Intelligent zoeken met Oracle Text

Functionaliteit voor de database en het web

Met behulp van Oracle Text (voorheen Oracle interMedia Text, en daarvoor Oracle Context) kan zogenaamde full text search functionaliteit geïmplementeerd worden. Hieronder wordt het snel (en intelligent) doorzoeken van tekst verstaan. Deze teksten kunnen complete documenten zijn, opgeslagen in de database zelf, in een file system of op het web, of (kortere) teksten opgeslagen in een database-kolom. In dit artikel beschrijft Erwin Groenendal stapsgewijs de ontwikkeling van een applicatie die gebruik maakt van Oracle Text.

Oracle Text is volledig geïntegreerd in de Oracle database en bestaat onder andere uit een aantal speciale indextypen, PL/SQL database packages en speciale SQL operatoren. In producten zoals Oracle Portal en Oracle Collaboration Suite en Oracle Files wordt gebruik gemaakt van Oracle Text. Maar deze krachtige functionaliteit kan ook benut worden in maatwerkapplicaties. Oracle Text is beschikbaar in zowel de standard- (SE) als de enterprise edition (EE) van de Oracle database.

Toepassingen

Oracle Text is vooral goed bruikbaar in content management toepassingen, waarbij grote hoeveelheden documenten efficiënt doorzocht moeten worden. Denk hierbij bijvoorbeeld aan een verzameling documentatie, white papers of conferentie papers waarin de meest relevante documenten gevonden moeten worden met betrekking tot een aantal zoektermen of een bepaalde zoekvraag. Ook het zoeken naar relevante HTML pagina's binnen bijvoorbeeld de intranet sites van een bedrijf of organisatie is een toepassing waarvoor Oracle Text zeer geschikt is.

Naast het doorzoeken van complete documenten kan Oracle Text ook toegepast worden voor het zoeken binnen grote tekstvelden in administratieve toepassingen. Hierbij kan gedacht worden aan omschrijvingen, opmerkingen, beschrijvingen, titels, et cetera die in een normale VARCHAR2 kolom zijn opgeslagen. Zo kan bijvoorbeeld gezocht worden naar het voorkomen van bepaalde zoektermen in de tekstvelden van een aangifte bij een reisverzekering.

Gebruik van Oracle Text

Om de full text search functionaliteit die Oracle Text biedt te kunnen gebruiken moeten speciale indexen aangemaakt worden. Vervolgens kan gewoon van SQL gebruik worden gemaakt om full text search query's uit te voeren. Hierbij worden speciale Oracle Text operatoren, zoals CONTAINS en SCORE, toegepast.

Bij het ontwikkelen van een applicatie die gebruik maakt van Oracle Text kunnen de volgende stappen of onderdelen onderscheiden worden:

1. Aanmaken van een *datastore* waarin de teksten of documenten worden opgeslagen.
2. Aanmaken van een Oracle Text index op de datastore.
3. Laden van teksten of documenten in de datastore.
4. Uitvoeren van zoekvragen.
5. Presenteren van de zoekresultaten.

Deze vijf onderdelen worden in de rest van dit artikel beschreven en aan de hand van een voorbeeld toegelicht.

Aanmaken van een datastore

Voor het opslaan van de teksten die doorzocht moeten kunnen worden zijn drie locaties beschikbaar (zie tabel 1). Indien de tekst wordt opgeslagen in een kolom kan gebruik worden gemaakt van de datatypen, zoals vermeld in tabel 2.

Opslag	Omschrijving
Database	De tekst is opgeslagen in een kolom van een tabel.
File system	De tekst is opgeslagen in een file in het file system. Hierbij bevat een kolom van een tabel de naam van de file.
Web	De tekst is opgeslagen in een file op het Web. Hierbij bevat een kolom van een tabel de URL van de file.

Tabel 1.

CHAR	Geschikt voor korte teksten met een vaste (beperkte) lengte.
VARCHAR2	Geschikt voor korte teksten met een variabele (beperkte) lengte.
CLOB	Geschikt voor lange teksten en ASCII documenten (bijvoorbeeld HTML).
XMLTYPE	Geschikt voor XML documenten.
BLOB	Geschikt voor documenten in verschillende formaten. Wordt binnen een table space opgeslagen en volgt het transactiemodel van de database.
BFILE	Geschikt voor documenten in verschillende formaten. Wordt buiten een table space opgeslagen en volgt niet het transactiemodel van de database.

