Text Datenbanken

Oracle*8i interM*edia Text

Version:
04.11.2000

Autoren:

Bent Großmann
Niels Bauer

Zusammenfassung:

Dieses Dokument befaßt sich mit Volltextdatenbanken und stellt diese anhand Oracle*8i interM*edia Text vor. Verfahren und Funktionsweisen werden kurz in Form von erläuternden Beispielen gezeigt.
Besonderes Augenmerk liegt auf dem Laden von Texten, deren Indizierung, dem stellen von Anfragen sowie die Präsentation von Ergebnisdokumenten. Ein Vergleich mit SQL/MM Full Text schließt das Dokument ab.

Inhalt:

Text Datenbanken

Oracle*8i interM*edia Text

Kapitel 1

Einführung

1. Einführung

Oracle interMedia Text ist Bestandteil von Oracle interMedia und wird mit Version 8i der Oracle Datenbank ausgeliefert. Oracle*8i inter*Media Text erlaubt es strukturierte Texte zu verwalten, indizieren und danach zu suchen.

Die Freitextsuche erlaubt es Dokumente innerhalb und außerhalb der Oracle8i Datenbank zu durchsuchen. Es ermöglicht content-based query's, d.h. das finden von Dokumenten oder Texten die ein bestimmtes Wort enthalten.

1.1 Ein kurzes Beispiel

- *anlegen einer Tabelle*

```
create table docs (id number primary key, text varchar2( 80 ) );
```

- *einfügen von Daten*

```
insert into docs values ( 1, 'erstes Dokument' ) ;
insert into docs values ( 2, 'zweites Dokument' ) ;
commit ;
```

- *bilden eines Index*

```
create index doc_index on docs ( text )
indextype is ctxsys.context;
```

- *nun eine Anfrage*

```
select id from docs
where constains ( text , ' first ' ) > 0 ;
```

Es werden alle Zeilen , in denen die Textspalte, das Wort „first" enthält gefunden.
Der Ausdruck „ > 0 „ ist notwendig um ein erlaubtes Oracle SQL Kommando zu benutzen, da boolsche Werte im Moment nicht unterstützt werden.

1.2 Wichtige Änderungen im Vergleich mit Oracle ConTex 2.x

- Sämtliche Funktionalitäten sind schon im Kernel enthalten und laufen somit nicht mehr als Dämon wie dies in älteren Versionen der Fall war. Dies erhöht die Performance und erleichtert die Entwicklung deutlich.
- Viele ConText Kommandos änderten sich von PL/SQL zu Standard SQL .
- Das Query – Model wurde vereinfacht und besteht normalerweise nur noch aus single-step-query's
- Two-step-query's werden in Oracle 8i nicht mehr unterstützt und In-memory-query's werden durch einfache SQL-Cursors ersetzt.
- Index Statements wechselten von PL/SQL zu normalen SQL

1.3 System definierte Rollen

- *CTXSYS Rolle*

Mit der CTXSYS Rolle ermöglicht dem Benutzer das starten des CTXSRV Servers.
Weiterhin kann er alle ausführen aller Tätigkeiten eines CTXAPP Benutzers ausführen.

- *CTXAPP Rolle*

Mit der CTXAPP Rolle kann der Benutzer Indizes erstellen, das Textdaten Wörterbuch verwalten sowie Präferenzen können eingebunden, angelegt und gelöscht werden.
Das benutzen von Anfragen (Text Anfragen) ist genauso möglich wie auch die Benutzung der *inter*Media Text PL/SQL Funktionalitäten.

1.4 Laden von Dokumenten / gültige Datenformate

Die Default - Einstellung erwartet, das die Dokumente in eine Textfeld geladen werden.
Das Einbinden des Dateisystems bzw. das Einbinden von URL 's ist möglich.

Oracle unterstützt verschiedene Varianten zum laden von Daten

- *SQL INSERT Anweisung*
- *ctxload executable*
- *SQL*Loader*
- DBMS_LOB.LOADFROMFILE() PL/SQL Prozedur zum laden von LOBs aus BFILE's

Textfelder können verschiedene Datentypen wie VARCHAR2, CLOB, BLOB, CHAR oder BFILE haben.

Als Dokumentenformate sind HTML, PDF, Microsoft Word sowie Plain Text möglich.
(Diese Formate können in ein Textfeld geladen werden)

1.5 Indexe erstellen

Wenn der Text in eine Textspalte geladen wurde kann damit begonnen werden eine Index zu erstellen. Hier ein kurzes Beispiel:

create index myindex on docs(text) indextype is ctxsys.context;

Beim erstellen von Indexen gilt es die Besonderheiten der verschiedenen Sprachen zu beachtet.

- *Generelle Einstellungen für alle Sprachen*

  - die Datentypen NCLOB, DATE und NUMBER können nicht indiziert werden
  - Zum indizieren von verschiedenen Dokumenten ist es nötig das der im System vorhandene Inso - Filter richtig eingerichtet wurde.
  - Es wird die im System angegebene Sprache zum indizieren verwendet
  - Die Default Stop - List für die im System angegebene Sprache wird verwendet
  - fuzzy und stemming anfragen werden unterstützt wenn diese für ihre Sprache vorhanden sind.

  Diese Standardeinstellungen können geändert werden.

- *Sprachen spezifische Einstellungen*

Englisch	Themen Informationen sind möglich und damit auch ABOUT Anfragen Diese werden dadurch präziser.
Deutsch	CASE - sensitives Indizieren Composite Indexing Alternate spelling
Holländisch	Composite Indexing Alternate spelling
Schwedisch	Alternate spelling

Eine Indexwartung ist nach dem Einfügen, Update oder dem löschen von Sätzen notwendig ,dazu wird INDEX ALTER verwendet.
Der im Hintergrund laufende ctxsrv - Server synchronisiert den Index in festen Intervallen automatisch.

1.6 Anfragen

Es werden grundsätzlich zwei Arten von Anfragen unterschieden,

word query und about-query.

- *Wort - Anfragen*

  Eine word - query ist eine von einfachen Hochkommas eingeschlossenen Anfrage, nach einem bestimmten Wort oder einer Phrase

  Ein Beispiel (word-querys):

  ```
  SELECT SCORE(1) title  from news
        WHERE CONTAINS ( text , ' oracle ' , 1) > 1
  ```

  Es können AND , OR Operatoren sowie strukturierte Prädikate benutzt werden

- *About - Anfragen*

  In allen Sprachen zählen ABOUT Anfragen die Anzahl von relevanten Dokumenten welche die Kriterien einer gestellten Anfrage erfüllen.
  In englischer Sprache ist es möglich die Themen -Komponenten des Index zu benutzen.
  Diese wird automatisch erstellt und erlaubt es Dokumente zu finden die konzeptionell mit der gestellten Anfrage beschäftigen.

