

# Oracle8i *interMedia* Text

Migration

Release 8.1.5

February 1999

Part No. A67845-01

**ORACLE**

---

Oracle8i ConText to *interMedia* Text Migration, Release 8.1.5

Part No. A67845-01

Copyright © 1996, 1999, Oracle Corporation. All rights reserved.

Primary Author: Colin McGregor

Contributors: Chandu Bhavsar, D. Yitzik Brenman, Steve Buxton, Anny Chan, Chung-Ho Chen, Paul Dixon, Mohammad Faisal, Garret Kaminaga, Jeff Krauss, Bryn Llewellyn, Kavi Mahesh, Yasuhiro Matsuda, Monal Sonecha, and Steve Yang.

**The Programs are not intended for use in any nuclear, aviation, mass transit, medical, or other inherently dangerous applications. It shall be the licensee's responsibility to take all appropriate fail-safe, backup, redundancy and other measures to ensure the safe use of such applications if the Programs are used for such purposes, and Oracle disclaims liability for any damages caused by such use of the Programs.**

The Programs (which include both the software and documentation) contain proprietary information of Oracle Corporation; they are provided under a license agreement containing restrictions on use and disclosure and are also protected by copyright, patent, and other intellectual and industrial property laws. Reverse engineering, disassembly, or decompilation of the Programs is prohibited.

The information contained in this document is subject to change without notice. If you find any problems in the documentation, please report them to us in writing. Oracle Corporation does not warrant that this document is error free. Except as may be expressly permitted in your license agreement for these Programs, no part of these Programs may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of Oracle Corporation.

If the Programs are delivered to the U.S. Government or anyone licensing or using the Programs on behalf of the U.S. Government, the following notice is applicable:

**Restricted Rights Notice** Programs delivered subject to the DOD FAR Supplement are "commercial computer software" and use, duplication, and disclosure of the Programs including documentation, shall be subject to the licensing restrictions set forth in the applicable Oracle license agreement. Otherwise, Programs delivered subject to the Federal Acquisition Regulations are "restricted computer software" and use, duplication, and disclosure of the Programs shall be subject to the restrictions in FAR 52.227-19, Commercial Computer Software - Restricted Rights (June, 1987). Oracle Corporation, 500 Oracle Parkway, Redwood City, CA 94065.

Oracle is a registered trademark, and ConText, Net8, PL/SQL, Oracle7, Oracle8, Oracle8i, Oracle Call Interface, SQL\*Plus, and SQL\*Loader are trademarks or registered trademarks of Oracle Corporation. All other company or product names mentioned are used for identification purposes only and may be trademarks of their respective owners.

---

---

# Contents

<b>Send Us Your Comments .....</b>	<b>ix</b>
<b>Preface .....</b>	<b>xi</b>
<b>1 Overview of New Features and Changes</b>	
<b>Terminology .....</b>	<b>1-2</b>
General .....	1-2
8.1.5 Terminology .....	1-2
<b>New Features and Enhancements .....</b>	<b>1-3</b>
Integration with Oracle .....	1-3
New Default System .....	1-3
Filtering .....	1-3
Extensible Knowledge Base .....	1-4
Hierarchical Query Feedback .....	1-4
User Datastore .....	1-4
Stoplist Enhancements .....	1-5
Section Searching Enhancements .....	1-5
Alternate Spelling .....	1-6
<b>Changed Features .....</b>	<b>1-7</b>
<b>SQL .....</b>	<b>1-8</b>
Changes .....	1-8
New SQL Commands .....	1-8
<b>Operators .....</b>	<b>1-9</b>
Obsolete Operators .....	1-9
Changed Operators .....	1-9

New Operators.....	1-9
<b>Index Preference Objects</b> .....	1-11
Changes to Index Preference Objects.....	1-11
New Index Objects.....	1-11
<b>PL/SQL</b> .....	1-12
PL/SQL Changes.....	1-12
New PL/SQL.....	1-12
<b>Executables</b> .....	1-13
Changes.....	1-13
New Executables.....	1-13
<b>Views</b> .....	1-14

## 2 Migrating Your Application

<b>Migration Path</b> .....	2-2
Oracle8 Users.....	2-2
Oracle7 Users.....	2-2
<b>Migration Plan</b> .....	2-3
Migrating Your Index.....	2-3
Migrating Your Application Code.....	2-4
Migrating Your Data: Upgrading to Oracle8i.....	2-5
<b>Migration Steps</b> .....	2-6
Disk Space Requirements.....	2-6
Hardware Considerations.....	2-6
Steps.....	2-7

## 3 Administration

<b>Servers</b> .....	3-2
Pre-8.1.5.....	3-2
8.1.5.....	3-2
Viewing the Status of Servers.....	3-2
<b>Roles and Users</b> .....	3-3
pre-8.1.5.....	3-3
8.1.5.....	3-3
Granting Roles and Privileges to Users.....	3-3
<b>Document Loading</b> .....	3-4

pre-8.1.5 .....	3-4
8.1 .....	3-4
<b>Queues and Pipes .....</b>	<b>3-5</b>
Pre-8.1.5 .....	3-5
8.1.5 .....	3-5
<b>Enabling One-Step Queries .....</b>	<b>3-6</b>
Pre-8.1.5 .....	3-6
8.1.5 .....	3-6
<b>Administration Tool .....</b>	<b>3-7</b>
Pre-8.1.5 .....	3-7
8.1.5 .....	3-7

## 4 Index Preferences

<b>Preference Objects .....</b>	<b>4-2</b>
Changes to Index Preference Objects .....	4-2
New Indexing Objects .....	4-5
<b>System-Defined Preferences .....</b>	<b>4-6</b>
<b>Using the Index Migration Scripts .....</b>	<b>4-7</b>
User-Defined Preference Transformation .....	4-7
System-Defined Preference Transformations .....	4-8
Index Creation Warnings .....	4-10

## 5 Indexing

<b>About the Text Index .....</b>	<b>5-2</b>
Merged Word and Theme Index (English only) .....	5-2
Columns with Multiple Indexes .....	5-2
Indexing Views .....	5-2
<b>Procedure for Creating Index .....</b>	<b>5-4</b>
<b>Creating Preferences .....</b>	<b>5-6</b>
<b>Creating an Index .....</b>	<b>5-7</b>
Pre-8.1.5 .....	5-7
8.1.5 .....	5-7
<b>Dropping a Preference .....</b>	<b>5-8</b>
Pre-8.1.5 .....	5-8
8.1.5 .....	5-8

Example.....	5-8
<b>Dropping an Index.....</b>	5-9
Pre-8.1.5.....	5-9
8.1.5.....	5-9
<b>Resuming Failed Index.....</b>	5-10
Pre-8.1.5.....	5-10
8.1.5.....	5-10
<b>Rebuilding an Index.....</b>	5-11
Example.....	5-11
<b>Optimizing an Index.....</b>	5-12
Pre-8.1.5.....	5-12
8.1.5.....	5-12
<b>Updating the Index - Background DML.....</b>	5-13
<b>Updating the Index - Batch DML.....</b>	5-14
Pre-8.1.5.....	5-14
8.1.5.....	5-14
<b>Stoplists and Stopwords.....</b>	5-15
Pre-8.1.5.....	5-15
8.1.5.....	5-15
<b>Document Sections.....</b>	5-17
Pre-8.1.5.....	5-17
8.1.5.....	5-17

## 6 Querying

<b>Overview of Text Queries.....</b>	6-2
Text Query Expressions.....	6-2
Text Query Methods.....	6-2
<b>Text Query.....</b>	6-3
Pre-8.1.5 Method.....	6-3
8.1.5 Method.....	6-3
<b>Cursor Query.....</b>	6-5
Pre-8.1.5 Method.....	6-5
8.1.5 Method.....	6-5
<b>Structured Text Query.....</b>	6-7
Pre-8.1.5 Method.....	6-7

8.1.5 Method .....	6-8
<b>Theme Query (English Only)</b> .....	6-9
Pre-8.1.5 Method .....	6-9
8.1.5 Method .....	6-9
<b>Composite Textkey Query</b> .....	6-10
Pre-8.1.5 Method .....	6-10
8.1.5 Method .....	6-10
<b>Max and First/Next Operators</b> .....	6-11
Pre-8.1.5 Method .....	6-11
8.1.5 Method .....	6-11
<b>PL/SQL Operator</b> .....	6-13
Pre-8.1.5 Method .....	6-13
8.1.5 Method .....	6-13
<b>Counting Hits</b> .....	6-14
Pre-8.1.5 Method .....	6-14
8.1.5 Method .....	6-14
<b>Stored Query Expressions</b> .....	6-16
Pre-8.1.5 Method .....	6-16
8.1.5 Method .....	6-17
<b>Query Explain Plan</b> .....	6-19
Obtaining Explain Information .....	6-19

## 7 Document Presentation

<b>Highlighting</b> .....	7-2
Pre-8.1.5 Method .....	7-2
New 8.1.5 Solutions .....	7-3
<b>Obtaining List of Themes, Gists, and Theme Summaries</b> .....	7-5
Pre-8.1.5 Method .....	7-6
New 8.1.5 Solution .....	7-8

## A PL/SQL Changes

<b>CTX_ADM Package</b> .....	A-2
<b>CTX_DML Package</b> .....	A-3
<b>CTX_DDL Package</b> .....	A-4
<b>CTX_INFO Package</b> .....	A-6

<b>CTX_LING Package</b> .....	A-7
<b>CTX_QUERY Package</b> .....	A-8
<b>CTX_SVC Package</b> .....	A-10
<b>CTX_THES Package</b> .....	A-11

## Index



---

---

# Send Us Your Comments

**Oracle8i ConText to *interMedia* Text Migration, Release 8.1.5**

**Part No. A67845-01**

Oracle Corporation welcomes your comments and suggestions on the quality and usefulness of this publication. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most about this manual?

If you find any errors or have any other suggestions for improvement, please indicate the chapter, section, and page number (if available). You can send comments to us in the following ways:

- Electronic mail: [infodev@us.oracle.com](mailto:infodev@us.oracle.com)
- Fax: (650) 506-7228
- Postal service:  
Oracle Corporation  
Attn: Oracle8i Server Documentation  
500 Oracle Parkway  
Redwood Shores, CA 94065  
USA

If you would like a reply, please give your name, address, and telephone number below.

---

---

---

If you have problems with the software, please contact your local Oracle World Wide Support Center.



---

# Preface

This manual is a guide to help you migrate a ConText Cartridge 2.X application to an *interMedia* Text 8.1.5 application.

## Audience

This manual is intended for a *interMedia* Text application developer or a system administrator responsible for maintaining the *interMedia* Text system.

## Prerequisites

This document assumes that you have experience with the Oracle relational database management system, SQL, SQL\*Plus, and PL/SQL. See the documentation provided with your hardware and software for additional information.

If you are unfamiliar with the Oracle RDBMS and related tools, read Chapter 1, “An Introduction to the Oracle Server”, in *Oracle8 Concepts*. The chapter is a comprehensive introduction to the concepts and terminology used throughout Oracle documentation.

## Related Publications

For more information about *interMedia* Text, see:

- *Oracle8i interMedia Text Reference*

For more information about Oracle8i, see:

- *Oracle8i Concepts*
- *Oracle8i Administrator's Guide*
- *Oracle8i Utilities*
- *Oracle8i Tuning*
- *Oracle8i SQL Reference*
- *Oracle8i Reference*
- *Oracle8i Application Developer's Guide - Fundamentals*

For more information about PL/SQL, see:

- *PL/SQL User's Guide and Reference*

## How This Manual Is Organized

See the table of contents.

## Type Conventions

This manual adheres to the following type conventions:

Type	Meaning
UPPERCASE	Uppercase letters indicate Oracle commands, standard database objects and constants, and standard Oracle PL/SQL procedures.
<i>lowercase italics</i>	Italics indicate variable names, names of user objects (tables, views, preferences, policies, etc.), PL/SQL parameter/argument names, and names of example PL/SQL procedures.  Italics also indicate emphasis.
<code>monospace</code>	Monospace type indicate example SQL*Plus commands and example PL/SQL code. Type in the command or code exactly as it appears.

## Customer Support

You can reach Oracle Worldwide Customer Support 24 hours a day.

In the USA: **1.650.506.1500**

In Europe: + **44.344.860.160**

Please be prepared to supply the following information:

- your CSI number (This helps Oracle Corporation track problems for each customer)
- the release numbers of the Oracle Server and associated products
- the operating system name and version number
- details of error numbers and descriptions (Write down the exact errors you encounter)
- a description of the problem
- a description of the changes made to the system

## **Your Comments Are Welcome**

Please use the Reader's Comment Form at the back of this document to convey your comments to us. You can also contact us at:

Documentation Manager  
Oracle8i Server Documentation  
Oracle Corporation  
500 Oracle Parkway  
Redwood Shores, California 94065  
Phone: 1.650.506.7000 FAX: 1.650.506.7200

---

# Overview of New Features and Changes

This chapter provides an overview of the new features, enhancements, and changes to Oracle8i *interMedia* Text, version 8.1.5, the product formerly known as ConText.

The following topics are covered in this chapter:

- Terminology
- New Features and Enhancements
- Changed Features
- SQL
- Operators
- Index Preference Objects
- PL/SQL
- Executables
- Views

## Terminology

The following terms are used throughout this manual:

### General

#### **Oracle8i *interMedia* Text**

The product formerly known as Oracle ConText Cartridge. Oracle8i *interMedia* Text might also be abbreviated as iMT.