Tabel 2.

Het voorbeeld dat gebruikt wordt in dit artikel om het gebruik van Oracle Text toe te lichten betreft een applicatie voor het doorzoeken van een verzameling documenten in verschillende formaten. Voor dit voorbeeld worden de documenten in een file system opgeslagen. Dit betekent dat de datastore een kolom moet hebben waarin de naam van de file wordt aangegeven. Hieronder staan de DDL statements waarmee de datastore (tabel DOCUMENTS) wordt aangemaakt. Deze tabel heeft twee foreign keys naar de referentietabellen DOCUMENT_TYPES en FILE_TYPES, respectievelijk een tabel met documenttypen (Oracle Business White Paper, Oracle Technical White Paper, et cetera) en een tabel met fileformaten (Microsoft Word, Microsoft PowerPoint, PDF, HTML, et cetera).

```
CREATE
TABLE   document_types
(       id              NUMBER          NOT NULL
,       description     VARCHAR2(40)    NOT NULL
,       CONSTRAINT document_types_pk PRIMARY KEY (id)
,       CONSTRAINT document_types_uk1 UNIQUE (description)
)
/
CREATE
TABLE   file_types
(       extension       VARCHAR2(4)     NOT NULL
,       description     VARCHAR2(40)    NOT NULL
,       CONSTRAINT file_types_pk PRIMARY KEY (extension)
)
/
CREATE
TABLE   documents
(       id              NUMBER          NOT NULL
,       title           VARCHAR2(100)   NOT NULL
,       author          VARCHAR2(100)   NOT NULL
,       document_type   NUMBER          NOT NULL
,       file_type       VARCHAR2(4)     NOT NULL
,       file_name       VARCHAR2(100)   NOT NULL
,       document_date   DATE                NULL
,       CONSTRAINT documents_pk PRIMARY KEY (id)
,       CONSTRAINT documents_uk1 UNIQUE (file_name)
,       CONSTRAINT documents_document_types_fk FOREIGN KEY
(document_type)
           REFERENCES document_types (id)
,       CONSTRAINT documents_file_types_fk FOREIGN KEY (file_type)
           REFERENCES file_types (extension)
)
/
```

Aanmaken van index

Het aanmaken van een Oracle Text index gebeurt, net als bij een normale index, met het CREATE INDEX statement. De Oracle Text index wordt aangemaakt op de kolom die de tekst of het document zelf bevat of de kolom die de filenaam of URL van het document bevat.

```
CREATE INDEX <index name> ON <table name> (<column name>)
INDEXTYPE IS <index type>
[PARAMETERS ('<parameters string>')]
/
```

In het CREATE INDEX statement wordt aangegeven wat het indextype is. Naast het standaard type CTXSYS.CONTEXT kan gebruik worden gemaakt van een aantal andere indextypen (zie tabel 3).

Indextype	**Omschrijving**
CTXSYS.CTXCAT	Voor het indexeren van korte tekstkolommen die doorzocht worden in combinatie met andere kolommen.
CTXSYS.CTXRULE	Voor het rubriceren van teksten.
CTXSYS.CTXPATH	Voor het versnellen van query's op XMLTYPE waarden die gebruik maken van de ExistsNode() functie.

Tabel 3.

Van CTXSYS.CTXCAT zal in de regel gebruik worden gemaakt voer het zoeken binnen grote tekstvelden in administratieve toepassingen. CTXSYS.CTXRULE en de bijhorende Oracle Text database package CTX_CLS biedt zeer interessante functionaliteit voor het automatisch rubriceren van documenten. Bijvoorbeeld het indelen van nieuwsberichten in categorieën zoals sport, financiën en overheid. De rubricering vindt plaats aan de hand van een aantal regels. Oracle Text kan deze regels

zelfs afleiden uit een gerubriceerde voorbeeld verzameling, een zogenaamde training set. Het laatste indextype in bovenstaand lijstje wordt alleen toegepast bij XML documenten opgeslagen in een XMLTYPE kolom. Deze drie indextypen worden verder niet behandeld in dit artikel.
De eenvoud van het CREATE INDEX statement is enigszins misleidend. De complexiteit zit namelijk verstopt zit in de parameter string. Met de PARAMETERS clause wordt de configuratie van de Oracle Text index aangegeven. De volgende aspecten, of preference classes in Oracle Text terminologie, kunnen worden aangegeven zoals in tabel 4.