  Ein Beispiel (about query):

  ```
  SELECT SCORE(1) title from news
        WHERE CONTAINS(text, ' about( politics ) ' ,1) > 0;
  ```

  Diese Anfrage findet alle Dokumente die von ‚politics' handeln, also nicht nur das Wort enthalten.

1.7 Dokumentenpräsentation

Normalerweise werden die Resultate einer Anfrage dem Benutzer angezeigt. Der Benutzer wählt einen Eintrag aus und die Anwendungsumgebung wird das Dokument präsentieren.
Mit interMediaText bestehen zwei unterschiedliche Wege
Es besteht die Möglichkeit ein Dokument mit hervorgehobenen Begriffen anzuzeigen.
Dabei sind die Terme der gestellten Anfrage hervorgehoben.
Solche Terme können die Wörter der Anfrage oder die Terme einer ABOUT- Anfrage sein.
Dabei werden für verschiedene Dokumentenformate unterschiedliche Prozeduren verwendet.

Textausgabetyp	**Prozedur**
Hervorgehoben Plain Text	CTX_DOC.MARKUP
Hervorgehoben HTML_Version	CTX_DOC.MARKUP
Hervorgehoben offset Information Plain Text	CTX_DOC.HIGHLIGHT
Hervorgehoben offset Information HTML	CTX_DOC.HIGHLIGHT
Plain text	CTX_DOC.FILTER
HTML	CTX_DOC.FILTER

1.8 SQL-Kommandos

- *ALTER INDEX*

  ALTER INDEX wird verwendet um folgende Aufgaben zu realisieren

  Umbenennen eines Index
  Neuaufbau des Index mit anderen Einstellungen
  Weiterführen eine fehlgeschlagenen Indexoperation
  Durchführen von Indexwartung (DML)
  Indexoptimierung
  Hinzufügen von Stopwörtern zum Index

  Rename syntax: ALTER INDEX [schema.]index_name RENAME to new_index_name;

  schema.index_name Name des umzubenennenden Index
  new_index_name Neuer Name für den Index (max. 25 Zeichen)

  Rebuild syntax: ALTER INDEX [schema.]index REBUILD [online] [parameters (paramstring)];

  [online] Ermöglicht nicht blockierende Operation
  Ermöglicht Anfragen während der Wartung des Index
  PARAMETERS (paramstring) Angeben von neuen Parametern zur Indexwartung

  Beispiele:

  Beenden eines fehlgeschlagenen Index
  ```
  ALTER INDEX newsindex rebuild parameters('resume memory 2M');
  ```

  Neuaufbau eines Index
  ```
  ALTER INDEX newsindex rebuild parameters('replace stoplist new_stop');
  ```

  Index Optimierung im Fast Mode
  ```
  ALTER INDEX newsindex rebuild parameters('optimize fast');
  ```

- *DROP INDEX*

  DROP INDEX wird benutzt um einen Index zu verwerfen

  Syntax : drop index [schema.] index [force];

  [force] benutzen falls eine Indexoperation hängen geblieben ist

  Beispiel `drop index doc_index;`

- *CONTAINS*

  Stellt den enthält Operator dar.

  Syntax: CONTAINS([schema.] column, text_query VARCHAR2,[label NUMBER]) RETURN NUMBER;

  [schema.]column
  Gibt die Textspalte an in der gesucht wird. Diese Spalte muß einen Index haben

  text_query
  Der zu suchende Text

  Beispiel:

  SELECT SCORE(1) title from newsindex
  WHERE CONTAINS(text, 'oracle', 1) > 0;

- *CREATE INDEX*

  Dient zum anlegen eines interMediaText index, dies ist ein oracle domain index vom typ context welcher das extensible indexing framework benutzt.

  Syntax: CREATE INDEX [schema.] index on [schema.] table(column) INDEXTYPE IS ctxsys.context [PARAMETERS(paramstring)];

  [schema.]index
  Gibt den Namen des Textes an der zu Indizieren ist

  [schema.]table(column)
  Name der Tabelle und er Spalte die zu indizieren sind. Die Tabelle muß eine Primärschlüssel haben. Die Tabellenspalte muß von einem der folgenden Datentypen sein: CHAR, VARCHAR, VARCHAR2, BLOB, CLOB, or BFILE.

  Beispiele:

  Anlegen eines Index mit Default Einstellungen
  create index newsindex on mytable(news) indextype is ctxsys.context;

  Anlegen eines Index mit Benutzereinstellungen
  create index newsindex on mytable(news) indextype is ctxsys.context
  parameters('lexer MY_LEXER stoplist MY_STOP');

- *SCORE*

  Der score Operator in einem select Klausel gibt die Werte zurück die von einem CONTAINS Operator erzeugt werden

  Syntax: `SCORE(label NUMBER)`

  (Label NUMBER) Spezifiziert eine Zahl um die Score- Prozedur in einer Anfrage zu identifizieren.

  Beispiel:

  SELECT employee_name, SCORE(10), SCORE(20)
  FROM employee_database
  WHERE CONTAINS (emp.resume, 'software developer', 10) > 0 OR
  CONTAINS (emp.resume, 'java', 20) > 0
  ORDER BY NVL(SCORE(10),0), NVL(SCORE(20),0);

Text Datenbanken

Oracle*8i inter*Media Text

Kapitel 2

Laden von Dokumenten

2. Laden von Dokumenten

Die Default - Einstellung erwartet das Dokumente in eine Textfeld geladen werden, aber das einbinden des Dateisystems bzw. das Einbinden von URL's ist möglich.
Dies geschieht durch Verwendung von Datastore Objekts.