#### **ConText Pre-8.1.5 or ConText 2.X**

Oracle Context Cartridge, version 2.X.

#### **migration**

This term refers to moving from a ConText pre-8.1.5 application to an *interMedia* Text 8.1.5 application.

Old features are referred as features of ConText pre-8.1.5 or ConText 2.X. New features are referred to as features of *interMedia* Text 8.1.5.

**See Also:** For more information on the migration process, see [Chapter 2, "Migrating Your Application"](#).

## 8.1.5 Terminology

#### **Text index**

Domain index of type *context*.

#### **Text query**

A SELECT statement that uses the CONTAINS operator to query a Text index.

#### **word query**

A Text query that matches an exact word or phrase.

#### **ABOUT query**

A Text query that uses the ABOUT operator to search for concepts. This type of query was formerly referred to as a theme query.



## New Features and Enhancements

This section briefly describes the new features of *interMedia* Text 8.1.

### Integration with Oracle

The *interMedia* Text 8.1.5 index is created as an extensible (domain) index to Oracle. As a result, you create the Text index and issue Text queries using standard SQL.

Other benefits of integration include:

- no policies, since the *interMedia* Text domain index is identified by name
- a single index that contains word information and possibly theme information in English
- ability to rename the index
- better query performance
- ability to use the extensible query optimizer

**See Also:** For more information about renaming indexes and using the extensible optimizer, see *Oracle8i interMedia Text Reference*.

### New Default System

A new out-of-box default system enables you to create a Text index immediately without explicitly creating and setting your own custom preferences for indexing.

For example at install time, *interMedia* Text establishes default indexing options, automatically setting language-specific preferences such as lexer and stoplist to the language you specify in your database setup. Oracle also detects your text column format and automatically filters the formatted text contained in these columns.

**See Also:** For more information about the default system, see *Oracle8i interMedia Text Reference*.

### Filtering

Oracle8i *interMedia* text uses the Inso Corporation's filtering technology, which enables the system to automatically detect and filter most document formats. With this single filtering technology, the system can index single or mixed format columns.

You can also set up the system to use your own user filter.

The INSO filtering technology is also used to create plain-text and HTML output for document presentation.

**See Also:** For more information about supported document formats, filtering, and document presentation, see *Oracle8i interMedia Text Reference*.

## Extensible Knowledge Base

You can add custom concepts, categories, words or phrases to the knowledge base to improve theme capabilities. You do this by creating the extension as a thesaurus. You import the thesaurus with *ctxload* and then compile the thesaurus with *ctxkbtc* to augment the existing knowledge base.

**See Also:** For information about the thesaurus loader, *ctxload* and the knowledge base compiler, *ctxkbtc*, see *Oracle8i interMedia Text Reference*

## Hierarchical Query Feedback

Given a query expression, you can obtain related query term information (broader term, narrower term, related term). Your application can present this information to users to help them refine their queries.

**See Also:** For more information about the CTX\_DOC.HFEEDBACK procedure, see *Oracle8i interMedia Text Reference*

## User Datastore

An additional datastore method, *USER\_DATASTORE*, has been added. This data storage method enables you to define a procedure that synthesizes documents during indexing. Such virtual documents exist only during indexing, but content and structure are preserved in the index.

For example, a user-defined procedure might synthesize the date, author, and text columns into one document to have author and date information be indexed as part of a single document.

**See Also:** For more information about *USER\_DATASTORE*, see *Oracle8i interMedia Text Reference*

## Stoplist Enhancements

### Support for Stopthemes and Stopclasses in Stoplists

In addition to defining stopwords, you can define stopthemes and stopclasses, which can be added to a stoplist.

Stopthemes are themes that are not to be indexed. Stopclasses define classes of alphanumeric characters that are not to be indexed, such as numbers.

**See Also:** For more information about the adding stopthemes and stopclasses, see the CTX\_DDL package in the *Oracle8i interMedia Text Reference*

### Dynamic Addition of Objects To Stoplists

You can add stopwords, stopthemes, and stopclasses to a stoplist *after* the index has been created.

**See Also:** For more information, see the ALTER INDEX command in *Oracle8i interMedia Text Reference*

## Section Searching Enhancements

### Field Sections

Field sections are new for 8.1.5. These are sections that are indexed as sub-documents. They have some performance advantages over zone sections.

**See Also:** For more information about document field sections, see the CTX\_DDL.ADD\_FIELD\_SECTION procedure in *Oracle8i interMedia Text Reference*

### XML Section Group

The new section group object XML\_SECTION\_GROUP supports for defining sections in XML-style tagged documents.

### **News Group Section**

The new section group object NEWS\_SECTION\_GROUP supports defining sections in news group formatted documents according to the RFC 1036 specification.

**See Also:** For more information about defining sections in XML and news group formatted documents, see *Oracle8i interMedia Text Reference*

### **Alternate Spelling**

In German, Danish, and Swedish, *interMedia Text* recognizes the accepted alternate spellings of query terms. You enable alternate spelling with BASIC\_LEXER.

**See Also:** For information about alternate spelling, see *Oracle8i interMedia Text Reference*

## Changed Features

Because the Text index is now an Oracle domain index, most features such as indexing, querying, and document presentation have changed in iMT 8.1.5. How to migrate these features are discussed throughout this manual.

The following table lists the topics that are covered and where in this manual to look for more information:

<b>Topic</b>	<b>Where to Find More Migration Information</b>
Planning Your Migration	<a href="#">Chapter 2, "Migrating Your Application"</a>
Administration	<a href="#">Chapter 3, "Administration"</a>
Index Tables and Objects	<a href="#">Chapter 4, "Index Preferences"</a>
Index Creation and Management	<a href="#">Chapter 5, "Indexing"</a>
Querying	<a href="#">Chapter 6, "Querying"</a>
Document Presentation	<a href="#">Chapter 7, "Document Presentation"</a>

## SQL

### Changes

The CONTAINS and SCORE operators remain unchanged for 8.1.

**See Also:** For more information about migrating Text queries, see [Chapter 6, "Querying"](#).

For the syntax of these operators, see *Oracle8i interMedia Text Reference*.

### New SQL Commands

You use following standard SQL commands to create and manage the 8.1.5 Text index, which is a domain index of type *context*:

- Use CREATE INDEX to create a Text index.
- Use ALTER INDEX for managing Text indexes.
- Use DROP INDEX to drop a Text index.

**See Also:** For more information about migrating Text indexes with CREATE INDEX and ALTER INDEX, see [Chapter 5, "Indexing"](#).

For syntax information, refer to the *Oracle8i interMedia Text Reference*.

## Operators

Most of the operators available in pre-8.1.5 are available in 8.1.5. However, some are obsolete and have changed.

### Obsolete Operators

The following table list the operators that are obsolete in 8.1.5. The column entitled "How to Migrate" tells you how to migrate and where to look for more information:

Operator	Equivalent	Status	How to Migrate
EXECUTE	@	<i>Obsolete</i>	Call the function in SELECT statement. See " <a href="#">PL/SQL Operator</a> " in <a href="#">Chapter 6</a> .
First/Next	#	<i>Obsolete</i>	Replace result-set selectivity with cursor query. See " <a href="#">Max and First/Next Operators</a> " in <a href="#">Chapter 6</a> .
Max	:	<i>Obsolete</i>	Replace result-set selectivity with cursor query. See " <a href="#">Max and First/Next Operators</a> " in <a href="#">Chapter 6</a>

### Changed Operators

The following operator has changed in 8.1.5

Operator	Equivalent	Status	Migration Notes
ACCUMulate	,	Changed	Scoring method different. See <i>Oracle8i interMedia Text Reference</i>

### New Operators

Oracle 8i *interMedia* Text 8.1.5 provides the following new operators:

- ABOUT
- TR
- TRSYN

**See Also:** For more information on the ABOUT operator, see "[Theme Query \(English Only\)](#)" in [Chapter 6](#).

For complete syntax information about these and existing operators, see *Oracle8i interMedia Text Reference*



## Index Preference Objects

### Changes to Index Preference Objects

Most index preference objects and system-defined preferences have changed in 8.1.5.

**See Also:** For more information on index object changes, see [Chapter 4, "Index Preferences"](#).

### New Index Objects

The following index objects have been added:

#### Data Storage

- USER\_DATASTORE

#### Section Group

- NEWS\_SECTION\_GROUP
- XML\_SECTION\_GROUP

**See Also:** For more information about new and existing objects, see the *Oracle8i interMedia Text Reference*.

## PL/SQL

### PL/SQL Changes

Most PL/SQL packages and procedures have changed.

**See Also:** For a list of PL/SQL changes, see [Appendix A, "PL/SQL Changes"](#).

### New PL/SQL

Oracle8i *interMedia Text* 8.1.5 has new packages and procedures.

**See Also:** For a complete list of the *interMedia Text* 8.1.5 packages and procedures, refer to the *Oracle8i interMedia Text Reference*.

## Executables

### Changes

The following executables have changed for 8.1.5:

#### **ctxsrv Server**

This executable has undergone the following changes:

- *ctxsrv* functions only as M personality, which runs in the background to process DML.
- Query (Q) and Reader (R) Linguistics (L) and DDL (D) personalities are obsolete.

#### **ctxload Loader**

This loader retains all its pre-8.1.5 functionality. It has added support of export/updating BLOB and CLOB columns.

**See Also:** To learn more about the *ctxsrv* and *ctxload* command syntax, see the *Oracle8i interMedia Text Reference*.

#### **ctxctl Utility**

This shell script that monitors and shuts down *ctxsrv* is obsolete.

**See Also:** For more information about how to monitor servers in 8.1.5, see "[Servers](#)" in [Chapter 3](#).

### New Executables

The following executable is new for 8.1:

#### **ctxkbtc**

This executable compiles an extended knowledge base from one or more thesauri (up to 16).

**See Also:** To learn more about the *ctxkbtc* command syntax, see the *Oracle8i interMedia Text Reference*.

## Views

Most views from ConText 2.X have been renamed. New views have also been added to the 8.1.5 release.

**See Also:** For a complete list all the views in the 8.1.5 release, see the *Oracle8i interMedia Text Reference*.

---

# Migrating Your Application

This chapter describes the *interMedia* Text migration process. The following topics are covered:

- [Migration Path](#)
- [Migration Plan](#)
- [Migration Steps](#)

## Migration Path

This book describes migrating a ConText 2.X application running on an Oracle 8.0 database to a *interMedia* Text 8.1.5 application running on an Oracle 8*i* database, version 8.1.5.

---

---

**Note:** You *cannot* run ConText 2.X applications on an Oracle8*i* database.

---

---

### Oracle8 Users

If you are running ConText 2.X on an Oracle8 database, you must migrate to *interMedia* Text 8.1.5, which also involves upgrading your Oracle8 database.

**See Also:** "[Migration Plan](#)" in this chapter.

### Oracle7 Users

If you are running ConText 2.X on an Oracle7 database, you must migrate your database to Oracle8*i* before you can migrate your application to *interMedia* Text 8.1.

**Note:** For more information about migrating an Oracle7 database to an Oracle8*i* database, refer to the *Oracle8i Migration*.

## Migration Plan

Migration assumes you have ConText 2.X installed and are running a ConText application. The goal of migration is to move your ConText 2.X/Oracle 8.0 production application to a working version of the same application in a *interMedia Text 8.1.5/Oracle8i* environment.

Migration of a ConText application is not a mechanical process as is a *migration* from an Oracle7 database to an Oracle8 database. This is because ConText 2.X applications cannot be run in a *interMedia Text 8.1.5* environment.

A migration to *interMedia Text 8.1.5* involves the general plan of:

1. [Migrating Your Index](#)
2. [Migrating Your Application Code](#)
3. [Migrating Your Data: Upgrading to Oracle8i](#)

## Migrating Your Index

The migration of a pre-8.1.5 index involves duplicating the indexing features of the old environment in the new 8.1.5 environment.

The way you characterize an index in pre-8.1.5 is similar to 8.1.5 in that you must create preferences. However, attaching preferences to an index is different in that policies do not exist in 8.1.5. In 8.1.5, you name your custom preferences directly in the CREATE INDEX statement.

---

---

**Note:** Creating your own preferences in 8.1.5 is necessary only when you require non-default index preferences.

For more information about the characteristics of a default index, see the first chapter in the *Oracle8i interMedia Text Reference*

---

---

Most system-supplied objects you use to create index preferences have been renamed, while others have been made obsolete. Migrating your pre-8.1.5 index thus involves knowing which system-supplied objects have changed or have been made obsolete.

You can migrate your index preferences and index by using one or both of the following methods:

- [Manual Index Migration](#)

- [Index Migration Scripts](#)

### Manual Index Migration

Manual index migration involves re-creating your pre-8.1.5 preferences with 8.1.5 preferences by rewriting your index creation scripts. You create new preferences with the renamed or new index objects in 8.1.5, and decide how to migrate preferences based on obsolete objects.

**See Also:** For more information about replaced, obsolete, and changed index objects, see ["Preference Objects"](#) in [Chapter 4](#).

For more information about how to create preferences and indexes with the new syntax, see [Chapter 5, "Indexing"](#).

Manual index migration might be useful if you decide to re-design your application using new features in 8.1.5.

### Index Migration Scripts

To help with migrating a pre-8.1.5 index to 8.1 index, you can use the migration scripts supplied with the 8.1.5 installation. You run these scripts in the pre-8.1.5 environment to create *migrate.sql*, and then you run *migrate.sql* in your new 8.1.5 environment.

