.

- 프로토타입

```
DROP_STOPLIST (stoplist_name IN VARCHAR2);
```

- 파라미터

파라미터	설명
stoplist_name	삭제할 STOPLIST 이름이다.

- 예제

```
SQL>EXEC TEXT_DDL.DROP_STOPLIST('new_stoplist');
```

3.9.9. REMOVE_SECTION

SECTION GROUP에서 SECTION을 삭제하기 위한 프로시저이다.

- 프로토타입

```
REMOVE_SECTION (  
    group_name      IN VARCHAR2,  
    section_name    IN VARCHAR2)
```

- 파라미터

파라미터	설명
group_name	삭제할 대상의 SECTION GROUP 이름이다.
section_name	삭제할 SECTION 이름이다.

- 예제

```
SQL>EXEC TEXT_DDL.REMOVE_SECTION('nullgroup','test_section');  
SQL>EXEC TEXT_DDL.REMOVE_SECTION('nullgroup','SENTENCE');
```

3.9.10. REMOVE_STOPWORD

STOPLIST에서 STOPWORD를 삭제하기 위한 프로시저이다.

- 프로토타입

```
REMOVE_STOPWORD (  
    stoplist_name    IN VARCHAR2,  
    stopword         IN VARCHAR2,  
    lang             IN VARCHAR2 DEFAULT 'ALL' )
```

- 파라미터

파라미터	설명
stoplist_name	삭제할 대상의 STOPLIST 이름이다.
stopword	삭제할 STOPWORD이다.
lang	STOPWORD가 적용되는 언어이다. (미지원)

- 예제

```
SQL>EXEC TEXT_DDL.REMOVE_STOPWORD('DEFAULT_STOPLIST','test_word');
```

3.9.11. SET_ATTRIBUTE

생성된 PREFERENCE의 속성값을 설정하기 위한 프러시저이다.

- 프로토타입

```
SET_ATTRIBUTE (  
    udef_pref_name      IN  VARCHAR2,  
    attribute_name      IN  VARCHAR2,  
    value               IN  VARCHAR2)
```

- 파라미터

파라미터	설명
udef_pref_name	속성값을 변경할 사용자가 정의한 PREFERENCE이다.
attribute_name	변경할 대상 속성이다.
value	변경할 값이다.

- 예제

```
SQL>EXEC TEXT_DDL.SET_ATTRIBUTE('TEST1','PREFIX_INDEX','TRUE');
```

3.10. CTX_DDL 패키지

CONTEXT 인덱스를 관리하기 위한 CTX_DDL 패키지에서 제공하는 프러시저들을 알아본다.

3.10.1. OPTIMIZE_INDEX

인덱스를 동기화 한 후 인덱스를 최적화하는 방법이다. 인덱스를 최적화하면 이전 데이터가 제거되고 최소화되므로 쿼리 응답시간이 향상 될 수 있다.

- 프로토타입

```
CTX_DDL.OPTIMIZE_INDEX (  
    idx_name            IN  VARCHAR2,  
    optlevel            IN  VARCHAR2,  
    maxtime             IN  NUMBER DEFAULT NULL,  
    token               IN  VARCHAR2 DEFAULT NULL,  
    part_name           IN  VARCHAR2 DEFAULT NULL,  
    token_type          IN  NUMBER DEFAULT NULL,  
  
    parallel_degree     IN  NUMBER DEFAULT 1)
```

- 파라미터

파라미터	설명
idx_name	최적화할 인덱스 이름이다.
optlevel	최적화 방법을 설정한다. 다음과 같이 FAST, FULL 2가지 방법이 있다. - FAST : 중복된 토큰을 합쳐주고, 이전 데이터(삭제된 행)를 제거하지 않는다. - FULL : 중복된 토큰을 합쳐주고, 이전 데이터(삭제된 행)를 제거한다.
maxtime	최대 최적화 시간을 설정한다. (미지원)
token	최적화할 토큰을 설정한다. (미지원)
part_name	로컬 인덱스의 경우 파티션 이름을 설정한다.
token_type	최적화할 토큰 타입을 설정한다. (미지원)
parallel_degree	최적화할 parallel_degree를 설정한다. (미지원)

- 예제

```
begin
  ctx_ddl.optimize_index('myindex', 'FAST');
end
```

3.10.2. SYNC_INDEX

기본 테이블에 대한 삽입, 업데이트 및 삭제가 있을 때 CONTEXT 인덱스의 자동 동기화를 위해 사용된다.