2.1 Datastored Objekts

Datastore Objekte legen fest, wie Text gespeichert wird

Objekte	Beschreibung	Attribute
DIRECT_DATASTORE	Die Daten werden intern in einer Text - Spalte gespeichert. Jede Zeile wird als einzelnes Dokument indiziert.	hat keine Attribute
DETAIL_DATASTORE	Die Daten werden intern in einer Text - Spalte gespeichert. Das Dokument besteht aus einer oder mehreren Zeilen in einer Dateiltabelle. "Header Information" in Haupttabelle	*binary* wenn TRUE, fügt ORACLE kein Zeilenende nach dem Ende jeder Detailzeile ein *detail_table* Name der Detailtabelle kann angegeben werden *detail_key* Name des Fremdschlüssels der Detailtabelle *detail_lineno* Name der Sequenzspalte der Detailtabelle *detail_text* Name der Textspalte der Detailtabelle
FILE_DATASTORE	Die Daten werden extern in Daten des Betriebssystems gespeichert. In der Textspalte wird der Dateiname gespeichert	*path* gibt an, wo das externe Textfile gespeichert ist. Es ist möglich, mehrere Pfade mit Semikolon getrennt anzugeben. Die Dateinamen werden im Textfeld in der Tabelle gespeichert. Wird kein Pfad für externe Files über "path" angegeben, so muß dieser im Textfeld mit eingetragen werden.
URL_DATASTORE	Die Daten werden extern im Internet / Intranet gespeichert. Jede URL kann in einem extra Textfeld gespeichert werden.	*timeout* in Sekunden (mögl.: 15...3600s, Default: 30) *maxthreads* Anzahl der Threads, die parallel laufen können (möglich: 1..1024, voreingestellt: 8) *Urlsize* Länge der URL in Bytes, die der URL-Dateinspeicher unterstützt (möglich:

URL_DATASTORE		32...65535, voreingestellt: 256) *Maxurls* Die maximale Anzahl von Spalten, die der interne Puffer für HTML-Dokumente enthalten kann, welche aus der Texttabelle stammen (möglich: 32...65535, voreingestellt: 256) *urlsize maxurls* < 5MB --> Speicherpuffer für das URL Objekt hat eine maximale Größe von 5MB. *Maxdocsize* maximale Größe in Bytes, die der URL-Datenspeicher für HTML-Dokumente unterstützt (möglich: 1Byte...2GByte, voreingestellt: 2MB) *http_proxy* Name des Gateways für HTTP, für Authentifizierung im Intranet nötig *ftp_proxy* Name des Gateways für FTP, für Authentifizierung im Intranet nötig *no_proxy* String; an alle Maschine die hier aufgelistet werden, werden keine Anfragen für http_proxy oder ftp_proxy geschickt (max. 16 Rechner)
USER_DATASTORE	wird benutzt, um "stored procedures" zu definieren, die Dokumente während des Indizierens synthetisieren wird z.B. benutzt, um zusätzliche Informationen in den Text einzufügen	*Procedure* Name der Prozedur, Parameter (IN ROWID, IN OUT CLOB) Die Prozedur wird für jede zu indizierende Zeile aufgerufen. CTXSYS muß "Besitzer" der Prozedur sein, der "Indexbesitzer" muß sie ausführen können. Die Prozedur darf keine DDL oder Transaktionskontrollaufrufe (wie COMMIT) verwenden.

2.2 Die SQL –Insert Anweisung

Ein einfacher Weg Daten zu laden besteht darin eine Tabelle mit zwei Spalten (id, text) anzulegen und diese dann mit dem insert Kommando zu füllen.

Beispiel:

Create table docs (id number primary key , text varchar2 (80));

Insert into docs values (1 , ' this isthe text in the first dokument ');
Insert into docs values (2 , ' this isthe text in the second dokument ');

2.3 SQL – Loader

Das folgende Beispiel lädt Dateien mit verschieden Formaten in eine Tabelle mit eine Textspalte von Typ BLOB.

Anlegen der Tabelle:

```
CREATE TABLE articles_formatted
(
        ARTICLE_ID NUMBER PRIMARY KEY ,
        AUTHOR VARCHAR2(30),
        FORMAT VARCHAR2(30),
        PUB_DATE DATE,
        TITLE VARCHAR2(256),
        TEXT BLOB
);
```

Lesen des Kontrollfiles

```
sqlldr userid=demo/demo control=loader1.dat log=loader.log
```

Inhalt des Kontrollfiles:

```
INFILE 'loader2.dat'
INTO TABLE articles_formatted
APPEND
FIELDS TERMINATED BY ','
(article_id SEQUENCE (MAX,1),
author CHAR(30),
format,
pub_date SYSDATE,
title,
ext_fname FILLER CHAR(80),
text LOBFILE(ext_fname) TERMINATED BY EOF)
```

Diese Kontrollfile weist dem Loader die Informationen zu ,welche zum Laden von LOADER2.DAT in die Tabelle benötigt werden.
Es werden folgende Schritte ausgeführt:

1. Die Felder in LOADER2.DAT sind in der *article_id* Spalte beschrieben.
2. Das erste Feld wird in die *author* Spalte geschrieben.
3. das zweite Feld wird in die *format Spalte geschrieben.*.
4. Das aktuelle Datum wird aus SYSDATE gelesen und in die *pub_date Spalte geschrieben*.
5. das dritte Feld wird in die *title* Spalte geschrieben.
6. Das aktuelle Dokument wird in die *text* BLOB Spalte geladen:

Beispiel LOADER2.DAT

```
Ben Kanobi, plaintext,Kawasaki news article,../sample_docs/kawasaki.txt,
Joe Bloggs, plaintext,Java plug-in,../sample_docs/javaplugin.txt,
John Hancock, plaintext,Declaration of Independence,../sample_docs/indep.txt,
M. S. Developer, Word7,Newsletter example,../sample_docs/newsletter.doc,
M. S. Developer, Word7,Resume example,../sample_docs/resume.doc,
X. L. Developer, Excel7,Common example,../sample_docs/common.xls,
```

2.4 ctxload

ctxload stellt folgende Operationen zur Verfügung:

- Thesaurus Import und Export
- Laden von Texten
- Dokument Update / Export

➢ *Thesaurus Import und Export*

Benutzt ctxload zum laden von Thesaurusdaten aus einem File in die iMT Thesaurus Tabelle.
Als Importfile dient ein ASCII File das Einträge für Synonyms, broader Terme , narrower Terme, or related Terme welche zum erweitern von Anfragen benutzt werden können. ctxload kann auch zum exportieren eines Thesaurus durch Ausgabe des Inhaltes in eine Datei verwendet werden.

➢ *Laden von Text*

ctxload kann auch zum laden eines Textes aus einer Datei in eine LONG or LONG RAW Spalte einer Tabelle benutzt werden. Als Importfile dient ein ASCII File welches den Text enthält. Für strukturierte Daten muß ein File mit den entsprechenden Headerdaten vorhanden sein.

➢ *Dokument Update / Export*

Das ctxload utility unterstützt das aktualisieren von datenbankspalten aus Dateien (normale Files) Sowie das exportieren von datenbankspalten (LONG RAW, LONG, BLOB and CLOB Spalten) in Dateien.