The *migrate.sql* script, written in *interMedia Text 8.1.5* code, attempts to replicate your pre-8.1.5 index preferences and index in your *interMedia Text 8.1.5* environment. Where there is a 2.X indexing feature or object that is obsolete in 8.1, the migration script issues a warning as a comment. You must make a decision on how to migrate the feature and edit the migration script accordingly.

**See Also:** For more information about when you use the migration scripts, see ["Migration Steps"](#) in this chapter.

For more information about using the migration scripts, see ["Using the Index Migration Scripts"](#) in [Chapter 4](#).

## Migrating Your Application Code

To migrate a ConText pre-8.1.5 application to an iMT application, you must completely re-rewrite your application using 8.1.5 code. This is because pre-8.1.5 applications are not forward compatible.



In addition to the code that creates indexes, application code requiring migration includes all code that issues queries and presents documents to users.

The new code is a result of text searching and indexing integration with the Oracle server. For example, the Text index is now a domain index, and as such, you index and query with standard SQL. In addition, there are new PL/SQL procedures that create output for document presentation.

Recreating your application code involves creating a test environment as part of your migration process.

**See Also:** For more information about creating a test environment, see ["Migration Steps"](#) in this chapter.

For more information about migrating queries, see [Chapter 6, "Querying"](#).

For more information about new solutions for document presentation, see [Chapter 7, "Document Presentation"](#).

## Migrating Your Data: Upgrading to Oracle8i

Migration of data is the transfer of your pre-8.1.5 data to your 8.1.5 database. This is also known as upgrading an Oracle8 database to an Oracle8i database. You do this after installing *interMedia Text*.

You can migrate your data manually using export/import, or you can use the Oracle Data Migration Assistant, which is started automatically after installing *interMedia Text* with the Oracle Universal Installer.

**See Also:** For more information about upgrading to Oracle8i, see *Oracle8i Migration*.

For more information about when you migrate your data, see ["Migration Steps"](#) in this chapter.

### Long Columns

If your pre-8.1.5 application uses LONG and LONG RAW text columns, you might also consider migrating these columns to LOBs as part of your data migration process.

**See Also:** For more information about migrating LONG and LONG RAW columns, see *Oracle8i Migration*.

## Migration Steps

This section describes the requirements for migration as well as the steps you must take to migrate your application.

### Disk Space Requirements

The migration step assumes that you have enough disk space to create a separate test environment, in addition to your production environment, that contains the following elements:

- all or a subset of your production data
- new application code

### Hardware Considerations

A migration to Oracle8i *interMedia* Text involves using three Oracle homes:

- pre-8.1.5 production Oracle home
- test 8.1.5 Oracle home
- new production 8.1.5 Oracle home

You might consider creating your test oracle home on a separate machine from your pre-8.1.5 production oracle home. Do this so that your test environment does not compete for resources such as disk space and processing time with your production environment while you recreate your application code.

You must create a new 8.1.5 oracle home to upgrade your database. The upgrade process for Oracle8 to Oracle8i expects this to be on the same machine, but this is not necessary.

## Steps

**Table 2–1** outlines a possible plan for migrating your pre-8.1.5 application running on Oracle8 to a working 8.1.5 application running on Oracle8i. This plan suggests setting up a test environment next to your production environment, outlining the steps you execute in each environment.

---



---

**Note:** The following table is a recommended plan. The exact details of a migration plan will differ from site to site depending on how your application is designed.

---



---

**Table 2–1**

Step	Production Environment	Test Environment
1.	Running	Install <i>interMedia</i> Text 8.1.5 and Oracle8i software in an Oracle home separate from your production Oracle home. This test Oracle home is your test environment.
2.	Copy <i>drminst</i> and <i>drmrn</i> from test environment. See <a href="#">"Using the Index Migration Scripts"</a> in Chapter 4.	
3.	Run <i>drminst</i> and <i>drmrn</i> scripts to generate <i>migrate.sql</i> . See <a href="#">"Using the Index Migration Scripts"</a> in Chapter 4.	
4.	Export all or subset of production data to be used in the test environment.	
5.	Running	Start Oracle8i database server.
6.	Running	Import data or subset of production data.
7.	Running	Copy <i>migrate.sql</i> from production environment.
8.	Running	Edit <i>migrate.sql</i> . See <a href="#">"Using the Index Migration Scripts"</a> in Chapter 4.

**Table 2–1**

Step	Production Environment	Test Environment
9.	Running	<p>Migrate your index by running either <i>migrate.sql</i> or your own index creation scripts.</p> <p>See <a href="#">"Migrating Your Index"</a> in this chapter.</p> <p>You now have an <i>interMedia Text 8.1.5</i> environment with schema and indexes similar to your pre-8.1.5 production environment.</p>
10.	Running	<p>Migrate pre-8.1.5 application code.</p> <p>See <a href="#">"Migrating Your Application Code"</a> in this chapter.</p>
11.	Running	<p>Test application in <i>interMedia Text 8.1.5</i> environment.</p>
12.	Backup your pre-8.1.5 system.	
13.	Drop your indexes and CTXSYS schema.	
14.	<p>Install <i>interMedia Text 8.1.5</i> using the Oracle Universal Installer.</p> <p>In this process, you must create a new Oracle home for your Oracle8i <i>interMedia Text</i> installation. This new Oracle home will soon become your new production Oracle home. This new home is different from the old production Oracle home (which runs on Oracle8) and the test Oracle home.</p> <p>This installation process also creates a new CTXSYS user and the 8.1.5 <i>interMedia Text</i> data dictionary.</p>	
15.	<p>Migrate your data (same as upgrading to Oracle8i) using the Oracle Data Migration Assistant.</p> <p>The Oracle Universal Installer starts this assistant automatically after you install <i>interMedia Text</i>.</p> <p>See <a href="#">"Migrating Your Data: Upgrading to Oracle8i"</a> in this chapter.</p>	

**Table 2–1**

---

<b>Step</b>	<b>Production Environment</b>	<b>Test Environment</b>
16.	Copy <i>migrate.sql</i> and other newly created index creation scripts from your test Oracle home.	
17.	Run <i>migrate.sql</i> and/or other index creating scripts to recreate index.	
18.	Copy new 8.1.5 application code from test oracle home.	
19.	Run 8.1.5 application.	

---



---

# Administration

This chapter describes the administration concepts that have changed from 2.X to 8.1.5. The following topics are covered:

- [Servers](#)
- [Roles and Users](#)
- [Document Loading](#)
- [Queues and Pipes](#)
- [Enabling One-Step Queries](#)
- [Administration Tool](#)

## Servers

### Pre-8.1.5

In pre-8.1.5, the ConText server, *ctxsrv*, has five personalities, R Q D M and L. A single server can have one or more personalities. A combination of personalities is called a mask. With personality masks, you can assign to a server different functions, such as query processing, DDL processing, or linguistic processing.

### 8.1.5

In 8.1.5, the *ctxsrv* executable is not required for indexing or querying. Indexing and querying are performed with standard SQL. You run the server only for background DML processing. No server is needed for performing batch DML, which you do by synchronizing the index with ALTER INDEX.

The *ctxsrv* executable is also not required for producing document services output, such as theme summaries and Gists. You obtain document services output synchronously with procedures the CTX\_DOC package.

When you start *ctxsrv*, the only personality you can specify is M, which is the default.

**See Also:** For more information about *ctxsrv*, ALTER INDEX and the CTX\_DOC package, see the *Oracle8i interMedia Text Reference*.

## Viewing the Status of Servers

*interMedia Text 8.1.5* does not support the *ctxctl* command that is used in pre-8.1.5 to view the status of ConText servers.

In 8.1.5 to view the status of a server, you can use the CTX\_SERVERS view. You can also use the Oracle8i *interMedia Text Manager*, which is a Java application available with the Oracle Enterprise Manager.

---

---

**Note:** The Oracle8i Enterprise Manager is shipped on a separate CD from Oracle8i *interMedia Text*.

---

---



## Roles and Users

### pre-8.1.5

In pre-8.1.5, a ConText user can be assigned one of three pre-defined ConText roles

- CTXADMIN
- CTXAPP
- CTXUSER

### 8.1.5

In 8.1.5, the CTXADMIN and CTXUSER roles are obsolete. A Text user can only be assigned the CTXSYS or CTXAPP roles.

#### **CTXSYS Role**

The CTXSYS role enables users to do the following

- start a *ctxsrv* server
- perform all the tasks of a CTXAPP user

#### **CTXAPP Role**

The CTXAPP role enables users to do the following:

- create indexes
- manage the Text data dictionary, including creating and deleting preferences
- issue Text queries
- use the *interMedia* Text PL/SQL packages

## Granting Roles and Privileges to Users

In pre-8.1.5, the system uses the standard SQL model for granting roles to users. This model has not changed for *interMedia* Text 8.1.5. To grant a Text role to a user, you use SQL GRANT command.

In addition, to allow application developers to call procedures in the *interMedia* Text PL/SQL packages, you must explicitly grant to each user EXECUTE privileges for the *interMedia* Text package.

## Document Loading

### pre-8.1.5

You use the *ctxload* executable to load documents into LONG or LONG RAW text columns. You also use it to import thesauri.

### 8.1

#### **ctxload**

In 8.1.5, *ctxload* still supports the loading of LONGs and LONG RAWs as well as importing thesauri.

---

---

**Suggestion:** If your pre-8.1.5 application uses LONG or LONG RAW to store your documents, Oracle recommends that you migrate these columns to LOBs. To do so, you can use the SQL operator TO\_LOB.

For more information about migrating your LONGs and LONG RAWs to LOBs, see *Oracle8i Migration*.

For more information about loading LONGs with *ctxload*, see *Oracle8i interMedia Text Reference*.

---

---

#### **SQL\*Loader**

In 8.1.5, you can store documents in the columns types of CHAR, CLOB, BLOB, BFILE, or VARCHAR2. To load documents to any of these data types, you can use SQL\*Loader.

**See Also:** For an example of using SQL\*Loader, see *Oracle8i interMedia Text Reference*.

## Queues and Pipes

### Pre-8.1.5

In pre-8.1.5, ConText uses the text request queue to process text operations. The text request queue is made up of a query pipe, a DDL pipe, a DML queue, and a services queue.

### 8.1.5

In 8.1.5, text indexing and querying are performed with standard SQL. As such, all queues and pipes except the DML queue are obsolete.

The DML queue stores requests for index updates and is populated by Oracle.

Pending DML requests can be viewed with `CTX_PENDING` and `CTX_USER_PENDING` views.

DML errors can be viewed with the `CTX_INDEX_ERRORS` or `CTX_USER_INDEX_ERRORS` view.

## Enabling One-Step Queries

### Pre-8.1.5

In pre-8.1.5, you enable one-step queries by setting the *text\_enable* variable to TRUE in the *initsid.ora* file.

### 8.1.5

In 8.1.5, you use standard SQL to issue a Text query. After installing *interMedia* Text 8.1.5, no extra setup steps are necessary. You can issue Text queries once you have created a valid Text index.

**See Also:** For more information about creating Text indexes, see [Chapter 5, "Indexing"](#).

## Administration Tool

### Pre-8.1.5

In pre-8.1.5, you can use the GUI Administration Tool to create indexes, start servers etc. This tool is available with the ConText Workbench.

### 8.1.5

The *interMedia* Text 8.1.5, the administration tool has been replaced with the Oracle8*i interMedia* Text Manager. This tool is a Java application integrated with the Oracle Enterprise Manager, which is available on a separate CD.

The Text Manager enables administrators to create preferences, stoplists, sections, and indexes. This tool also enables administrators to start, monitor and shutdown *ctxsrv* DML servers.

**See Also:** for more information about the Oracle8*i interMedia* Text Manager, see the online help shipped with this tool.



---

# Index Preferences

Use this chapter as a guide to replacing ConText pre-8.1.5 objects with *interMedia* Text 8.1.5 objects. The following topics are covered:

- [Preference Objects](#)
- [System-Defined Preferences](#)
- [Using the Index Migration Scripts](#)

## Preference Objects

### Changes to Index Preference Objects

[Table 4-1](#) lists the objects that are used to create preferences for indexing that have changed or are obsolete from 2.x to 8.1.

You can migrate preferences defined with 2.x objects manually using the information in [Table 4-1](#), or you can use the migration scripts.

**See Also:** For more information on using the migration scripts, see "[Using the Index Migration Scripts](#)" in this chapter.

For more information about creating preferences, see [Chapter 5, "Indexing"](#).