Alle preferences moeten worden aangemaakt met behulp van de CTX_DDL database package. Deze package bevat ook een aantal standaard preferences waarvan gebruik kan worden gemaakt. Om van CTX_DDL gebruik te kunnen maken moet aan de betreffende gebruiker (schema) de role CTXAPP toegekend zijn. Voor het voorbeeld moet een datastore preference worden aangemaakt waarmee wordt aangegeven waar de documenten zijn opgeslagen:

```
BEGIN
   CTX_DDL.CREATE_PREFERENCE('demo_datastore', 'FILE_DATASTORE');
   CTX_DDL.SET_ATTRIBUTE('demo_datastore'
   ,                     'path'
   ,
'E:\Cumquat\Projects\Optimize\Oracle_Text\Demo\documents\'
   );
END;
/
```

De preference zelf wordt aangemaakt met CTX_DDL.CREATE_PREFERENCE. De eerste parameter geeft de naam van de preference aan, de tweede parameter de naam van een object in een bepaalde preference class, in dit geval de datastore class. Omdat in het voorbeeld de documenten zijn opgeslagen in het file system wordt FILE_DATASTORE gebruikt. Voor opslag van teksten of documenten in de database zelf wordt DIRECT_DATASTORE gebruikt en voor opslag op het Web URL_DATASTORE. Ieder object in een preference class heeft een aantal attributen. Aan deze attributen wordt met CTX_DDL.SET_ATTRIBUTE een waarde gegeven. In het voorbeeld wordt met een attribuut aangegeven wat het pad is naar de locatie waar de documenten zijn opgeslagen.
Naast de drie genoemde mogelijkheden voor het opslaan van teksten of documenten (in een enkele kolom) in de database, in het file system of op het Web ondersteunt Oracle Text nog vier andere datastores. Namelijk DETAIL_DATASTORE, MULTI_COLUMN_DATASTORE en NESTED_DATASTORE voor opslag in meerdere (geneste of detail) kolommen. En USER_DATASTORE, waarbij een stored procedure moet worden aangegeven die de te indexeren tekst genereert. Dit laatste type kan in sommige situaties goed uitkomst bieden.
Na het aanmaken van de preference kan de index voor het voorbeeld daadwerkelijk aangemaakt worden:

```
CREATE
INDEX       demo_index
ON          documents (file_name)
INDEXTYPE
IS          CTXSYS.CONTEXT
PARAMETERS ('datastore demo_datastore')
/
```

Laden van teksten of documenten

Vervolgens zullen de teksten of documenten geladen moeten worden. In het geval van een file system of URL datastore is het laden heel eenvoudig. De naam of URL van de file moet simpelweg als kolomwaarde worden meegegeven in een insert statement. Uiteraard moet de file op dat moment (of om precies te zijn, zoals later in dit artikel wordt beschreven, op het moment van indexeren) op de aangegeven locatie benaderbaar zijn. Bij een database datastore kan het laden van teksten minder eenvoudig zijn. Dit is afhankelijk van het datatype van de kolom waarin de teksten worden bewaard. In het geval van VARCHAR2 zal de tekst ook weer met een insert statement kunnen worden geladen. Betreft het datatype CLOB, BLOB of BFILE dan zal van PL/SQL (of een andere programmeertaal zoals Java of C) gebruik moeten worden gemaakt. Ook is het mogelijk om met behulp van SQL*Loader documenten te laden. In het voorbeeld kan een document dus geladen worden met een insert statement:

```
INSERT INTO documents VALUES
( 1
, 'Oracle Text - Business White Paper'
, 'Omar Alonso'
, 1
, 'PDF'
, '9ir2text_bwp_f.pdf'
, to_date('01-03-2002', 'DD-MM-YYYY'))
/
```

Indexeren

Bij het laden van een document in een tabel waarop een CTXSYS.CONTEXT is aangemaakt zal de tekst niet direct worden geïndexeerd. Het document dat geïndexeerd moet worden kan namelijk (erg) groot zijn waardoor het indexeren veel tijd kan kosten. Het indexeren vindt daarom niet plaats binnen de transactie. Dit is wel het geval bij een CTXSYS.CTXCAT index. Deze kan namelijk alleen op kortere teksten worden aangemaakt waarbij de indexering snel genoeg kan plaatsvinden.