Beispiele :

Thesaurus Import
Das folgende Beispiel importiert ein Thesaurus (*tech_doc) aus der Datei tech_thesaurus.tx*t:

ctxload -user jsmith/123abc -thes -name tech_doc -file tech_thesaurus.txt

Thesaurus Export
Das folgende Beispiel exportiert den Inhalt des Thesaurus (*tech_doc) in die Datei*

*tech_thesaurus.tx*t:

ctxload -user jsmith/123abc -thesdump -name tech_doc -file tech_thesaurus.out

Exportieren eines einzelnen Textfeldes

Das folgende Beispiel exportiert ein einzelnes Textfeld , welches durch einen Primärschlüsselwert = 1 gekennzeichnet ist in die Datei *myfil*e. Der Index *myindex zeigt die Textspalte an.*

ctxload -user scott/tiger -export -name myindex -file myfile -pk 1

oder , wenn der Primärschlüssel nicht numerisch ist …

ctxload -user scott/tiger -export -name myindex -file myfile -pk "Oracle,1"

Updating a Single Text Field

Das folgende Beispiel aktualisiert ein einzelnes Textfeld , welches durch einen Primärschlüsselwert = 1 gekennzeichnet ist mit dem Inhalt der Datei *myfil*e.
Der Index *myindex zeigt die Textspalte an.*

ctxload -user scott/tiger -update -name myindex -file myfile -pk 1

oder , wenn der Primärschlüssel nicht numerisch ist …

ctxload -user scott/tiger -update -name myindex -file myfile -pk "Oracle,1"

2.5 DBMS_LOB.LOADFROMFILE() PL/SQL Procedure

Für mehr Informationen über die DBMS_LOB package, siehe :
Oracle8I Supplied Packages Reference.

2.6 Unterstützte Filterformate – Inso Filter

Die von Oracle8i *inter*Media Text benutzte Dokument Filter Technologie wurde von der Inso Corporation lizensiert
Diese Filtertechnologie ermöglicht das indizieren der meisten Dokumentenformate.
Weiterhin ist das konvertieren z.B. in HTML möglich.
Um mit dem Filter arbeiten zu können muß ein entsprechendes Inso Filter Objekt in der Filter Beschreibung angegeben sein.
Um Dokumente in das HTML Format zu konvertieren wird die Inso Filter Index Beschreibung nicht benötigt. Aber die Umgebungsvariablen für den Inso Filter müssen korrekt eingestellt sein.

Hier einige der unterstützten Formate:

- *Generische Formate*

ASCII Text (7 &8bit versions)	Alle Versionen
ANSI Text (7 & 8 bit)	Alle Versionen
Unicode Text	Alle Versionen
HTML	Alle Versionen
IBM Revisable Form Text	Alle Versionen
IBM FFT	Alle Versionen bis 3.0
Microsoft Rich Text Format (RTF)	Alle Versionen

- *DOS Formate*

IBMWriting Assistant	Version 1.01
Lotus Manuscript	Versionen bis 2.0
Microsoft Word	Versionen bis 6.0
Microsoft Works	Versionen bis 2.0
Office Writer	Versionen 4.0 bis 6.0
WordPerfect	Versionen bis 7.0
WordStar	Versionen bis 7.0
WordStar 2000	Versionen bis 3.0
XyWrite	Versionen bis III Plus

- *Internationale Formate*

Ichitaro	Version 8

- *Windows Formate*

Corel WordPerfectfor Windows	Versionen bis 8.0
Lotus WordPro	
Lotus SmartSuite	
Microsoft Windows Works	Versionen bis 4.0
Microsoft Windows Write	Versionen bis 3.0
Microsoft Word 97 Word 97	
Microsoft Word 2000 Beta 2	
Microsoft Word for Windows	Versionen bis 7.0
Microsoft WordPad	Alle Versionen
Novell Perfect Works	Version 2.0
Novell WordPerfect for Windows	Versionen bis 7.0
WordStar for Windows	Version 1.0

Text Datenbanken

Oracle*8i inter*Media Text

Kapitel 3

Indizieren von Dokumenten

3. Indizieren von Dokumenten

CREATE INDEX bzw. ALTER INDEX erlauben das verändern von verschiedenen Voreinstellungen bzw. Optionen.
Diese beeinflussen die Art und Weise, wie Oracle Text indiziert.

3.1 Vorgaben / Voreinstellungen

Voreinstellungen für:

Datastore	*Wie sind die Dokumente gespeichert ?*
Filter	*Wie können die Dokumente in puren Text umgewandelt werden ?*
Lexer	*Welche Sprache soll verwendet werden ?*
Wordlist	*Wie sollen stem and fuzzy queries erweitert werden ?*
Storage	*Wie sollen die Indextabellen gespeichert werden ?*
Stop List	*Welche Wörter und Themen sollen nicht indiziert werden ?*
Section Group	*Sind Anfragen in den Abschnitten eines Dokumentes möglich, und wie werden die Dokumentenabschnitte definiert ?*

Erstellen von Voreinstellungen:

Erstellen von Datastore, lexer, Filter, wordlist und storage Voreinstellungen:	CTX_DDL.CREATE_PREFERENCE
Setzen von Attributen:	CTX_DDL.SET_ATTRIBUTE
Erstellen von Stoplisten:	CTX_DDL.CREATE_STOPLIST
Erstellen von section groups:	CTX_DDL.CREATE_SECTION_GROUP

3.2 Filter Objekte:

Filter Objekte legen fest, wie Text für die spätere Indizierung gefiltert wird.
Sie erlauben Textverarbeitungsdokumente, formatierte Dokumente, puren Text und HTML.
Formatierte Dokumente speichert Oracle in ihrer Originalform, und benutzt Filter, um zwischenzeitlich puren Text oder HTML-Code zu erzeugen. Oracle indiziert dann diese temporären Texte.

mögliche Objekte:

- *NULL_FILTER - ASCII-Filter:*
  Es wird purer Text gespeichert, eine Filterung ist nicht nötig.

- *INSO_FILTER - INSO-Filter für formatierte Dokumente:*
  Dieser Filter ist ein universeller Filter, er dient zum indizieren von gleich oder gemischt formatierten Texten. Formatliste ...

- *USER_FILTER - benutzerdefinierter Filter:*
  Dieser Filter wird verwendet, um einen externen (selbstgeschriebenen) Filter einsetzten zu können. Das "filter executable" ist anzugeben. (muß sich in $ORACLE_HOME/ctx/bin/ befinden) Diese Filterart ist sorgfältig zu verwenden.