**Table 4–1**

Preference Class	Preference Object	Status	Migration Notes
Datastore	DIRECT	Changed	Renamed to DETAIL_DATSTORE
	MASTER DETAIL	<i>Obsolete</i>	In 8.1, MASTER DETAIL is obsolete. You set up master/detail tables using the DETAIL_DATASTORE object (MASTER DETAIL NEW in 2.x).
	MASTER DETAIL NEW	Changed	Renamed to DETAIL_DATASTORE
	OSFILE	Changed	Renamed to FILE_DATASTORE
	URL	Changed	Renamed to URL_DATASTORE
Filter	NULL FILTER	Changed	Renamed to NULL_FILTER
	BLASTER FILTER	<i>Obsolete</i>	In 8.1, BLASTER FILTER is obsolete.  To index single or mixed format columns, use INSO_FILTER object. The INSO filter autodetects, so format attribute is not needed.  To index formats (single or mixed column) not supported by INSO_FILTER, create/supply a user filter and use USER_FILTER object, specifying the executable name with the <i>command</i> attribute.
	HTML FILTER	<i>Obsolete</i>	To index HTML documents without indexing HTML tags, create and specify a section group preference of type HTML_SECTION_GROUP.  If Japanese CODE_CONVERSION is on for HTML documents in 2.x, use the CHARSET_FILTER filter preference with CHARSET set to JAAUTO. KEEP_TAG is not needed anymore as HTML filtering and sectioning have been combined in one object.
	USER FILTER	Changed	Renamed to USER_FILTER
Lexer	BASIC LEXER	Changed	Renamed to BASIC_LEXER  In addition, the SENT_PARA is no longer a valid attribute. To enable sentence or paragraph searching, add a special section to the section group preference.
	JAPANESE V-GRAM LEXER	Changed	Renamed to JAPANESE_VGRAM_LEXER  In addition, in 8.1, this object has no attributes.
	KOREAN LEXER	Changed	Renamed to KOREAN_LEXER  In addition, in 8.1, attributes have been added to allow users to control what is indexed.
	CHINESE V-GRAM LEXER	Changed	Renamed to CHINESE_VGRAM_LEXER  In addition, this object has no attributes.

**Table 4–1**

Preference Class	Preference Object	Status	Migration Notes
	THEME LEXER	<i>Obsolete</i>	To index themes, use BASIC_LEXER with INDEX_THEMES attribute set to TRUE.
	NLS_LEXER	<i>Obsolete</i>	This object was not implemented in 2.x and has been removed for 8.1
Wordlist	GENERIC WORD LIST	Changed	<p>Renamed to BASIC_WORDLIST</p> <p>In addition, the STCLAUSE, INSTCLAUSE, SOUNDEX_AT_INDEX, and SECTION_GROUP attributes are obsolete.</p> <p>Soundex is automatically enabled and does not require any attributes.</p> <p>To specify a section group for an index, first create a section group using CTX_DDL.CREATE_SECTION_GROUP, then specify the section group in the parameter string of CREATE INDEX.</p> <p>BASIC_WORDLIST only requires attributes for the STEMMER and FUZZY attributes.</p>
Stoplevel	GENERIC STOP LIST	<i>Obsolete</i>	Create stoplists using the CTX_DDL.CREATE_STOPLIST procedure and specify the stoplist in the parameter string of CREATE INDEX.
Engine	GENERIC ENGINE	Changed	<p>Renamed to BASIC_STORAGE</p> <p>In 8.1, Engine preference renamed to Storage preference.</p> <p>In addition, all the 2.X attributes are obsolete.</p> <p>No INDEX_MEMORY attribute. Set indexing memory in the parameter string of CREATE INDEX and ALTER INDEX.</p> <p>Various options have been combined. So, instead of specifying three clauses for each table as I1I_TABLESPACE, I1I_STORAGE etc., specify all clauses with one attribute I_TABLE_CLAUSE.</p>
Loader	GENERIC LOADER	<i>Obsolete</i>	<p>These preference objects were used in 2.x to create sources for automating text loading using ctxload.</p> <p>In 8.1, ctxload no longer supports text loading. As a result, these preference objects are obsolete.</p> <p>To load text in database, use SQL*Loader.</p>
	NULL TRANSLATOR	<i>Obsolete</i>	
	USER TRANSLATOR	<i>Obsolete</i>	
	DIRECTORY READER	<i>Obsolete</i>	
Compressor	NULL COMPRESSOR	<i>Obsolete</i>	This object was not implemented in 2.x and has been removed for 8.1.

## New Indexing Objects

Oracle8i *interMedia* Text 8.1.5 has some new indexing objects.

**See Also:** For a complete list of the *interMedia* Text 8.1.5 indexing objects, see *Oracle8i interMedia Text Reference*.

## System-Defined Preferences

Oracle8i *interMedia* Text has renamed most system-defined preferences and added some new ones.

**See Also:** For a complete list of the *interMedia* Text 8.1.5 system-defined preferences, see *Oracle8i interMedia Text Reference*.

The migration script automates the transformation of system-defined preferences.

**See Also:** For a complete list on the automated transformations, see "[System-Defined Preference Transformations](#)" in the next section.

## Using the Index Migration Scripts

The migration scripts are the following:

- `drminst`
- `drmrn`
- `migrate.sql`

The migration scripts `drminst` and `drmrn` are obtained from your 8.1.5 installation. They are located in the `$ORACLE_HOME/ctx/migrate` directory.

You must copy these scripts into your production environment (pre-8.1). Run these scripts to generate `migrate.sql`.

The `migrate.sql` script attempts to recreate the pre-8.1.5 index as closely as possible. It does so by creating corresponding preferences and attributes and finally the index using *interMedia* Text 8.1.5 code.

You then copy `migrate.sql` to your test environment, edit it appropriately, and run the edited `migrate.sql` to transform your index.

**See Also:** For more information on planning your migration, see "[Migration Plan](#)" in [Chapter 2](#).

## User-Defined Preference Transformation

The migration script attempts to replace all preferences defined with pre-8.1.5 objects with preferences defined with the renamed 8.1.5 objects.

For example, if you have a pre-8.1.5 policy that uses a preference named `mydatapref` set to the `OSFILE` object, the migration script creates an 8.1.5 preference called `mydatapref` and sets it to use the `FILE_DATASTORE` object. The 8.1.5 index is created with the `mydatapref` storage preference.

### Warning for Preferences Based on Obsolete Objects

When an pre-8.1.5 object or attribute has been made obsolete in 8.1, the migration script issues a warning in the form of a `REM` statement and creates no corresponding preference. In such cases, you must edit the migration script to reflect the correct replacement preference, if any, for your system.

**See Also:** For a list of renamed and obsolete objects, see "[Preference Objects](#)" in this chapter.

## System-Defined Preference Transformations

The migration script replaces pre-8.1.5 system-defined preferences with 8.1.5 system-defined preferences.

### Unchanged Preferences

When a system-defined preference name has not changed from pre-8.1.5 to 8.1, the migration script uses the same preference name and attribute. For example, if your pre-8.1.5 policy used the `DEFAULT_LEXER` system-defined preference, your 8.1.5 index will be created with the `DEFAULT_LEXER` system-defined lexer preference.

### Renamed Preferences

When system-defined preference names or attributes have been renamed for 8.1, the migration script replaces the old name with the new name.

### Warning for Obsolete Preferences

When an pre-8.1.5 system-defined preference or attribute has been made obsolete in 8.1, the migration script issues a warning in the form of a `REM` statement and creates no corresponding preference. In such cases, you must edit the migration script to reflect the correct replacement preference, if any, for your system.

### Table of Transformations

The migration script replaces system-defined 2.X preferences with the new 8.1.5 preferences. The following table lists these replacements:

<b>2.x System-Defined Preference</b>	<b>8.1.5 Replacement</b>
<code>DEFAULT_DIRECT_DATASTORE</code>	<code>DEFAULT_DATASTORE</code>
<code>MD_TEXT</code>	Obsolete
<code>MD_BINARY</code>	Obsolete
<code>OSFILE</code>	<code>FILE_DATASTORE</code>
<code>DEFAULT_URL</code>	<code>URL_DATASTORE</code>
<code>DEFAULT_NULL_COMPRESSOR</code>	Obsolete
<code>DEFAULT_NULL_FILTER</code>	<code>NULL_FILTER</code>
<code>AUTOB</code>	<code>INSO_FILTER</code>
<code>WW6B</code>	<code>INSO_FILTER</code>
<code>HTML_FILTER</code>	Obsolete

<b>2.x System-Defined Preference</b>	<b>8.1.5 Replacement</b>
BASIC_HTML_FILTER	Obsolete
DEFAULT_LEXER	DEFAULT_LEXER
VGRAM_JAPANESE	JAPANESE_LEXER
VGRAM_JAPANESE_1	JAPANESE_LEXER
VGRAM_JAPANESE_2	JAPANESE_LEXER
KOREAN	KOREAN_LEXER
VGRAM_CHINESE	CHINESE_LEXER
VGRAM_CHINESE_1	CHINESE_LEXER
VGRAM_CHINESE_2	CHINESE_LEXER
THEME_LEXER	Obsolete
BASIC_HTML_LEXER	Obsolete
BASIC_HTML_SECTION	Obsolete
SOUNDEX	DEFAULT_WORDLIST
NO_SOUNDEX	DEFAULT_WORDLIST
VGRAM_JAPANESE_WORDLIST	JAPANESE_WORDLIST
KOREAN_WORDLIST	KOREAN_WORDLIST
VGRAM_CHINESE_WORDLIST	CHINESE_WORDLIST
BASIC_HTML_WORDLIST	Obsolete
DEFAULT_STOPLIST	DEFAULT_STOPLIST
NO_STOPLIST	EMPTY_STOPLIST
DEFAULT_INDEX	DEFAULT_STORAGE
DEFAULT_LOADER	Obsolete
DEFAULT_TRANSLATOR	Obsolete
FRENCH_STOPLIST	FRENCH_STOPLIST
ITALIAN_STOPLIST	ITALIAN_STOPLIST
SPANISH_STOPLIST	SPANISH_STOPLIST
GERMAN_STOPLIST	GERMAN_STOPLIST

## Index Creation Warnings

After attributes and preferences have been created, the migration script attempts to recreate the pre-8.1.5 indexes from the pre-8.1.5 policy information.

Policies do not exist in *interMedia* Text 8.1. In 8.1, you create a Text index as an extensible type of Oracle index. In addition, you can create only a single index per column.

Therefore, the migration script issues a warning as a REM statement if your pre-8.1.5 environment includes any of the following non-migratable structures:

- multiple policies on a single column
- non-indexed policies
- template policies
- indexes on views

**See Also:** For more information on the Text index, see "[About the Text Index](#)" in [Chapter 5](#).



The chapter discusses the changes to the Text indexing process that might affect your applications. The following topics are covered:

- [About the Text Index](#)
- [Procedure for Creating Index](#)
- [Creating Preferences](#)
- [Creating an Index](#)
- [Dropping a Preference](#)
- [Resuming Failed Index](#)
- [Rebuilding an Index](#)
- [Optimizing an Index](#)
- [Updating the Index - Background DML](#)
- [Updating the Index - Batch DML](#)
- [Stoplists and Stopwords](#)
- [Document Sections](#)

## About the Text Index

In pre-8.1.5, the index is created with the CTX\_DDL package by first creating a policy and then using the policy to create the index.

In 8.1.5, a Text index is created as a special type of extensible index to Oracle using standard SQL. This means that a *interMedia* Text 8.1.5 index operates like a Oracle index. It has a name by which it is referenced, and policies do not exist.

**See Also:** For more information about creating a Text index, see "[Procedure for Creating Index](#)" in this chapter.

## Merged Word and Theme Index (English only)

In 8.1.5, a single text index can contain both theme and word information. This is different from pre-8.1.5 where you needed a theme index in addition to a text index to issue theme queries.

By default in English, *interMedia* Text indexes theme information with word information. You can optionally enable and disable theme indexing with your lexer preference.

**See Also:** To learn more about indexing theme information, see "[Creating Preferences](#)" in this chapter.

## Columns with Multiple Indexes

In pre-8.1.5, the system allows you to create more than one index on a text column. This is useful when you want a text column to have a text and theme index.

In 8.1.5, a column can have no more than a single domain index attached to it, which is keeping with Oracle standards. However, a single Text index can contain theme information in addition to word information.

## Indexing Views

In pre-8.1.5, you can create a ConText index on a view. This might be useful when you need to index documents whose content is pieced together from different tables.

However, Oracle SQL standards does not support creating indexes on views. Therefore in 8.1.5, if you need to create and index documents whose contents are in different tables, you can create a data storage preference using the USER\_

DATSTORE object, which is new for 8.1.5. With this object, you define a procedure that synthesizes documents at install time.

**See Also:** To learn more about USER\_DATASTORE, see *Oracle8i interMedia Text Reference*.

## Procedure for Creating Index

### Pre-8.1.5

The pre-8.1.5 procedure for creating an index is

1. determine indexing preferences
2. create index preferences
3. create index policy
4. Call CTX\_DDL.CREATE\_INDEX procedure, specifying the policy

### 8.1.5

The process for creating an index is simpler because of the following

- policies do not exist in 8.1.5
- by default, the system automatically detects your language, the datatype of the text column, format of documents, and sets indexing preferences accordingly. This means that creating your own preferences is now optional for creating an index.

By default, the system expects your documents to be stored in a text column. Once this requirement is satisfied, you can create a text index using the CREATE INDEX SQL command as an extensible index of type ConText, without explicitly specifying any preferences.

**See Also:** For more information about the out-of-box defaults, see *Oracle8i interMedia Text Reference*.

The 8.1.5 procedure for creating an index is:

1. Optionally, determine your custom indexing preferences if not using defaults. In this step, you determine the following preferences:

Preference Class	Description
Datastore	How are your documents stored?
Filter	How can the documents be converted to plaintext?
Lexer	What language is being indexed?
Wordlist	How should stem and fuzzy queries be expanded?
Storage	How should the index data be stored?
Stop List	What words or themes are not to be indexed?
Section Group	How are documents sections defined?

**See Also:** For more information about the preference objects available in the 8.1.5 release, see *Oracle8i InterMedia Text Reference*.

2. Optionally, create your own custom preferences. See "[Creating Preferences](#)" in this chapter.
3. Create the Text index with the SQL command CREATE INDEX, naming your index and optionally specifying preferences. See "[Creating an Index](#)" in this chapter.