- 프로토타입

```
CTX_DDL.SYNC_INDEX (
  idx_name          IN  VARCHAR2 DEFAULT NULL,
  memory            IN  VARCHAR2 DEFAULT NULL,
  part_name         IN  VARCHAR2 DEFAULT NULL,
  parallel_degree   IN  NUMBER  DEFAULT 1,
  maxtime           IN  NUMBER  DEFAULT NULL,
  locking           IN  NUMBER  DEFAULT LOCK_WAIT)
```

- 파라미터

파라미터	설명
idx_name	최적화할 인덱스 이름이다.
memory	동기화에 사용할 런타임 메모리를 설정한다. 1K에서 256G까지의 값을 지정할 수 있다.
part_name	로컬 인덱스의 경우 파티션 이름을 설정한다.
parallel_degree	최적화할 parallel_degree를 설정한다. (미지원)
maxtime	최대 최적화 시간을 설정한다. (미지원)
locking	SYNC_INDEX가 동일한 인덱스 또는 파티션에서 이미 다른 동기화가 실행되고 있는 상황을 처리하는 방법을 구성한다. (미지원)

- 예제

```
begin
  ctx_ddl.sync_index('myindex');
end
```

3.11. CTX_OUTPUT 패키지

CONTEXT 인덱스를 관리하기 위한 CTX_OUTPUT 패키지에서 제공하는 프리시저들을 알아본다.

3.11.1. END_LOG

인덱싱 과정의 로깅을 중지한다.

- 프로토타입

```
CTX_OUTPUT.END_LOG;
```

- 예제

```
begin
  ctx_output.end_log;
end
```

3.11.2. START_LOG

인덱싱 과정의 로깅을 시작한다.

- 프로토타입

```
CTX_OUTPUT.START_LOG (  
    file_name          IN  VARCHAR2,  
    write_option       IN  BOOLEAN DEFAULT TRUE)
```

- 파라미터

파라미터	설명
file_name	로그 파일의 이름을 지정한다.
write_option	이미 로그 파일이 존재하는 경우 파일을 덮어 쓰지 새로운 로그 파일을 추가할지 지정한다. 기본값은 파일을 덮어 쓰는 것이다.

- 예제

```
begin  
    ctx_output.start_log('myindex', true);  
end
```

Appendix A. STOPWORDS

본 장에서는 Tiberio TEXT에서 사용하는 STOPWORD를 알파벳 순으로 기술한다.

A.1. 영어

STOPWORD	STOPWORD	STOPWORD	STOPWORD	STOPWORD	STOPWORD
a	did	in	only	then	where
all	do	into	onto	there	whether
almost	does	is	or	therefore	which
also	either	it	our	these	while
although	for	its	ours	they	who
an	from	just	s	this	whose
and	had	ll	shall	those	why
any	has	me	she	though	will
are	have	might	should	through	with
as	having	Mr	since	thus	would
at	he	Mrs	so	to	yet
be	her	Ms	some	too	you
because	here	my	still	until	your
been	hers	no	such	ve	yours
both	him	non	t	very	
but	his	nor	than	was	
by	how	not	that	we	
can	however	of	the	were	
could	i	on	their	what	
d	if	one	them	when	

색인

A

ADD_SPECIAL_SECTION, 18
ADD_STOPWORD, 18

B

BASIC_LEXER, 14
BASIC_LEXER 속성
 BASE_LETTER, 14
 MIXED_CASE, 14
 WHITESPACE, 14

C

CATSEARCH, 7
CATSEARCH 쿼리 연산, 7
 ", 7
 (), 7
 Logical AND, 7
 Logical NOT, 7
 Logical OR, 7
 와일드카드, 7
CHINESE_LEXER, 15
CHINESE_VGRAM_LEXER, 15
CONTAINS, 8
CONTAINS 쿼리 연산, 9
 Logical AND, 9
 Logical NOT, 9
 Logical OR, 9
 SEQUENCE, 9
 WITHIN, 9
 와일드카드, 9
CONTEXT 인덱스, 3
 인덱스 관련 객체, 5
 인덱스 생성, 3
 인덱스 제거, 2, 5
 인덱스 제약, 5
CONTEXT 인덱스 관련 객체, 5