- *CHARSET_FILTER - Filter zum konvertieren von Zeichensätzen:*
  Dieser Filter wird eingesetzt, um das Dokument von einem bestimmten Zeichensatz in den Zeichensatz der Datenbank zu konvertieren. (Japanische Texte konvertieren ...)

3.3 Lexer Objekte:

Lexer Objekte werden verwendet, um die Sprache des Textes festzulegen

mögliche Objekte:

- BASIC_LEXER:
  Lexer, der benutzt wird, um einzelne Zeichen aus dem Text zu extrahieren. Wird für Englisch und die meisten westeuropäischen Sprachen verwendet die "single-byte" Zeichensätze verwenden.
  Attribute:

continuation	numgroup	numjoin	printjoins	punctuations
skipjoins	startjoins	endjoins	whitespace	newline
base_letter	mixed_case	composite	index_themes	index_text
alternate_spelling				

continuation - characters (string)
- spezifiziert Zeichen, die angeben, das es auf der nächsten Zeile mit dem gleichen Wort weitergeht ("-", "/")

numgroup
- spezifiziert Zeichen, die in einer Anzahl von Ziffern auftreten, um Zahlen besser lesbar zu machen "," bei: 1,000,000.00

numjoin
- wenn diese Zeichen auftreten trennt Oracle die Zahl nicht, sondern indiziert sie als Ganzes "." bei: 1000.00

printjoins
- spezifiziert nicht-alphanumerische Zeichen; wenn diese in einem Wort auftreten (Anfang, Mitte oder Ende) so werden sie mit dem Word indiziert
  "_" ist definiert als printjoin, so wird "_file_" auch als "_file_" indiziert (nicht "file")

punctuations
- spezifiziert nicht-alphanumerische Zeichen; wenn diese am Ende eines Wortes auftreten, wird das als Satzende erkannt
  ".", "?", "!"
- werden vor dem indizieren entfernt
- ist das gleiche Zichen als printjoin definiert, so wird es nur am Ende entfernt, nicht mittendrin
  d.h "text.doc" bleibt "text.doc" ; " ... dog." wird "dog"

skipjoins
- spezifiziert nicht-alphanumerische Zeichen; wenn diese in einem Wort auftreten, werden sie entfernt
  d.h. "pseudo-intellectual" wird "pseudointellectual"

startjoins
- diese Zeichen geben explizit den Start eines Token an
- diese Zeichen dürfen nicht in anderen Attributen auftreten

endjoins
- diese Zeichen geben explizit das Ende eines Token an
- diese Zeichen dürfen nicht in anderen Attributen auftreten

whitespace
- "space", "tab" zwischen den Token
- im Zusammenhang mit "punctuation" und "newline" als Satztrenner

newline
- im Zusammenhang mit "punctuation" und "whitespace" als Satz- und Abschnittstrenner

base_letter
- gibt an, ob bestimmte Zeichen wie z.B. Umlaute in ihre Basisform umgewandelt werden
- default: NO

mixed_case
- gibt an, ob alle Zeichen exakt wie im Text gespeichert werden, oder ob sie in Großbuchstaben umgewandelt werden
- default: NO (alles Großbuchstaben)

composite
- gibt an, ob zusammengesetzte Wörter als Ganzes indiziert werden, oder nur die einzelnen Teile
- für Deutsch
- default: NO

index_themes
- nur für Englisch
- "YES", wenn Themeninformationen indiziert werden sollen

index_text
- "YES", um Wortinformationen zu indizieren

index_themes und index_text können nicht beide "NO" sein.

alternate_spelling
- ermöglicht alternative Schreibweisen von einigen Worten
- in deutsch, dänisch und schwedisch
- default: enabled

- CHINESE_VGRAM_LEXER:
  Lexer, der benutzt wird um Zeichen aus chinesischem Text zu extrahieren.

- JAPANESE_VGRAM_LEXER:
  Lexer, der benutzt wird um Zeichen aus japanischem Text zu extrahieren.

- KOREAN_LEXER:
  Lexer, der benutzt wird um Zeichen aus koreanischem Text zu extrahieren.

3.4 Wordlist Objects:

Wordlist Objekte werden verwendet, um erweiterte Anfrageoptionen zu ermöglichen

mögliche Objekte: BASIC_WORDLIST für "stemming" und "fuzzy queries"

Die zu benutzenden Sprache muß angegeben werden:
NULL, ENGLISH, DERIVATIONAL (English), DUTCH, FRENCH, GERMAN, ITALIAN, SPANISH

Bei "NULL" wird der operator "$" ignoriert.

"fuzzy_match"
- GENERIC, ENGLISH, GERMAN, OCR ...
- gibt an, welche Routinen eingesetzt werden, default ist "GENERIC"
- unterstützt in Englisch, einigen westeuropäischen Sprachen und Japan

"fuzzy_score"
- gibt ein unteres Limit des "fuzzy score" an (Werte zwischen 0 ... 5000)
- keine Scores unter diesem Limit werden erzeugt

"fuzzy_numresults"
- gibt die maximale Anzahl der Erweiterungen der Anfrage an (Werte zwischen 0 ... 5000)

3.5 Storage Objects:

Storage Objects werden verwendet, um die Tabellengrösse und die Erstellungsparameter einer Tabelle anzugeben, die mit einem Textindex verbunden ist

mögliche Objekte: BASIC_STORAGE

i_table_clause
- bei CREATE TABLE, Indexdatentabelle

k_table_clause
- bei CREATE TABLE, Schlüsselworttabelle

r_table_clause
- bei CREATE TABLE, rowid - Tabelle

n_table_clause
- bei CREATE TABLE, "negative list" - Tabelle

i_index_clause
- bei CREATE INDEX

3.6 Section Group Types:

mögliche Typen:

NULL_SECTION_GROUP
- default
- nur Sätze und Abschnitte im Text

BASIC_SECTION_GROUP
- für Textteile, die Start- und Endtags der Form <A> ... </A> haben

HTML_SECTION_GROUP
- Abschnitte in HTML - Dokumenten

XML_SECTION_GROUP
- für Dokumente mit XML - Tags

NEWS_SECTION_GROUP
- für Dokumente mit Newsgroup-Format

Text Datenbanken
Oracle*8i interMedia* Text

Kapitel 4

Stellen von Anfragen

4. Anfragen

Es werden grundsätzlich zwei Arten von Anfragen unterschieden, Wort Anfragen und about-Anfragen.