## Creating Preferences

In 8.1.5, the syntax for the `CTX_DDL.CREATE_PREFERENCE` and `CTX_DDL.SET_ATTRIBUTE` procedures have changed. In addition, the order in which you call these procedures has changed.

In 8.1.5, you create the preferences then set the attributes, which is the opposite order of what you do in pre-8.1.5.

**See Also:** For a complete list of preference objects and their associated attributes, and the syntax for the `CTX_DDL.CREATE_PREFERENCE` and `CTX_DDL.SET_ATTRIBUTE` procedures, see the *Oracle8i interMedia Text Reference*.

### Example: Specifying File Data Storage

The following example creates a custom data storage preference called *mypref* that tells the system that the files to be indexed are stored in the operating system. The example then uses `CTX_DDL.SET_ATTRIBUTE` to set the `PATH` attribute of to the directory `/docs`.

```
begin
ctx_ddl.create_preference('mypref', 'FILE_DATASTORE');
ctx_ddl.set_attribute('mypref', 'PATH', '/docs');
end;
```

**See Also:** For more information about data storage, see *Oracle8i interMedia Text Reference*.

## Creating an Index

### Pre-8.1.5

In pre-8.1.5, you create an index using `CTX_DDL.CREATE_INDEX` and name a policy.

### 8.1.5

In 8.1.5, you create the Text index as a type of extensible index using the `CREATE INDEX SQL` command. You name the index and optionally specify the preferences such as `lexer` and `filter` in the parameter string.

**See Also:** To learn more about the `CREATE INDEX` command syntax, see the *Oracle8i InterMedia Text Reference*.

#### Create Index Example

The following example creates a Text index called *newsindex* on the *news* column in *mytable*. The index is created with the `lexer` preference called *my\_lexer* and the `stoplist` called *my\_stop*. Default attributes are used for the unspecified preferences.

```
create index newsindex on mytable(news) indextype is ctxsys.context
  parameters('lexer my_lexer stoplist my_stop');
```

## Dropping a Preference

### Pre-8.1.5

In pre-8.1.5, you drop preferences using `CTX_DDL.DROP_PREFERENCE`, and you can only do so when all referenced policies have been deleted from the data dictionary.

### 8.1.5

In 8.1.5, you drop index preferences with the same procedure `CTX_DDL.DROP_PREFERENCE`. Because preferences exist separately from the index and because policies do not exist in 8.1.5, you need not drop your index before you drop a preference.

Dropping a preference does not affect the index that is using the dropped preference.

**See Also:** To learn more about the syntax for the `CTX_DDL.DROP_PREFERENCE` procedure, see the *Oracle8i interMedia Text Reference*.

### Example

The following code drops the preference *my\_lexer*:

```
begin
ctx_ddl.drop_preference('my_lexer');
end;
```



## Dropping an Index

### Pre-8.1.5

In pre-8.1.5, you drop an index using `CTX_DDL.DROP_INDEX`.

### 8.1.5

In 8.1.5, you drop an index using the `DROP INDEX` command in SQL.

For example, to drop an index called *newsindex*, issue the following SQL command:

```
drop index newsindex;
```

If Oracle cannot determine the state of the index, for example as a result of an indexing crash, you cannot drop the index as described above. Instead use:

```
drop index newsindex force;
```

**See Also:** To learn more about the `DROP INDEX` command syntax, see the *Oracle8i interMedia Text Reference*.

## Resuming Failed Index

### Pre-8.1.5

In pre-8.1.5, when an indexing operation fails (creation or optimization), you can resume the operation using `CTX_DDL.RESUME_FAILED_INDEX`.

### 8.1.5

In *interMedia Text* 8.1.5, you resume a failed index operation using the `ALTER INDEX` command.

**See Also:** To learn more about the `ALTER INDEX` command syntax, see the *Oracle8i interMedia Text Reference*.

#### Example

The following command resumes the indexing operation on *newsindex* with 2 megabytes of memory:

```
ALTER INDEX newsindex rebuild parameters('resume memory 2M');
```

## Rebuilding an Index

You can rebuild a valid index using ALTER INDEX. You might rebuild an index when you want to index with a new preference.

**See Also:** To learn more about the ALTER INDEX command syntax for rebuilding an index, see the *Oracle8i InterMedia Text Reference*.

### Example

The following command rebuilds the index, replacing the lexer preference with *my\_lexer*.

```
ALTER INDEX newsindex rebuild parameters('replace lexer my_lexer');
```

## Optimizing an Index

### Pre-8.1.5

In pre-8.1.5 to optimize an index, you use `CTX_DDL.OPTIMIZE_INDEX` and specify one of five different optimizing methods.

### 8.1.5

In 8.1.5 to optimize an index, you use the `ALTER INDEX` command in SQL with the `REBUILD` parameter. You can optimize the index in either *fast* or *full* mode.

**See Also:** To learn more about optimizing the index with `ALTER INDEX`, see the *Oracle8i interMedia Text Reference*.

## Updating the Index - Background DML

As in pre-8.1.5, the 8.1.5 Text index is updated automatically whenever there is an insert, delete, or update to the base table. A *ctxsrv* server must be running. This is known as background DML processing.

The following example starts a server and writes all server messages to a file named *ctx.log*:

```
ctxsrv -user ctxsys/ctxsys -personality M -log ctx.log &
```

**See Also:** To learn more about background DML with *ctxsrv*, see the specification for *ctxsrv* in the *Oracle8i interMedia Text Reference*.

## Updating the Index - Batch DML

### Pre-8.1.5

In pre-8.1.5, you synchronize the index using CTX\_DML.SYNC. In addition, a ConText M server must be running.

### 8.1.5

You can update your index in batch mode by executing the `ALTER INDEX` command with the `sync` parameter. When you synchronize the index in batch mode, Oracle processes pending updates and inserts stored in the DML queue.

Because synchronizing an index in batch works on batches of inserts, updates and deletes, batch DML usually results in less index fragmentation than synchronizing the index immediately by running the `ctxsrv` daemon.

---

---

**Note:** No background `ctxsrv` server is required to synchronize an index in batch. If the `ctxsrv` daemon is running, it synchronizes the index immediately.

---

---

**See Also:** To learn more about the `ALTER INDEX` command syntax, see the *Oracle8i interMedia Text Reference*.

### Example

The following example synchronizes the index with a runtime memory of 2 megabytes:

```
ALTER INDEX newsindex rebuild PARAMETERS('sync memory 2M');
```

## Stoplists and Stopwords

### Pre-8.1.5

In pre-8.1.5 a stoplist consisted of words that are not to be indexed. You recorded these words by calling `CTX_DDL.SET_ATTRIBUTE` for each stopword and then by creating a stoplist preference with `CTX_DDL.CREATE_PREFERENCE`.

Default stoplists in most of the supported languages are available. You manually set the stoplist for your language.

### 8.1.5

#### Default Stoplist

By default, the system sets the default stoplist to the language you specify in your database setup. There is no need to create or set stoplists, unless you want to customize the list.

#### Stopthemes and Stopclasses

In addition to defining your own stopwords in 8.1.5, you can define stopthemes, which are themes that are not to be indexed. This is available for English only.

You can also specify that numbers are not to be indexed. A class of alphanumeric characters such as numbers that is not to be indexed is a *stopclass*.

You record your own stopwords, stopthemes, stopclasses by creating a single stoplist, to which you add the stopwords, stopthemes, and stopclasses. You specify the stoplist in the *paramstring* for `CREATE INDEX`.

#### New Procedures

In 8.1.5, you use the following procedures to manage stopwords, stopthemes, and stopclasses:

- `CTX_DDL.CREATE_STOPLIST`
- `CTX_DDL.ADD_STOPWORD`
- `CTX_DDL.ADD_STOPTHEME`
- `CTX_DDL.ADD_STOPCLASS`
- `CTX_DDL.REMOVE_STOPWORD`

- CTX\_DDL.REMOVE\_STOPTHEME
- CTX\_DDL.REMOVE\_STOPCLASS
- CTX\_DDL.DROP\_STOPLIST

**See Also:** To learn more about using these commands, see the *Oracle8i interMedia Text Reference*.



## Document Sections

Defining document sections before you index enables you to query within the sections using the WITHIN operator. You define sections as part of a section group.

### Pre-8.1.5

In pre-8.1.5, you create a section group and specify it in the Wordlist preference. You can create only user-defined zone sections and sentence and paragraph sections.

### 8.1.5

#### Section Groups

In 8.1.5, you create a section group and specify it in the *paramstring* for CREATE INDEX. To create a section group, use CTX\_DDL.CREATE\_SECTION\_GROUP.

**See Also:** to learn more about using CTX\_DDL.CREATE\_SECTION\_GROUP, see its specification in the *Oracle8i interMedia Text Reference*.

Within a section group, you can create three types of sections:

- [Zone Sections](#) (formerly user-defined section)
- [Field Sections](#) (new)
- [Special Sections](#) (sentence and paragraph sections)

#### Zone Sections

Zone sections (formerly known as user-defined sections in pre-8.1.5) are sections delimited by start and end tags. The <B> and </B> tags in HTML for instance, marks a range of words which are to be rendered in boldface.

Zone sections can be nested within one another, can overlap, and can occur more than once in a document.

You create zone sections as part of a section group with CTX\_DDL.ADD\_ZONE\_SECTION.

**See Also:** to learn more about using CTX\_DDL.ADD\_ZONE\_SECTION, see its specification in the *Oracle8i interMedia Text Reference*.

## Field Sections

Field sections are new for 8.1.5. Field sections are delimited by start and end tags. By default, the text within field sections are indexed as a sub-document separate from the rest of the document.

Unlike zone sections, field sections cannot nest or overlap. As such, field sections are best suited for non-repeating, non-overlapping sections such as TITLE and AUTHOR sections in news type documents.

Because of how field sections are indexed, WITHIN queries on field sections are usually faster than WITHIN queries on zone sections.

You create a field section as part of a section group using CTX\_DDL.ADD\_FIELD\_SECTION procedure.

**See Also:** to learn more about using CTX\_DDL.ADD\_FIELD\_SECTION, see its specification in the *Oracle8i interMedia Text Reference*.

## Special Sections

In 8.1.5, special sections are the same as paragraph and sentence sections in pre-8.1.5.

To create sentence and paragraph sections, use the CTX\_DDL.ADD\_SPECIAL\_SECTION procedure.

**See Also:** to learn more about using CTX\_DDL.ADD\_SPECIAL\_SECTION, see its specification in the *Oracle8i interMedia Text Reference*.

This chapter describes how to migrate your pre-8.1.5 queries to *interMedia* Text 8.1.5. The following topics are covered:

- [Overview of Text Queries](#)
- [Text Query](#)
- [Cursor Query](#)
- [Structured Text Query](#)
- [Theme Query \(English Only\)](#)
- [Composite Textkey Query](#)
- [Max and First/Next Operators](#)
- [PL/SQL Operator](#)
- [Counting Hits](#)
- [Stored Query Expressions](#)
- [Query Explain Plan](#)

## Overview of Text Queries

The basic *interMedia* Text query takes a query expression, usually a word with or without operators, as input. Oracle returns all documents (previously indexed) that contain that satisfy the expression along with a relevance score for each document. Scores can be used to order the documents in the result set.

A Text query can include one or more CONTAINS clauses and one or more structured clauses.

The basic Text query has *not* changed for 8.1.5.

## Text Query Expressions

Apart from a few operators (discussed in this chapter) that are no longer supported, the basic Text query expression syntax in 8.1.5 (everything between the single quotes) is the same as in pre-8.1.5.

## Text Query Methods

In pre-8.1.5, the system enabled you to execute queries using one of three methods, namely the one-step, two-step, or cursor query, formerly known as an in-memory query.

In 8.1.5, Oracle no longer supports the two-step method that uses the PL/SQL CONTAINS procedure followed by a join on the result and base table. The two-step query functionality is available through the new Text query which uses the SELECT statement.

In 8.1.5, the only query method is the standard SQL SELECT statement in which you use the CONTAINS operator in the WHERE clause. As this query is standard SQL, you can use it programatically wherever you can use the SELECT statement, such as in PL/SQL cursors.

## Text Query

In 8.1.5, the Text query replaces the pre-8.1.5 two-step method. The Text query is akin to the pre-8.1.5 one-step query in so far as it is executed with a single SELECT statement. In addition, the new 8.1.5 Text query uses no result tables.

This section describes how to migrate your two-step queries to the new Text query.

### Pre-8.1.5 Method

In the pre-8.1.5 method, you create a result table as follows:

```
create table CTX_TEMP(
  textkey varchar2(64),
  score number,
  conid number);
```

Alternatively, you can also create a result table using CTX\_QUERY.GETTAB.