Preference class	Omschrijving
Datastore	Geeft aan hoe (en waar) de teksten of documenten zijn opgeslagen.
Filter	Geeft aan hoe de documenten naar platte tekst kunnen worden geconverteerd, wat moet gebeuren voordat Oracle Text een document kan indexeren. Standaard wordt hiervoor het INSO filter geboden dat (vrijwel) alle bekende fileformaten ondersteunt.
Lexer	Geeft aan in welke taal of talen de teksten of documenten zijn opgesteld en hoe moet worden omgegaan met bepaalde taalaspecten (zoals verbindingstekens).
Wordlist	Geeft aan of zogenaamde *stem* en *fuzzy* query's moeten kunnen worden uitgevoerd.
Storage	Geeft de storage parameters aan van de index tabellen.
Stoplist	Geeft aan welke woorden of thema's niet geïndexeerd hoeven te worden.
Section Group	Geeft aan of query's binnen secties in de teksten of documenten moeten kunnen worden uitgevoerd en hoe deze secties zijn gedefinieerd. Standaard secties zijn paragrafen en zinnen. Section groups worden onder andere toegepast bij het indexeren van XML documenten, zodat gezocht kan worden binnen bepaalde tags.

Tabel 4.

Voor CTXSYS.CONTEXT indexen wordt bijgehouden voor welke records de index moet worden bijgewerkt (inzichtelijk voor de gebruiker via de view CTX_USER_PENDING). CTX_DDL.SYNC_INDEX en het ALTER INDEX statement maken van deze informatie gebruik om de index te synchroniseren, hetgeen regelmatig zal moeten gebeuren. In de regel gebeurt dit door om de zoveel tijd een job te starten. Bij het aanmaken van een index worden alle records in de datastore direct geïndexeerd. Fouten die tijdens het indexeren optreden kunnen worden opgevraagd via de view CTX_USER_INDEX_ERRORS. Bij het creëren van de Oracle Text index worden een aantal database objecten aangemaakt in het betreffende schema:

```
SQL> select object_name, object_type
  2  from user_objects
  3* order by created desc
SQL> /

OBJECT_NAME                    OBJECT_TYPE
______________ ________

DR$DEMO_INDEX$X                INDEX
DEMO_INDEX                      INDEX
DR$DEMO_INDEX$I                TABLE
DR$DEMO_INDEX$K                TABLE
DR$DEMO_INDEX$R                TABLE
SYS_IOT_TOP_31702              INDEX
SYS_LOB0000031699C00002$$      LOB
SYS_LOB0000031694C00006$$      LOB
SYS_IOT_TOP_31697               INDEX
DR$DEMO_INDEX$N                TABLE
...
```

Van deze objecten is eigenlijk alleen DR$DEMO_INDEX$I interessant. Deze tabel bevat namelijk informatie met betrekking tot de geïndexeerde woorden en thema's. Met de onderstaande query kan achterhaald worden welke woorden hoe vaak voorkomen in de geïndexeerde verzameling documenten. Na het laden en indexeren van negen documenten geeft dit bijvoorbeeld het volgende resultaat:

```
SQL> select token_text
  2  , token_count
  3  from dr$demo_index$i
  4  where token_type = 0
  5* order by token_count desc
SQL> /

TOKEN_TEXT                     TOKEN_COUNT
_____________                  _____________

1                              9
2                              9
4                              9
3                              9
A                              9
APPLICATION                    9
AS                             9
EXTERNAL                       9
INTO                           9
SERVER                         9
S                              9

TOKEN_TEXT                     TOKEN_COUNT
_____________                  _____________

STORED                         9
OTHER                          9
ORACLE9I                        9
ORACLE                         9
OR                             9
ON                             9
JAVA                           9
XML                            9
WITH                           9
TO                             9
INTERNET                       9
```

```
TOKEN_TEXT                     TOKEN_COUNT
_____________                  _____________
E                              9
APPLICATIONS                   9
AND                            9
6                              8
BUSINESS                       8
ACCESS                         8
AN                             8
PROVIDE                        8
PRODUCT                        8
PLATFORM                       8
NEW                            8
...
```