4.1 Wort - Anfragen

Eine word - query ist eine von einfachen Hochkommas eingeschlossenen Anfrage, nach einem bestimmten Wort oder einer Phrase.
Es können AND , OR Operatoren sowie strukturierte Prädikate benutzt werden.
Dieses Wort steht in Hochkommas im CONTAINS Operator.
Die Treffer der Anfrage können durch benutzen von count(*), CTX_QUERY.COUNT_HITS oder CTX_QUERY.EXPLAIN gezählt werden.

CTX_QUERY.COUNT_HITS

Gibt die Anzahl der Treffer einer Anfrage zurück.
COUNT_HITS kann im Exact mode oder im Estimate mode Modi ausgeführt werden:

Exact mode	:	Gibt die exakte Anzahl der Treffer einer Anfrage zurück
Estimate mode	:	Gibt nur eine ungefähre Schätzung zurück , ist dafür aber deutlich schneller.

Syntax:

```
COUNT_HITS (
        index_name IN VARCHAR2,
        text_query IN VARCHAR2,
        exact IN BOOLEAN DEFAULT TRUE
                ) RETURN NUMBER;
```

Elemente:

index_name	:	index name.
text_query	:	Die Anfrage
Exact	:	TRUE für exact mode. FALSE für eine Schätzung

4.2 About - Anfragen

In allen Sprachen zählen ABOUT Anfragen die Anzahl von relevanten Dokumenten welche die Kriterien einer gestellten Anfrage erfüllen.
In englischer Sprache ist es möglich die Themen - Komponente des Index zu benutzen.
(wird benötigt) Diese erlaubt es Dokumente zu finden die konzeptionell mit der gestellten Anfrage beschäftigen.

Syntax: ABOUT (Phrase)

Die Phrase kann ein einzelnes Wort oder ein Satz bzw. eine wahllose Aneinanderreihung von Wörtern sein.

Beispiele:

Einfaches Wort: ' about (Database) '

Wortphrasen: ' about(soccer rules in international competition) '

In Englischer Sprache werden alle Dokumente zurückgegeben die soccer , rules oder international competition enthalten.
Enthält ein Dokument alle Wörter wird dieses höher Bewertet als andere Dokumente.

Nichtstrukturierte Wortphrasen

```
'about(japanese banking investments in indonesia)'
```

4.3 Gemischte Anfragen

Es ist Grundsätzlich möglich die beiden Anfragetypen zu mischen.
Dies geschieht durch Verwendung von anderen Operatoren wie AND oder OR.
Allerdings kommt es zu einem Fehler wenn ein Index nur Themeninformationen enthält und der ABOUT Operator sowie der Operand in der selben Anfrage stehen müssen, die ein und die selbe Textspalte betrifft.

ABOUT und word Anfrage:

' about(dogs) and cat '

ABOUT Anfrage mit einer anderen ABOUT Anfrage:

' about (dogs) not about(labradors) '

4.4 Operator Prioritäten

Operator Prioritäten beschreiben die Reihenfolge in der die Komponenten einer Anfrage bearbeitet werden. Textanfragen Operatoren können in zwei Gruppen geteilt werden.
Die Prioritäten entsprechen der Reihenfolge der Aufzählung.

Gruppe 1 (Operatoren)

EQUIValence (=)
NEAR (;)
weight (*), threshold (>)
MINUS (-)
NOT (~)
WITHIN
AND (&)
OR (|)
ACCUMulate (,)

Gruppe 2 (Operatoren und Zeichen)

Wildcard Characters
ABOUT
stem ($)
fuzzy (?)
soundex (!)

Andere , hier nicht aufgeführte Operatoren sind Prozedurale Operatoren. Bei diesen Operatoren macht es keinen Sinn Prioritäten zu vergeben.
Ein Beispiel dafür ist zum Beispiel ein Thesaurus Operator

Im Allgemeinen werden Anfrage Operatoren von links nach rechts abgearbeitet.
Operatoren mit höheren Prioritäten werden zuerst abgearbeitet.

Beispiele für Prioritäten von Operatoren

Anfrage	Reihenfolge der Bearbeitung
w1 | w2 & w3	(w1) | (w2 & w3)
w1 & w2 | w3	(w1 & w2) | w3
?w1, w2 | w3 & w4	(?w1), (w2 | (w3 & w4))
abc = def ghi & jkl = mno	((abc = def) ghi) & (jkl=mno)
dog and cat WITHIN body	dog and (cat WITHIN body)

Im ersten Beispiel hat UND eine höhere Priorität als ODER
Damit werden alle Dokumente zurückgegeben die W1 entsprechen und alle Dokumente die W2 und W3 enthalten

Im zweiten Beispiel zwei werden alle Dokumente zurückgegeben die W1 und W2 enthalten und alle Dokumente die W3 enthalten.

Das dritte Beispiel fuzzy zu W1 dann W3 und W4 zu UND danach W2 zu ODER und zum Ergebnis von UND zum Schluß die Bewertung des fuzzy von W1 zu ODER

Das vierte Beispiel zeigt das der Äquivalenz Operator eine höhere Priorität hat als der UND Operator .

WITHIN hat laut dem fünften Beispiel eine höhere Priorität als der UND Operator.

4.5 Spezielle Zeichen in Anfragen

- *Wildcard Zeichen*

  % Das Prozentzeichen sagt aus, das irgendein Zeichen an dieser Stelle, welche durch das % beschrieben wird, auftreten kann. (auch mehrmalig)

  _ Der Unterstrich sagt aus das irgendein einzelnes Zeichen an dieser Stelle auftreten kann.

  'scal%' findet alle Terme die mit dem Wort *scal* beginnen

  Um alle Worte die auf ' ing ' enden und zufuhr nur ein Zeichen haben zu finden muß *'_ing'* geschrieben werden

- *Gruppierende Zeichen*

  ( ) Klammern dienen zum gruppieren von Termen und Operatoren

  [ ] Klammern dienen zum gruppieren von Termen und Operatoren. Sie verhindern das Eindringen von expansions Operatoren wie fuzzy oder stem

- *Escape Zeichen*

  Für Anfragen nach speziellen Wörtern oder Symbolen oder auch nach reservierte Wörter oder Zeichen müssen diese als Escape - Sequenz angegeben werden.
  (and - & , or |)

  Es gibt zwei Möglichkeiten dies zu tun.

  {} Benutze geschwungene Klammern um eine String von Zeichen und Symbolen als Escape Sequenz zu kennzeichnen.

  \ Mit dem Slash wird ein einzelnes Zeichen als Escape – Sequenz gekennzeichnet.