You execute the CONTAINS procedure as follows:

```
execute ctx_query.contains('ARTICLE_POLICY', 'petroleum', 'CTX_TEMP');
```

You then join the result table with the base table to retrieve the document text as follows:

```
SELECT SCORE, title
FROM CTX_TEMP, TEXTTAB
WHERE texttab.PK=ctx_temp.textkey
ORDER BY SCORE DESC;
```

### 8.1.5 Method

#### SQL Example

In the SELECT statement, specify the query in the WHERE clause with the CONTAINS operator. Also specify the SCORE operator to return the score of each hit in the hitlist. The following example shows how to issue a query:

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

You can order the results from the highest scoring documents to the lowest scoring documents using the ORDER BY clause as follows:

```
SELECT SCORE(1), title from news
```

```
WHERE CONTAINS(text, 'oracle', 1) > 0
ORDER BY SCORE(1) DESC;
```

### PL/SQL Example

In a PL/SQL application, you can use a cursor to fetch the results of the query. The following example issues a query against the NEWS table to find all articles that contain the word *oracle*. The titles and scores of the first ten hits are output to standard out.

```
declare
  rowno number := 0;
begin
  for c1 in (SELECT SCORE(1) score, title FROM news
            WHERE CONTAINS(text, 'oracle', 1) > 0
            ORDER BY SCORE(1) DESC)
  loop
    rowno := rowno + 1;
    dbms_output.put_line(c1.title||': '||c1.score);
    exit when rowno = 10;
  end loop;
end;
```

This example uses a cursor FOR loop to retrieve the first ten hits. An alias *score* is declared for the return value of the SCORE operator. The score and title are output to standard out using cursor dot notation.

You can also optimize this query for response time.

**See Also:** For more information about optimizing for response time, see "[Cursor Query](#)" in this chapter.

## Cursor Query

In pre-8.1.5, you use a cursor query, formerly known as an in-memory query, over a Text query when you want only a small portion of a potentially large hitlist.

In 8.1.5, the PL/SQL interface for in-memory queries is obsolete. This means that the following procedures are obsolete in 8.1.5:

- CTX\_QUERY.OPEN\_CON
- CTX\_QUERY.FETCH\_HIT
- CTX\_QUERY.CLOSE\_CON
- CTX\_QUERY.COUNT\_LAST

To migrate pre-8.1.5 in-memory queries, use a cursor. Use the `FIRST_ROWS` hint in the `SELECT` statement to obtain the first `n` hits of a potentially large hitlist.

### Pre-8.1.5 Method

The following in-memory query finds all documents that contain the word *oracle* and returns them in score sorted order. The mechanism of the query returns the hits row by row in order, thus allowing you to extract the first `n` hits without spending the overhead of obtaining the entire hitlist first.

```
declare
  pk  varchar2(80);
  scr number;
  cur number;
begin
  cur := ctx_query.open_con('mypolicy', 'oracle', TRUE);
  while (ctx_query.fetch_hit(cur, pk, scr) > 0)
  loop
    -- deal with hit
  end loop;
  ctx_query.close_con(cur);
end;
```

### 8.1.5 Method

The pre-8.1.5 cursor query procedures `OPEN_CON`, `FETCH_HIT`, `CLOSE_CON`, `COUNT_LAST` are obsolete in 8.1.5.

To obtain the first n hits of a potentially large hitlist, execute the CONTAINS query using a cursor. Use the FIRST\_ROWS hint to optimize for response time in the SELECT statement as follows:

```
begin
  for c1 in (select /*+ FIRST_ROWS */ pk, score(1) scr
            from basetable
            where contains(textcol, 'oracle', 1) > 0
            order by scr desc)
  loop
    -- deal with hit
    dbms_output.put_line('KEY is '||c1.pk);
    dbms_output.put_line('SCORE is '||c1.scr);
  end loop;
end;
```

**See Also:** To learn more about using the FIRST\_ROWS hint with CONTAINS queries, see the *Oracle8i interMedia Text Reference*.



## Structured Text Query

A structured Text query, also called a mixed query, is a query that has a CONTAINS predicate to query a text column and has another predicate to query a structured data column.

In pre-8.1.5, you specified the structured predicate as a parameter to the CONTAINS procedure.

In 8.1.5, a Text query uses standard SQL. To issue a structured query, you specify the structured clause in the WHERE condition of the SELECT statement.

### Pre-8.1.5 Method

#### Example1: CONTAINS struct\_query Parameter

To query on structured columns, you use the *struct\_query* parameter in the CONTAINS procedure. The following example returns all articles that contain the word *oracle* that were written on or after October 1st, 1996:

```
exec ctx_query.contains('news','oracle','res_tab',
struct_query => 'issue_date >= (''1-OCT-1996'')
```

#### Example 2: Two-Step Join Method

In older versions of ConText, the *struct\_query* parameter is not available in the CONTAINS procedure. In these releases, you specify the structured condition when you join the result and base table.

For example, a query on the word *oracle* against a base table CTX\_TEMP, looks like this:

```
execute ctx_query.contains('ARTICLE_POLICY','oracle','CTX_TEMP')
```

When you join the result table with the base table, you specify the structured condition to retrieve the document text as follows:

```
SELECT score, title
FROM CTX_TEMP, TEXTTAB
WHERE texttab.PK=ctx_temp.textkey
      AND texttab.issue_date >= ('01-OCT-96')
ORDER BY score DESC;
```

## 8.1.5 Method

Specify the structured condition in the WHERE condition of the SELECT statement. The following SELECT statement does the same thing as the above query. It returns all articles that contain the word *Oracle* that were written on or after October 1st, 1997:

```
SELECT SCORE(1), title, issue_date from news
      WHERE CONTAINS(text, 'oracle', 1) > 0
      AND issue_date >= ('01-OCT-97')
      ORDER BY SCORE(1) DESC;
```

## Theme Query (English Only)

A theme query is a query on a concept. The query string is usually a concept or theme that represents the idea to be searched on. Oracle returns the documents that contain the theme.

In pre-8.1.5, you issue a theme query by first creating a theme policy to create a separate theme index. You then specify the theme policy in the CONTAINS procedure.

In 8.1.5, a single Text index contains word and theme information. You issue them queries using the ABOUT operator.

### Pre-8.1.5 Method

To issue a theme query, you first index your text column with a policy that has a theme lexer associated with it. To issue the query, you specify the same theme policy and the string for the theme query.

For example, assuming that THEME\_POL is a theme policy, you retrieve all documents about the theme of *insects* using a two-step query as follows:

```
execute ctx_query.contains('THEME_POL', 'insects', 'CTX_TEMP');
```

### 8.1.5 Method

Word information and theme information are combined into a single index. To issue a theme query, your index must have a theme component

**See Also:** For more information about creating a theme component to your index, see [Chapter 5, "Indexing"](#).

You issue a theme query using the ABOUT operator inside the query expression. For example, to retrieve all documents that are about *insects*, write your query as follows:

```
SELECT SCORE(1), title FROM news
       WHERE CONTAINS(text, 'about(insects)', 1) > 0
       ORDER BY SCORE(1) DESC;
```

**See Also:** For more information about using the ABOUT operator, see *Oracle8i InterMedia Text Reference*

## Composite Textkey Query

Composite textkey queries are queries on a base table that is indexed with a composite textkey.

### Pre-8.1.5 Method

The first step in issuing a composite textkey query is to create a result table manually with a composite textkey consisting of two columns as follows:

```
create table CTX_TEMP2(  
    textkey varchar2(64),  
    textkey2 varchar2(64),  
    score number,  
    conid number);
```

You then join the result and base table with and AND operator in the WHERE condition. For example:

```
exec ctx_query.contains('ARTICLE2_POLICY','petroleum','CTX_TEMP2')  
SELECT score, title  
FROM CTX_TEMP2, TEXTTAB2  
WHERE texttab2.PK=ctx_temp2.textkey AND  
    texttab2.PK2=ctx_temp2.textkey2  
ORDER BY score DESC;
```

### 8.1.5 Method

The 8.1.5 query is a basic SELECT statement. Because no result tables are used, there is no join between a result table and a base table as in pre-8.1.5 CONTAINS. You thus issue queries against a composite textkey table the same way you issue a query against a table with a single column textkey.

If *textab2* is the composite textkey table, the above query is written as:

```
SELECT SCORE(1),title FROM textab2  
    WHERE CONTAINS(text,'petroleum') > 0  
    ORDER BY SCORE(1) DESC;
```

## Max and First/Next Operators

The max and first/next result-set operators are no longer supported in *interMedia* Text 8.1.5.

### Pre-8.1.5 Method

#### Max

The Max operator is used to obtain a given number of the highest scoring documents in a query result set. For example, to obtain the twenty highest scoring documents that contain the word dog, you can write:

```
'dog:20'
```

#### First/Next

The first/next operator is used to obtain a range of documents in an unsorted query result-set. For example, to obtain documents 11 through twenty that contain the word dog, you can write:

```
'dog#11-20'
```

### 8.1.5 Method

The max and first/next operators are not supported in *interMedia* Text 8.1.5. You can use a cursor query optimized for response time in PL/SQL to achieve the results for a max or first/next type of query.

#### Solution for Max

A query optimized for response time provides a fast solution for when you need the highest scoring documents from a hitlist.

The example below returns the first twenty hits to standard out. This example uses the `FIRST_ROWS` hint and a cursor.

```
declare
cursor c is
  select /*+ FIRST_ROWS */ title, score(1) score
    from news
   where contains(txt_col, 'dog', 1) > 0
   order by score(1) desc;
begin
  for c1 in c
```

```
    loop
      dbms_output.put_line(c1.score||':'||substr(c1.title,1,50));
      exit when c%rowcount = 20;
    end loop;
end;
/
```

**See Also:** To learn more about optimizing queries for response time, see the *Oracle8i interMedia Text Reference*.

### Solution for First/Next

A query optimized for response time provides a fast solution for when you need a range of documents from a hitlist sorted by score.

The solution is similar to the max doc solution in that it uses the `FIRST_ROWS` hint in a cursor. The code loops through the cursor to process only the hits in the required range. The example below returns the sorted documents 11 to 20 to standard out.

```
declare
cursor c is
  select /*+ FIRST_ROWS */ title, score(1) score
    from news
   where contains(txt_col, 'dog', 1) > 0
   order by score(1) desc;
begin
  for c1 in c
  loop
    if (c%rowcount > 10) then
      dbms_output.put_line(c1.score||':'||substr(c1.title,1,50));
    end if;
    exit when c%rowcount = 20;
  end loop;
end;
/
```

**See Also:** To learn more about optimizing queries for response time, see the *Oracle8i interMedia Text Reference*.

## PL/SQL Operator

Oracle8i *interMedia* Text 8.1.5 no longer supports the *execute* operator which allows you to call a PL/SQL function in a query.

As a result of *interMedia* Text's integration with Oracle8, you can use standard SQL, which allows you to call functions in a SELECT statement as long as the function satisfies the requirements for being named in a SQL statement.

### Pre-8.1.5 Method

Calling a PL/SQL function within a query is useful for converting words to alternate forms. For example, assuming the function *french* returns the French equivalent of English words, you as *ctxuser* can search on the French word for *cat* by issuing:

```
'@ctxuser.french(cat)'
```

### 8.1.5 Method

You can call user functions directly in the CONTAINS clause as long as the function satisfies the requirements for being named in a SQL statement. The caller must also have EXECUTE privilege on the function.

For example, assuming the function *french* returns the French equivalent of English words, you can search on the French word for *cat* by writing:

```
SELECT SCORE(1), title from news
WHERE CONTAINS(text, french('cat'), 1) > 0
ORDER BY SCORE(1);
```

**See Also:** For more information about creating user functions and calling user functions from SQL, see *Oracle8 SQL Reference*.

## Counting Hits

The *interMedia* Text 8.1.5 release supports the `CTX_QUERY.COUNT_HITS` function, which you use in pre-8.1.5 to count the number of hits in a query before issuing the query. However in 8.1.5, you specify the index name rather than a policy. In addition, the *struct\_query* parameter used in pre-8.1.5 to specify the structured predicate is obsolete.

In 8.1.5, to count the number of hits returned from a query with only a `CONTAINS` predicate, you can use `CTX_QUERY.COUNT_HITS` or `COUNT(*)` in a `SELECT` statement.

To count the number of hits returned from a query that contains a structured predicate, use the `COUNT(*)` function in a `SELECT` statement.

Because in-memory queries are obsolete in 8.1.5, the pre-8.1.5 procedure `CTX_QUERY.COUNT_LAST` procedure is also obsolete in 8.1.5.

### Pre-8.1.5 Method

You count query hits with `COUNT_HITS` as follows:

```
declare count number;
begin
  count := ctx_query.count_hits(policy_name=>my_pol, text_query=> 'oracle' ,
                               exact => TRUE);
  dbms_output.put_line('Number of docs with oracle:');
  dbms_output.put_line(count);
end;
```

### 8.1.5 Method

#### **CONTAINS Predicate Only**

To find the number of documents that contain the word *oracle*, you can do one of the following:

- Issue the query with the SQL `COUNT` function as follows:

```
SELECT count(*) FROM news WHERE CONTAINS(text, 'oracle', 1) > 0;
```

- Use `COUNT_HITS` as follows:

```
declare count number;
begin
  count := ctx_query.count_hits(index_name=>my_index, text_query=> 'oracle' ,
```



```
                exact => TRUE);  
dbms_output.put_line('Number of docs with oracle:');  
dbms_output.put_line(count);  
end;
```

**See Also:** To learn more about the syntax of CTX\_QUERY.COUNT\_HITS, see the *Oracle8i InterMedia Text Reference*.

### Structured Predicate

To find the number of documents returned by a query with a structured predicate, use count(\*) as follows:

```
SELECT count(*) FROM news WHERE CONTAINS(text, 'oracle', 1) > 0 and author =  
'jones';
```

## Stored Query Expressions

In pre-8.1.5 you can store the definition and results of a query. You can then use the SQE operator in a query expression to obtain the results. For queries such as wildcard queries, using stored query expression improves performance since results are stored.

In 8.1.5, the procedure CTX\_QUERY.STORE\_SQE stores only the definition of the query. No results are stored. Referencing the query with the SQE operator merely references the definition of the query. In this way, SQEs make it easy for defining long or often used query expressions.