Een aantal woorden komt in ieder document voor (Oracle, Java, XML, Internet, et cetera) en het is duidelijk dat een groot aantal woorden eigenlijk niet geïndexeerd hoeven te worden (as, or, on, to, et cetera). Met behulp van zogenaamde *stoplists* kan dit laatste aangegeven worden. Er zijn standaard stoplists voor diverse talen. Door een taalkolom op te nemen in de datastore tabel kan de Oracle Text index dusdanig geconfigureerd worden dat de stoplist (en ook de lexer preferences) taalafhankelijk worden toegepast.

Uitvoeren van zoekvragen

Zodra de datastore succesvol is geïndexeerd kunnen de documenten of teksten doorzocht worden. Voor het uitvoeren van zoekvragen wordt gebruik gemaakt van de CONTAINS operator in SQL:

```
SQL> SELECT title
  2  FROM   documents
  3  WHERE  CONTAINS(file_name, 'SOAP') > 0
  4  /

TITLE
_____________________________

Oracle9i Web Services Overview
Oracle9i Application Server - Web Services Technical White Paper
Building and Assembling Web Services with Oracle9i Jdeveloper
```

Bovenstaande query vindt alle documenten waarin het woord 'SOAP' voorkomt. De eerste parameter van de CONTAINS operator geeft de kolom aan waarop de Oracle Text index is aangemaakt. De query zelf wordt aangegeven in de tweede parameter. Omdat de CONTAINS operator de score of relevantie retourneert is in de query opgenomen dat alleen de records getoond moeten worden waarvoor deze score groter dan 0 is. De score kan ook als onderdeel van het resultaat van de query getoond worden. Hiervoor moet een (optionele) derde parameter worden toegevoegd aan de CONTAINS operator en gebruik worden gemaakt van de SCORE operator:

```
SQL> SELECT title, score(1)
  2  FROM   documents
  3  WHERE  CONTAINS(file_name, 'SOAP', 1) > 0
  4* ORDER BY SCORE(1) DESC
SQL> /

TITLE                                                          SCORE(1)
_____________________________________- _____

Building and Assembling Web Services with Oracle9i JDeveloper       100
Oracle9i Application Server - Web Services Technical White Paper     89
Oracle9i Web Services Overview                                       27
```

Voor het bepalen van de score maakt Oracle Text gebruik van een gepatenteerd algoritme dat onder andere rekening houdt met hoe vaak het woord voorkomt in een document en hoe vaak het woord voorkomt in andere documenten. Op het Oracle Technology Network. OTN, (http://otn.oracle.com) zijn een aantal papers te vinden die ingaan op diverse (academische) aspecten van scoringsalgoritmen.

Binnen Oracle Text query's kan van een groot aantal operatoren gebruik worden gemaakt, waaronder, zoals verwacht kan worden, de logische operatoren AND, OR en NOT. Maar ook meer 'spannende' operatoren zoals NEAR waarmee gezocht kan worden naar documenten waarin bepaalde woorden dicht bij elkaar (namelijk binnen een aan te geven aantal woorden) en wel of niet in de aangegeven volgorde voorkomen. De onderstaande query vindt de documenten waarin de woorden 'SOAP' en 'WSDL' binnen tien woorden bij elkaar voorkomen:

```
SQL> SELECT title, score(1)
  2  FROM   documents
  3  WHERE  CONTAINS(file_name, 'NEAR((SOAP, WSDL), 5, FALSE)', 1) > 0
  4* ORDER BY score(1) DESC
SQL> /

TITLE                                                          SCORE(1)
_____________________________________- _____

Oracle9i Application Server - Web Services Technical White Paper    100
Building and Assembling Web Services with Oracle9i JDeveloper        86
```