  Beispiele

  'AT\&T' oder *'{AT&T}'*

  'high\-voltage' oder *'{high-voltage}'*

4.6 Weitere Anfrage Eigenschaften

In Anfragen können auch andere Anfrageeigenschaften benutzt werden.
Hier ein Überblick über weitere Möglichkeiten.

Eigenschaft	Beschreibung
Sektionen Suche	Zum definieren von Sektionen für Anfragen
Proximity Suche	Verwenden des NEAR Operators Mehrere Anfrage Terme werden bewertet. Näher zusammenliegende Terme werden höher bewertet.
Thesaural Suche	benutzen von Thesaurus Objekten
Stem and Fuzzy Suche	Stem und Fuzzy Operatoren verwenden Benutzt den Stem ($) Operator zum finden von Termen die den selben Wortstamm wie der Anfrageterm haben Benutze den fuzzy (?) Operator um Anfragen durch Worte zu erweitern die ähnlich gesprochen werden. Schreibfehler in der Datenbank werden somit umgangen.
Case Sensitive Suche	Lexer Objekte benutzen
Base Letter Conversion	Lexer Objekte benutzen
Word Decompounding	Lexer Objekte benutzen
Alternate Spelling	Lexer Objekte benutzen
Query Explain Plan	CTX.QUERY.EXPLAIN Prozedur benutzen um eine explain plan Information für eine Anfrage zu erstellen
Hierarchical Query Feedback	CTX_QUERY.HFEEDBACK Prozedur benutzen um eine Feedback Information zu erstellen

Beispiel Sectionensuche:

Der folgende Code definiert eine Section group *basicgroup welche vom* BASIC_SECTION_GROUP Typ ist.
Eine Feldsektion author wird angelegt. Das visible Flag wird auf FALSE gesetzt.

```
begin
ctx_ddl_create_section_group('basicgroup', 'BASIC_SECTION_GROUP');
ctx_ddl.add_field_section('basicgroup', 'Author', 'A', FALSE);
end;
```

Wenn das author Feld nicht sichtbar gesetzt ist muß der WITHIN Operator benutzt werden um Text in der Sektion zu finden.

```
'(Martin Luther King) WITHIN Author'
```

Eine Anfrage nach *Martin Luther King* ohne den WITHIN Operator wird keine Instanz des Termes zurückgeben. Eine Anfrage ohne den WITHIN Operator wird nur funktionieren wenn das visible Flag auf TRUE gesetzt wird.

```
begin
ctx_ddl.add_field_section('basicgroup', 'Author', 'A', TRUE);
end;
```

Text Datenbanken
Oracle*8i interM*edia Text

Kapitel 5

Dokumentenpräsentation

5. Dokumentenpräsentation

In den meisten Fällen in denen der Benutzer eine Anfrage gestellt hat, möchte er nicht nur das Ergebnis der Anfrage, sondern auch die dazugehörigen Dokumente sehen in dehnen sich die Treffer befinden.

Normalerweise wird eine Liste von Dokumenten die die gestellte Anfrage erfüllen angezeigt.
Der Benutzer wählt ein Dokument aus, was angezeigt werden soll.
Mittels interMedia Text konnen Dokumente auf verschiedene Art und Weise angezeigt werden.
Zum Beispiel können die Suchworte der Anfrage unterstrichen werden.

5.1 Vorhandene Prozeduren

zur Verfügung stehen folgende Prozeduren:

CTX_DOC.MARKUP
Dokument mit hervorgehobenen Textteilen in "plain text"-Version und HTML-Version

CTX_DOC.HIGHLIGHT
"offset information", "plain text"-Version und HTML-Version

CTX_DOC.FILTER
"plain text"-Version und HTML-Version, keine Hervorhebungen

5.2 CTX_DOC:

CTX_DOC ist ein PL/SQL Package und stellt Dokumentenservices zusammen

verfügbare Funktionen:

FILTER	generiert eine Version mit purem Text oder ein HTML-File des Dokumentes
GIST	generiert eine Zusammenfassung der Themen des Dokumentes
HIGHLIGHT	generiert eine Version mit purem Text oder ein HTML-File des Dokumentes mit "offset information"
MARKUP	generiert eine Version mit purem Text oder ein HTML-File des Dokumentes, die Anfragen werden hervorgehoben
PKENCODE	erzeugt einen Text-String, der in anderen CTX_DOC - Proceduren benutzt werden kann
THEMES	generiert eine Liste der Themen des Dokumentes

- *Beispiel: CTX_DOC.FILTER:*

```
CTX_DOC.FILTER(
        index_name IN VARCHAR2,                 - Name des Index
        textkey IN VARCHAR2,                    - eindeutiger Indentifizierer für das Dokument
                                                  (primary key)
        restab IN VARCHAR2,                     - Name der Ergebnisstabelle; Wo soll das
                                                  gefilterte Dokument gespeichert werden ?
        query_id IN VARCHAR2 DEFAULT 0,         - Indentifizierer, Zeile in restab
        plaintext IN BOOLEAN DEFAULT FALSE);    - TRUE - "plain text"; FALSE – HTML
```

Erzeugen der Ergebnistabelle für den Filter:

```
create table filtertab (query_id number, document clob);
```

Erzeugen einer Version des Dokumentes mit dem Textschlüssel 20, mit purem Text:

```
begin
   ctx_doc.filter('newsindex', 20, 'filtertab', 0, TRUE);
end;
```

- *Beispiel - CTX_DOC.MARKUP:*

```
CTX_DOC.MARKUP
(
   index_name IN VARCHAR2,                      - Name des Index
   textkey IN VARCHAR2,                         - eindeutiger Indentifizierer für das Dokument
                                                  (primary key)
   text_query IN VARCHAR2,                      - Orginalanfrage, bei der dieses Dokument
                                                  gefunden wurde
   restab IN VARCHAR2,                          - Name der Ergebnisstabelle; Wo soll das
                                                  gefilterte Dokument gespeichert werden ?
   query_id IN VARCHAR2 DEFAULT 0,              - Indentifizierer, Zeile in restab
   plaintext IN BOOLEAN DEFAULT FALSE,          - TRUE - "plain text"; FALSE - HTML
   tagset IN VARCHAR2 DEFAULT 'TEXT_DEFAULT',   - vordefinierte Tagsets Text_DEFAULT,
          HTML_DEFAULT ( <B> , </B>), HTML_NAVIGATE
   starttag IN VARCHAR2 DEFAULT NULL,           - Zeichen, die eingefügt werden sollen, um
                                                  anzuzeigen, das ein hervorgehobener Text
                                                  beginnt
   endtag IN VARCHAR2 DEFAULT NULL,             - Zeichen, die eingefügt werden sollen, um
                                                  anzuzeigen, das ein hervorgehobener Text
                                                  endet
   prevtag IN VARCHAR2 DEFAULT NULL,            - "markup sequence" um zum
                                                  vorhergehenden hervorgehobenen Text zu
                                                  springen
   nexttag IN VARCHAR2 DEFAULT NULL,            - "markup sequence" um zum nächsten
                                                  hervorgehobenen Text zu springen
);
```