Stored query expressions are not attached to an index. When you call CTX\_QUERY.STORE\_SQE, you specify only the name of the stored query expression and the query expression.

The pre-8.1.5 notion of a *session* SQE has gone away. The query definitions are stored in the Text data dictionary. Any user can reference a stored query expression.

**See Also:** To learn more about the syntax of CTX\_QUERY.STORE\_SQE, see the *Oracle8i interMedia Text Reference*.

The administrative procedures of REFRESH\_SQE and PURGE\_SQE are obsolete in *interMedia Text 8.1.5*.

### Pre-8.1.5 Method

In pre-8.1.5, you define and use a stored query expression as follows:

1. Call CTX\_QUERY.STORE\_SQE to store the results for the text column or policy. With STORE\_SQE, you specify a name for the SQE, a policy (which identifies the text column for the SQE), a query expression, and whether the SQE is a session or system SQE
2. Call the stored query expression in the query expression of a text (or theme) query. ConText returns the results of the SQE in the same way it returns the results of a regular query. If the results of the SQE are out-of-date, ConText automatically re-evaluates the SQE before returning the results.

Administration of stored query expressions can be performed using the REFRESH\_SQE, REMOVE\_SQE, and PURGE\_SQE procedures in the CTX\_QUERY PL/SQL package.

**Example**

To create a session SQE named `PROG_LANG`, use `CTX_QUERY.STORE_SQE` as follows:

```
exec ctx_query.store_sqe('emp_resumes', 'prog_lang', 'computer science',
'session');
```

This SQE queries the text column for the `EMP_RESUMES` policy (in this case, `EMP.RESUMES`) and returns all documents that contain the term *cobol*. It stores the results in the SQE table for the policy.

`PROG_LANG` can then be called within a query expression as follows:

```
select score, docid from emp
where contains(resume, 'sqe(prog_lang)')>0
order by score;
```

**8.1.5 Method**

You define and use a stored query expression as follows:

1. Call `CTX_QUERY.STORE_SQE` to store the results for the text column. With `STORE_SQE`, you specify a name for the SQE and a query expression. The session and system parameters have gone away.
2. Call the stored query expression in a query expression using the SQE operator. Oracle returns the results of the SQE in the same way it returns the results of a regular query. The query is evaluated at the time the SQE is called.

The procedures `REFRESH_SQE` and `PURGE_SQE` are obsolete. You delete using `REMOVE_SQE`.

**Example**

The following example creates a stored query expression called *disaster* that searches for documents containing the words *tornado*, *hurricane*, or *earthquake*:

```
begin
ctx_query.store_sqe('disaster', 'tornado | hurricane | earthquake');
end;
```

To execute this query in an expression, write your query as follows:

```
SELECT SCORE(1), title from news
WHERE CONTAINS(text, 'SQE(disaster)', 1) > 0
ORDER BY SCORE(1);
```

**See Also:** To learn more about the syntax of CTX\_QUERY.STORE\_SQE, see the *Oracle8i interMedia Text Reference*.

## Query Explain Plan

With query explain plan, formerly known as query expression feedback, you can obtain an execution plan of a Text query before actually issuing the query. Oracle returns the explain plan information in a table, from which you can construct a parse tree.

In *interMedia Text 8.1.5*, this feature has been renamed from *query expression feedback* to *query explain plan*. Query explain plan should not be confused with *hierarchical query feedback*, which is a new, different feature.

The user interface for query explain plan has changed in the following ways:

- the procedure `CTX_QUERY.FEEDBACK` has been renamed to `CTX_QUERY.EXPLAIN`
- in `CTX_QUERY.EXPLAIN`, you specify index name rather than policy
- the `FEEDBACK_ID` column in the result table has been renamed to `EXPLAIN_ID`
- the `OPERATION` column in the explain table has new `ABOUT` operation value
- `FIRST_NEXT_DOC` and `MAX_DOC` operation column values have been removed
- the numbers associated with `WEIGHT` and `THRESHOLD` have moved from `OPTIONS` column to the `OBJECT_NAME` column.

**See Also:** For more information on the syntax of `CTX_QUERY.EXPLAIN` and the structure of the explain table, see *Oracle8i interMedia Text Reference*.

## Obtaining Explain Information

You use the `CTX_QUERY.EXPLAIN` to obtain expression feedback. The procedure for obtaining this information has not changed from pre-8.1.5 to 8.1.5. You must do the following:

1. Create the explain table.
2. Execute `CTX_QUERY.EXPLAIN`
3. Retrieve data from explain table.
4. Optionally, construct expansion tree from table information.

You must use the new 8.1.5 explain syntax when you code the first three steps.

The way you construct and expansion tree from the explain table in step 4 is the same as in pre-8.1.5

**See Also:** For examples on constructing the expansion tree, see the CTX\_QUERY.EXPLAIN command syntax in the *Oracle8i interMedia Text Reference*.

---

# Document Presentation

This chapter describes how to migrate document presentation. The following topics are covered:

- [Highlighting](#)
- [Obtaining List of Themes, Gists, and Theme Summaries](#)

## Highlighting

In *interMedia* Text query applications, you can present selected documents with query terms highlighted for text queries or with themes highlighted for ABOUT queries.

You can generate three types of output associated with highlighting: a marked-up version of the document, a plain text version of the document (filtered output), and highlight offset information for the document.

In pre-8.1.5, you used the procedure `CTX_QUERY.CTX_HIGHLIGHT` the three types of output listed above, namely a marked-up version of the document, a plain text version of the document (filtered output), and highlight offset information for the document.

In *interMedia* Text 8.1.5, these three types of output are generated by three different procedures in the `CTX_DOC` (document services) package. In addition, you can get plain text and HTML versions for each type of output.

The result tables you use to store this output in 8.1.5.5 are also different from pre-8.1.5 result tables.

In *interMedia* Text 8.1.5, the output for theme highlighting is different from what is was in pre-8.1.5. In pre-8.1.5, the system highlighted paragraphs in the document that best represented the query. In *interMedia* Text 8.1.5, individual themes, which can be words or phrases, are highlighted.

### Pre-8.1.5 Method

Use `CTX_QUERY.HIGHLIGHT` to obtain highlight information, marked-up documents, and filtered documents.

For example, to highlight all the occurrences of the term *dog* in a document identified by textkey *14*, issue the following statement:

```
ctx_query.highlight (
    cspec=> 'text_policy',
    textkey => '14',
    query => 'dog',
    id=> 14,
    hightab => 'highlight_ascii',
    mutab => 'mu_ascii' );
```

This example stores the offset information in the `HIGHTAB` table and the highlighted marked-up document in the `MU_ASCII` table.



## New 8.1.5 Solutions

### Text highlighting

For text highlighting, the behavior is same as in pre-8.1.5. You supply the query, and Oracle highlights words in document that satisfy the query. You can obtain plain-text or HTML highlighting.

### Theme Highlighting

For theme queries, *interMedia* Text 8.1.5 procedures highlight and markup words or phrases that best represent the theme query. This behavior is different from pre-8.1.5 where paragraphs are highlighted for theme queries.

### Highlight Procedure

Highlight offset information is useful for when you write your own custom routines for displaying documents.

To obtain highlight offset information, use the `CTX_DOC.HIGHLIGHT` procedure. This procedure takes a query and a document, and returns highlight offset information for either plaintext or HTML formats.

With offset information, you have the freedom to highlight with different font types or colors rather than using the standard plain text markup obtained from `CTX_DOC.MARKUP`.

**See Also:** For more information about using `CTX_DOC.HIGHLIGHT`, see its specification in the *Oracle8i interMedia Text Reference*.

### Markup Procedure

The `CTX_DOC.MARKUP` procedure takes a document reference and a query, and returns a marked-up version of the document. The output can be either marked-up plaintext or marked-up HTML.

In 8.1.5, you can customize the markup sequence for HTML navigation.

**See Also:** For more information about `CTX_DOC.MARKUP`, see its specification in the *Oracle8i interMedia Text Reference*.

### **Filter Procedure**

When documents are stored in their native formats such as Microsoft Word, you can use the filter procedure `CTX_DOC.FILTER` to obtain either a plain text or HTML version of the document.

**See Also:** For more information about `CTX_DOC.FILTER`, see its specification in the *Oracle8i interMedia Text Reference*.

## Obtaining List of Themes, Gists, and Theme Summaries

The following changes have been made in 8.1.5:

- The CTX\_LING package is no longer supported. You obtain document services output list of themes, theme summaries, and Gists from the CTX\_DOC package.
- The new procedures CTX\_DOC.THEMES and CTX\_DOC.GIST replace CTX\_LING.REQUEST\_THEMES and CTX\_LING.REQUEST\_GIST. The new procedures in 8.1.5 have different specifications and the result tables have different schema.
- There is no need to submit requests to a services queue, since Oracle now handles document services requests synchronously.
- You can specify the size of the Gist or theme summary when you call CTX\_DOC.GIST.

The following table describes list of themes, Gists, and theme summaries. Their definitions have not changed in 8.1.5:

**Table 7-1**

<b>Output Type</b>	<b>Description</b>
List of Themes	A list of the main concepts of a document.  You can generate list of themes where each theme is a single word or phrase or where each theme is a hierarchical list of parent themes.
Gist	Text in a document that best represents what the document is about as a whole.
Theme Summary	Text in a document that best represents a given theme in the document.

## Pre-8.1.5 Method

### Creating Output Tables

Before you generate list of themes, theme summaries, or Gists, you must create result table to store the CTX\_LING output.

To create a theme table called CTX\_THEMES to store the list of themes from REQUEST\_THEMES, issue the following SQL statement:

```
create table ctx_themes (  
    cid      number,  
    pk      varchar2(64),  
    theme   varchar2(2000),  
    weight  number);
```

To create a Gist table called CTX\_GIST to store the Gist or theme summaries from REQUEST\_GIST, issue the following SQL statement:

```
create table ctx_gist (  
    cid      number,  
    pk      varchar2(64),  
    pov     varchar2(80),  
    gist    long);
```

### List of Themes

Use CTX\_LING.REQUEST\_THEMES to generate themes.

**Example** The following anonymous PL/SQL block generates a list of themes for document 20 by calling CTX\_LING.REQUEST\_THEMES and then CTX\_LING.SUBMIT.

```
declare handle number;  
begin  
    ctx_ling.request_themes('CTXSYS.DOC_POLICY', '20', 'CTX_THEMES');  
    handle := ctx_ling.submit;  
end;
```

### Theme Summaries and Gists

Use CTX\_LING.REQUEST\_GIST to generate theme summaries and gists.

**Example** The following anonymous PL/SQL block generates a theme summary for document 20 about the theme of *insects*. The theme summary is generated by calling CTX\_LING.REQUEST\_GIST and then CTX\_LING.SUBMIT.

```
declare handle number;
begin
ctx_ling.request_gist('CTXSYS.DOC_POLICY','20','CTX_GIST',
                    'PARAGRAPH','insects');
handle := ctx_ling.submit;
end;
```

### Full Theme Output

You can obtain a list of themes where each element in the list is a hierarchical list of parent themes. To do so, issue the following statements:

```
SQL> exec ctx_ling.set_full_themes(TRUE)
SQL> exec ctx_ling.request_themes('ctx_thidx', pk, 'ctx_themes')
SQL> exec ctx_ling.submit(200)
```

### Changing Gist Size

You change the default size of Gists using the ConText Workbench administration tool.

## New 8.1.5 Solution

The CTX\_LING package is no longer supported. The Gist and theme generation procedures are in the CTX\_DOC package. No need to explicitly submit document services requests, since requests are synchronous. No servers need to be running.

### List of Themes

A list of themes is a list of the main concepts in a document.

Use the CTX\_DOC.THEMES procedure to generate lists of themes.

**See Also:** To learn about the command syntax for CTX\_DOC.THEMES, see *Oracle8i interMedia Text Reference*.

**Theme Table** To create a theme table:

```
create table ctx_themes (query_id number,  
                        theme varchar2(2000),  
                        weight number);
```

**Single Themes** To obtain a list of themes where each element in the list is a single theme, issue:

```
begin  
ctx_doc.themes('newsindex',34,'CTX_THEMES',1,full_themes => FALSE);  
end;
```

**Full Themes** To obtain a list of themes where each element in the list is a hierarchical list of parent themes, issue:

```
begin  
ctx_doc.themes('newsindex',34,'CTX_THEMES',1,full_themes => TRUE);  
end;
```

### Gist and Theme Summary

The definition of a Gist and theme summary has not changed for 8.1.5. A Gist is the text of a document that best represents what the document is about as a whole. A theme summary is the text of a document that best represents a single theme in the document.

In 8.1.5, you can specify the size of the Gist or theme summary when you call the procedure.

Use the procedure `CTX_DOC.GIST` to generate Gists and theme summaries.