Met behulp van andere operatoren kan gezocht worden op woorden die lijken op een bepaald woord, klinken als een bepaald woord, afgeleid kunnen worden van een bepaald woord of gerelateerd zijn aan een bepaald woord volgens een thesaurus. Naast het indexeren van woorden kunnen ook thema's geïndexeerd worden. Deze indexering gebeurt op basis van een zogenaamde *knowledge base*. Deze wordt standaard meegeleverd voor Engels en Frans. Voor andere talen kunnen eigen knowledge bases gemaakt worden. Voor de negen documenten in het voorbeeld zijn de volgende thema's afgeleid:

```
SQL> select token_text
  2  , token_count
  3  from dr$demo_index$i
  4  where token_type = 1
  5* order by token_count desc
SQL> /

TOKEN_TEXT                     TOKEN_COUNT
_____________                  _____________
newness                        8
abstract ideas and concepts    8
Internet                       7
Oracle Corporation             7
ORACLE9I                       7
computer industry              7
computer software industry     7
creation                       7
XML                            7
performance                    7
static relations               7

TOKEN_TEXT                     TOKEN_COUNT
_____________                  _____________
science and technology         7
availabilities                 7
applications                   7
application                    7
addition                       7
height                         7
hard sciences                  7
provision                      7
construction                   6
databases                      6
existence                      6

TOKEN_TEXT                     TOKEN_COUNT
_____________                  _____________
maintenance                    6
information technology         6
users                          6
structure                      6
storage                        6
servers                        6
searches                       6
access                         6
indexes                        6
generating                     6
relation                       6
```

Op basis van deze thema's kunnen zogenaamde *about* query's worden uitgevoerd om documenten te vinden die over een bepaald thema gaan. Onderstaande query vindt alle documenten over het thema 'databases'.

```
SQL> SELECT title
  2  , SCORE(1)
  3  FROM documents
  4  WHERE CONTAINS(file_name, 'about(databases)', 1) > 0
  5* ORDER BY SCORE(1) DESC
SQL> /

TITLE                                                    SCORE(1)
_____________________________________ _____
Oracle XML DB - Key Features - Release 9.2                    29
Oracle XML DB - Technical White Paper - Release 9.2           28
Oracle XML DB - Frequently Asked Questions                    23
Oracle9i Web Services Overview                                18
Oracle Text - Technical White Paper                           18
Buidling and Assembling Web Services with Oracle9i JDeveloper  2
```

Presenteren zoekresultaten

Tenslotte zullen de zoekresultaten gepresenteerd moeten worden. Hoewel dit uiteraard voor het grootste deel in de toepassing zelf zit, biedt Oracle Text nog een aantal interessante features voor dit onderdeel. Zo kan er een algemene samenvatting, een zogenaamd gist, gegenereerd worden van een tekst of samenvattingen met betrekking tot een specifiek thema. Deze zijn opgebouwd uit de meest relevante paragrafen of zinnen in de oorspronkelijke tekst.

Daarnaast kan van ieder ondersteund fileformaat een HTML representatie gegenereerd worden waarin de zoektermen gemarkeerd zijn. Door deze HTML versie van het document te presenteren kan een gebruiker snel zien waar in het document de zoektermen voorkomen. Bij de markering van de zoektermen kan zelfs van hyperlinks gebruik worden gemaakt zodat de gebruiker naar de volgende of vorige zoekterm kan springen.

Aanvullende informatie

Meer informatie met betrekking tot Oracle Text kan natuurlijk op OTN (http://otn.oracle.com) gevonden worden. Daarnaast zijn er twee Oracle Text manuals in de standaard documentatie van Oracle9i: de Oracle Text Reference (A-96518-01) en de Oracle Text Application Developer's Guide (A-96517-01).

Erwin Groenendal is algemeen en technisch directeur van Cumquat Information Technology. Cumquat levert oplossingen op het gebied van Web Services, B2B Integration, Portals en Self-Service Applications en maakt daarbij gebruik van Oracle, XML en Java technologie. Erwin heeft meer dan 10 jaar ervaring met Oracle en is bereikbaar via erwin.groenendal@cumquat.nl.