- *Beispiel - CTX_DOC.HIGHLIGHT:*

Erzeugen der Ergebnistabelle für Highlighting:

```
create table markuptab (query_id number, document clob);
```

Erzeugen einer Version des Dokumentes mit den hervorgehobenen Wörtern "dog" und "cat", Schlüssel 23, HTML-Version:

```
begin
   ctx_doc.markup(index_name => 'my_index',
                  textkey => '23',
                  textquery => 'dog|cat',
                  restab => 'markuptab',
                  query_id = '1',
                  tagset v => 'HTML_DEFAULT');
end;
```

Text Datenbanken
Oracle*8i interMedia* Text

Kapitel 6

Vergleich mit SQL - MMText

6. Vergleich des Standards SQL99 (SQL/MM FullText) mit der Funktionalität und den Möglichkeiten von Oracle

6.1 Allgemeine Bemerkungen

Oracle interMedia Text unterstützt wie im Standart vorgesehen neue Suchmechanismen, denn normale Suchmechanismen wie "like" sind nicht leistungsfähig genug

- **es werden neue strukturelle Einheiten genutzt:**

  1. Wörter
  2. Sätze
  3. Abschnitte

  Im Standart gefordert, in Oracle umgesetzt.

- **es werden neue Operationen genutzt:**

  1. "Boolean Search", bedingt in Oracle umgesetzt, da Oracle keine Boolean-Ruckgabewerte akzeptiert
  2. "Ranking"
  3. "Conceptual Search"

  Im Standart gefordert, in Oracle umgesetzt.

- **die Sprache des Dokuments wird erfaßt**

  Im Standart gefordert, in Oracle umgesetzt.

- **es gibt neue Anfrage- ("Boolean query") und Bearbeitungsmöglichkeiten:**

  1. Einzelwortsuche
  2. Suche von Phrasen
  3. inhaltsbasierte Suche
  4. Stopwort Verarbeitung
  5. Filtermöglichkeiten
  6. erweiterte Suche (fuzzy Search, Synonyms, ...)

  Im Standart gefordert, in Oracle umgesetzt.

6.2 Beispiele:

- Single word Search
- Conceptual Search / About Query
- Phrase Search
- Kontext Suche
- Ranking

➢ *Single word search*

(Standart):

*SELECT * FROM myDocs WHERE 1 = CONTAINS(TextBody, '"specific"')*

(Oracle):

SELECT SCORE(1) title FROM news WHERE CONTAINS(text, 'oracle', 1) > 0;

➢ *Conceptual search / About Query*

(Standart):

*SELECT * FROM myDocs WHERE 1 = CONTAINS(TextBody, 'IS ABOUT "every text ..."')*

(Oracle):

SELECT SCORE(1) title FROM news WHERE CONTAINS(text, 'about(politics)', 1) > 0;
setzt englische Sprache voraus

➢ *Phrase search*

(Standart):

*SELECT * FROM myDocs WHERE 1 = CONTAINS(TextBody, '"specific language"')*

(Oracle):

SELECT employee_name, SCORE(1) FROM employee_database WHERE CONTAINS (emp.resume, 'software developer') > 0

- *Context search*

(Standart):

*SELECT * FROM myDocs WHERE 1 = CONTAINS(TextBody, '"text IN SAME SENTENCEAS language"')*

(Oracle):

in Oracle nicht möglich

- *Ranking*

(Standart):

*SELECT * FROM myDocs WHERE 1.2 < RANK(TextBody, '"specific"')*

(Oracle):

"RANK" existiert nicht
nur mit Umweg über SCORE(xxx) und sortieren bewerkstelligbar

WHERE CONTAINS (..) > 0 ORDER BY NVL(SCORE(10), 0)

Darüber hinaus bietet Oracle weitere Optionen, siehe "Query Operators / Anfragen".
z.B.: soundex, stem, weight, ...

Anhang A

Ausgeführtes Beispiel

```
drop table quick;

create table quick
  (
    quick_id               number
    constraint quick_pk     primary key,
    text                   varchar(80)
  );

insert into quick ( quick_id, text )  values ( 1, 'The cat sat on the mat' );
insert into quick ( quick_id, text )  values ( 2, 'The quick brown fox jumps over the lazy dog' );
insert into quick ( quick_id, text )  values ( 3, 'The dog barked like a dog' );
commit;

select * from quick;
            execute ctx_ddl.drop_preference('osdb08_datastore');
            execute ctx_ddl.drop_preference('osdb08_lexer');
            execute ctx_ddl.drop_preference('osdb08_wordlist');
            execute ctx_ddl.drop_stoplist  ('osdb08_stoplist');
            execute ctx_ddl.create_preference('osdb08_datastore','DIRECT_DATASTORE');
            execute ctx_ddl.create_preference('osdb08_lexer', 'BASIC_LEXER');
            execute ctx_ddl.set_attribute    ('osdb08_lexer', 'INDEX_TEXT', 'YES');
            execute ctx_ddl.set_attribute    ('osdb08_lexer', 'INDEX_THEMES', 'NO');
            execute ctx_ddl.create_preference('osdb08_wordlist', 'BASIC_WORDLIST');
            execute ctx_ddl.set_attribute    ('osdb08_wordlist', 'STEMMER','GERMAN');
            execute ctx_ddl.create_stoplist  ('osdb08_stoplist');
            execute ctx_ddl.add_stopword     ('osdb08_stoplist', 'und');

drop index quick_text;

create index quick_text on quick ( text ) indextype is ctxsys.context
parameters ( 'datastore osdb08_datastore lexer osdb08_lexer filter CTXSYS.NULL_FILTER
                    wordlist osdb08_wordlist stoplist osdb08_stoplist' );

commit;

col text format a45
col s format 999
select text, score(42) s from quick
  where contains ( text, 'dog', 42 ) >= 0
    order by s desc;



-- TEXT                                          S
-- --------------------------------------------- ----
-- The dog barked like a dog                        7
-- The quick brown fox jumps over the lazy dog      4
-- The cat sat on the mat                           0
```