DR\$index_name\$I, 5
DR\$index_name\$K, 5
DR\$index_name\$X, 5
CONTEXT 인덱스 생성, 3
CONTEXT 인덱스 제거, 2, 5
CONTEXT 인덱스 제약, 5
CREATE_PREFERENCE, 19
CREATE_SECTION_GROUP, 19
CREATE_STOPLIST, 20
CTX_DDL 패키지, 23
 OPTIMIZE_INDEX, 23
 SYNC_INDEX, 24
CTX_OUTPUT 패키지, 25
 ENT_LOG, 25
 START_LOG, 25
CTXCAT 인덱스, 1
 CTXCAT 인덱스 생성, 1
 인덱스 관련 객체, 2
 인덱스 제거, 2, 5
 인덱스 제약, 3
CTXCAT 인덱스 관련 객체, 2
 DR\$index_name_TRG_D, 3
 DR\$index_name_TRG_I, 3
 DR\$index_name_TRG_U, 3
 DR\$index_name\$I, 3
 DR\$index_name\$R, 3
 DR\$index_name\$X, 3
CTXCAT 인덱스 생성, 1
CTXCAT 인덱스 제거, 2, 5
CTXCAT 인덱스 제약, 3

D

DATASTORE, 16
 DIRECT_DATASTORE, 16
 MULTI_COLUMN_DATASTORE, 16
DIRECT_DATASTORE, 16
DR\$index_name_TRG_D, 3
DR\$index_name_TRG_I, 3
DR\$index_name_TRG_U, 3
DR\$index_name\$I, 3, 5
DR\$index_name\$K, 5
DR\$index_name\$R, 3

DR\$index_name\$X, 3, 5
DROP_PREFERENCE, 20
DROP_SECTION_GROUP, 21
DROP_STOPLIST, 21

E

END_LOG, 25

F

FILTER, 17
 NULL_FILTER, 17

J

JAPANESE_LEXER, 14
JAPANESE_LEXER 속성
 ASCII_MIXED_CASE, 14
 DELIMITER, 15

L

LEXER, 14
 BASIC_LEXER, 14
 CHINESE_LEXER, 15
 CHINESE_VGRAM_LEXER, 15
 JAPANESE_LEXER, 14
 WORLD_LEXER, 15

M

MULTI_COLUMN_DATASTORE, 16
MULTI_COLUMN_DATASTORE 속성
 COLUMNS, 17
 PREFIX_MAX_LENGTH, 17

N

NULL_FILTER, 17

O

OPTIMIZE_INDEX, 23

P

PREFERENCE 클래스
 DATASTORE, 13

FILTER, 13
LEXER, 13
SECTION_GROUP, 13
STOPLIST, 13
STORAGE, 13
WORDLIST, 13

R

REMOVE_SECTION, 22
REMOVE_STOPWORD, 22

S

SECTION_GROUP, 17
SET_ATTRIBUTE, 23
START_LOG, 25
STOPLIST, 16
STOPWORD, 27
STORAGE, 13
STORAGE 속성
 I_INDEX_CLAUSE, 13
 I_ROWID_INDEX_CLAUSE, 13
 K_TABLE_CLAUSE, 13
 N_TABLE_CLAUSE, 14
 P_TABLE_CLAUSE, 13
 R_TABLE_CLAUSE, 14
 S_TABLE_CLAUSE, 14
 STOPLIST, 13
SYNC_INDEX, 24

T

TEXT 인덱스, 1
TEXT_DDL 패키지, 17
 ADD_SPECIAL_SECTION, 18
 ADD_STOPWORD, 18
 CREATE_PREFERENCE, 19
 CREATE_SECTION_GROUP, 19
 CREATE_STOPLIST, 20
 DROP_PREFERENCE, 20
 DROP_SECTION_GROUP, 21
 DROP_STOPLIST, 21
 REMOVE_SECTION, 22
 REMOVE_STOPWORD, 22

SET_ATTRIBUTE, 23

W

WORDLIST, 16

WORDLIST 속성

 PREFIX_INDEX, 16

 PREFIX_MAX_LENGTH, 16

 PREFIX_MIN_LENGTH, 16

WORLD_LEXER, 15