**See Also:** To learn about the command syntax for `CTX_DOC.GIST`, see *Oracle8i interMedia Text Reference*.

**Gist Table** To create a gist table:

```
create table ctx_gist (query_id number,
                     pov      varchar2(80),
                     gist     CLOB);
```

**Gists** The following example returns a default sized paragraph level Gist for document 34:

```
begin
ctx_doc.gist('newsindex',34,'CTX_GIST',1,'PARAGRAPH', pov =>'GENERIC');
end;
```

The following example generates a non-default size Gist of ten paragraphs:

```
begin
ctx_doc.gist('newsindex',34,'CTX_GIST',1,'PARAGRAPH', pov =>'GENERIC',
numParagraphs => 10);
end;
```

The following example generates a Gist whose number of paragraphs is ten percent of the total paragraphs in document:

```
begin
ctx_doc.gist('newsindex',34,'CTX_GIST',1, 'PARAGRAPH', pov =>'GENERIC',
maxPercent => 10);
end;
```

**Theme Summary** The following example returns a theme summary on the theme of *insects* for document with textkey 34. The default Gist size is returned.

```
begin
ctx_doc.gist('newsindex',34,'CTX_GIST',1, 'PARAGRAPH', pov => 'insects');
end;
```





---

## PL/SQL Changes

This appendix describes the changes to pre-8.1.5 PL/SQL packages and procedures. The tables in this appendix include packages and procedures that are obsolete in 8.1.5.

This appendix describes only changes to pre-8.1.5 packages and procedures. It does not list new packages and procedures for 8.1.5.

**See Also:** For a complete list of all the procedures and packages in *interMedia Text 8.1.5* including the new ones, see the *Oracle8i interMedia Text Reference*.

The following topics are covered:

- [CTX\\_ADM Package](#)
- [CTX\\_DML Package](#)
- [CTX\\_DDL Package](#)
- [CTX\\_INFO Package](#)
- [CTX\\_LING Package](#)
- [CTX\\_QUERY Package](#)
- [CTX\\_SVC Package](#)
- [CTX\\_THES Package](#)

## CTX\_ADM Package

2.x Package	Procedure	Status	Migration Notes
CTX_ADM	CHANGE_MASK	<i>Obsolete</i>	Server runs only as M personality. See <i>Oracle8i interMedia Text Reference</i>
	GET_QUEUE_STATUS	<i>Obsolete</i>	All queues, except DML queue obsolete. DML queue status (Enabled or Disabled) cannot be changed.
	RECOVER	Unchanged	None
	SET_QUERY_BUFFER_SIZE	<i>Obsolete</i>	Query pipe obsolete. Queries processed by Oracle.
	SHUTDOWN	Unchanged	None
	UPDATE_QUEUE_STATUS	<i>Obsolete</i>	All queues, except DML queue, obsolete. DML queue status (Enabled or Disabled) cannot be changed

## CTX\_DML Package

2.x Package	Procedure	Status	Migration Notes
CTX_DML	REINDEX	Changed	CTX_DML package obsolete. REINDEX moved to CTX_DOC (new package).
	SYNC	<i>Obsolete</i>	Use SQL command, ALTER INDEX ... REBUILD, to manage DML in Text indexes See <a href="#">"Updating the Index - Batch DML"</a> in Chapter 5.
	SYNC_QUERY	<i>Obsolete</i>	

## CTX\_DDL Package

2.x Package	Procedure	Status	Migration Notes
CTX_DDL	ADD_SECTION	Changed	New section types. Specify only section start tag. See " <a href="#">Document Sections</a> " in <a href="#">Chapter 5</a> .
	CLEAR_ATTRIBUTES	<i>Obsolete</i>	Use CTX_DDL.UNSET_ATTRIBUTE. See <i>Oracle8i interMedia Text Reference</i>
	CREATE_INDEX	<i>Obsolete</i>	Use SQL command, CREATE INDEX, to create Text indexes. See " <a href="#">Creating an Index</a> " in <a href="#">Chapter 5</a> .
	CREATE_POLICY	<i>Obsolete</i>	Policies obsolete. Create indexes directly for a text column.
	CREATE_PREFERENCE	Changed	New method for creating preferences. See " <a href="#">Creating Preferences</a> " in <a href="#">Chapter 5</a> .
	CREATE_SECTION_GROUP	Changed	Specify section group name and section group type. See " <a href="#">Document Sections</a> " in <a href="#">Chapter 5</a> .
	CREATE_SOURCE	<i>Obsolete</i>	Use SQL*Loader to load text.
	CREATE_TEMPLATE_POLICY	<i>Obsolete</i>	Policies obsolete. Create indexes directly for a text column
	DROP_INDEX	<i>Obsolete</i>	Use SQL command, DROP INDEX, to drop Text indexes
	DROP_INTRIG	<i>Obsolete</i>	Deleting DML triggers no longer supported
	DROP_POLICY	<i>Obsolete</i>	Policies obsolete. Create indexes directly for a text column.
	DROP_PREFERENCE	Unchanged	None
	DROP_SECTION_GROUP	Unchanged	None
	DROP_SOURCE	<i>Obsolete</i>	Text loading no longer supported through ctxload - use SQL*Loader
	OPTIMIZE_INDEX	<i>Obsolete</i>	Use SQL command, ALTER INDEX ... REBUILD, to optimize Text indexes. See " <a href="#">Optimizing an Index</a> " in <a href="#">Chapter 5</a> .
	REMOVE_SECTION	Changed	New types of sections. See <i>Oracle8i interMedia Text Reference</i>
	RESUME_FAILED_INDEX	<i>Obsolete</i>	Use SQL command, ALTER INDEX .. REBUILD, to resume indexing operation. See " <a href="#">Resuming Failed Index</a> " in <a href="#">Chapter 5</a> .

---

<b>2.x Package</b>	<b>Procedure</b>	<b>Status</b>	<b>Migration Notes</b>
CTX_DDL cont.	SET_ATTRIBUTE	Changed	New method for creating preferences See " <a href="#">Creating Preferences</a> " in <a href="#">Chapter 5</a> .
	UPGRADE_INDEX	<i>Obsolete</i>	None. You must recreate indexes must for migration.
	UPDATE_POLICY	<i>Obsolete</i>	Policies obsolete. You create indexes directly for a text column.
	UPDATE_SOURCE	<i>Obsolete</i>	Text loading no longer supported through ctxload - use SQL*Loader

---

## CTX\_INFO Package

<b>2.x Package</b>	<b>Procedure</b>	<b>Status</b>	<b>Migration Notes</b>
CTX_INFO	GET_INFO	<i>Obsolete</i>	CTX_INFO package obsolete.
	GET_STATUS	<i>Obsolete</i>	
	GET_VERSION	<i>Obsolete</i>	

---

## CTX\_LING Package

2.x Package	Procedure	Status	Migration Notes
CTX_LING	CANCEL	<i>Obsolete</i>	CTX_LING package obsolete; Services queue obsolete; settings labels obsolete: <ul style="list-style-type: none"> <li>▪ Gist and theme generation performed synchronously; errors returned directly to requestor</li> <li>▪ REQUEST_GIST and REQUEST_THEMES moved to CTX_DOC (new package) and renamed (GIST and THEMES)</li> <li>▪ CTX_DOC.GIST include arguments for specifying Gist size settings.</li> <li>▪ CTX_DOC.THEMES include arguments for full theme output.</li> </ul> See <a href="#">"Obtaining List of Themes, Gists, and Theme Summaries"</a> in <a href="#">Chapter 7</a> .
	GET_COMPLETION_CALLBACK	<i>Obsolete</i>	
	GET_ERROR_CALLBACK	<i>Obsolete</i>	
	GET_FULL_THEMES	<i>Obsolete</i>	
	GET_LOG_PARSE	<i>Obsolete</i>	
	GET_SETTINGS_LABEL	<i>Obsolete</i>	
	REQUEST_GIST	<i>Obsolete</i>	
	REQUEST_THEMES	<i>Obsolete</i>	
	SET_COMPLETION_CALLBACK	<i>Obsolete</i>	
	SET_ERROR_CALLBACK	<i>Obsolete</i>	
	SET_FULL_THEMES	<i>Obsolete</i>	
	SET_LOG_PARSE	<i>Obsolete</i>	
	SET_SETTINGS_LABEL	<i>Obsolete</i>	
	SUBMIT	<i>Obsolete</i>	

## CTX\_QUERY Package

2.x Package	Procedure	Status	Migration Notes
CTX_QUERY	CLOSE_CON	<i>Obsolete</i>	In-memory query obsolete. Use FIRST_ROWS hint in SELECT statement to migrate in-memory queries. See <a href="#">"Cursor Query"</a> in <a href="#">Chapter 6</a> .
	CONTAINS	<i>Obsolete</i>	Two-step queries no longer supported - Use SELECT statement with CONTAINS function to perform queries See <a href="#">"Text Query"</a> in <a href="#">Chapter 6</a> .
	COUNT_HITS	Changed	Specify index rather than policy. The struct_query parameter is also obsolete. See <a href="#">"Counting Hits"</a> in <a href="#">Chapter 6</a> .
	COUNT_LAST	<i>Obsolete</i>	Use CTX_QUERY.COUNT_HITS or count(*) to count hits. See <a href="#">"Counting Hits"</a> in <a href="#">Chapter 6</a> .
	FEEDBACK	Changed	Procedure name changed to EXPLAIN; procedure behavior unchanged See <a href="#">"Query Explain Plan"</a> in <a href="#">Chapter 6</a> .
	FETCH_HIT	<i>Obsolete</i>	In-memory query obsolete. Use FIRST_ROWS hint in SELECT statement to migrate in-memory queries. See <a href="#">"Cursor Query"</a> in <a href="#">Chapter 6</a> .
	GETTAB	<i>Obsolete</i>	Create result tables with CREATE TABLE command. See <a href="#">Chapter 7, "Document Presentation"</a> .
	HIGHLIGHT	Replaced	HIGHLIGHT replaced by procedures in CTX_DOC (new package) See <a href="#">"Highlighting"</a> in <a href="#">Chapter 7</a> .
	OPEN_CON	<i>Obsolete</i>	In-memory query obsolete. Use FIRST_ROWS hint in SELECT statement to migrate in-memory queries. See <a href="#">"Cursor Query"</a> in <a href="#">Chapter 6</a> .
	PKDECODE	<i>Obsolete</i>	Decoding of composite textkeys no longer required, since Oracle returns rowids in cursor queries. See <a href="#">"Cursor Query"</a> in <a href="#">Chapter 6</a> .
	PKENCODE	Unchanged	None



---

<b>2.x Package</b>	<b>Procedure</b>	<b>Status</b>	<b>Migration Notes</b>
	PURGE_SQE	<i>Obsolete</i>	SQE behavior changed. See <a href="#">"Stored Query Expressions" in Chapter 6</a>
	REFRESH_SQE	<i>Obsolete</i>	SQE behavior changed; the system no longer stores results for SQEs. See <a href="#">"Stored Query Expressions" in Chapter 6</a>
	RELTAB	<i>Obsolete</i>	Drop result tables with DROP TABLE.
	REMOVE_SQE	Unchanged	None
	STORE_SQE	Changed	Stores query definition only. SQE not attached to an index. See <a href="#">"Stored Query Expressions" in Chapter 6</a>

---

## CTX\_SVC Package

<b>2.x Package</b>	<b>Procedure</b>	<b>Status</b>	<b>Migration Notes</b>
CTX_SVC	CANCEL	<i>Obsolete</i>	CTX_SVC package obsolete; Services queue obsolete:
	CANCEL_ALL	<i>Obsolete</i>	Gist and theme generation performed inline; errors returned directly to requestor
	CANCEL_USER	<i>Obsolete</i>	Index errors reported through new view, CTX_
	CLEAR_ALL_ERRORS	<i>Obsolete</i>	INDEX_ERRORS.
	CLEAR_ERROR	<i>Obsolete</i>	
	CLEAR_INDEX_ERRORS	<i>Obsolete</i>	
	CLEAR_LING_ERRORS	<i>Obsolete</i>	
	REQUEST_STATUS	<i>Obsolete</i>	

## CTX\_THES Package

2.x Package	Procedure	Status	Migration Notes
CTX_THES	CREATE_PHRASE	Changed	CREATE_PHRASE is no longer a function. It is now a procedure that must be called in the same manner as other PL/SQL procedures. The syntax for CREATE_PHRASE remains unchanged.
	CREATE_THESAURUS	Changed	CREATE_THESAURUS is no longer a function. It is now a procedure that must be called in the same manner as other PL/SQL procedures. The syntax for CREATE_THESAURUS remains unchanged.
	DROP_THESAURUS	Unchanged	None



---

---

# Index

## A

---

ABOUT query, 6-9  
    definition, 1-2  
ADD\_FIELD\_SECTION procedure, 5-18  
ADD\_SPECIAL\_SECTION procedure, 5-18  
ADD\_STOPCLASS procedure, 5-15  
ADD\_STOPTHEME procedure, 5-15  
ADD\_STOPWORD procedure, 5-15  
ADD\_ZONE\_SECTION procedure, 5-17  
administration  
    migration, 3-1  
administration tool, 3-2, 3-7  
ALTER INDEX command  
    optimizing index, 5-12  
    rebuilding index, 5-11  
    resuming failed index, 5-10  
    synchronizing index, 5-14  
alternate spelling, 1-6  
application code  
    migrating, 2-4, 2-8

## B

---

background DML, 3-2  
BASIC LEXER (pre-8.1), 4-3  
BASIC\_WORDLIST object, 4-4  
batch processing, 5-14  
BFILE columns  
    loading, 3-4  
BLASTER FILTER (pre 8.1), 4-3  
BLOB columns  
    loading, 3-4

## C

---

CHINESE V-GRAM LEXER (pre-8.1), 4-3  
CLOB columns  
    loading, 3-4  
CLOSE\_CON procedure (pre-8.1), 6-5  
code  
    migrating, 2-4  
columns with multiple indexes, 5-2  
composite textkey query  
    migrating, 6-10  
compressor object (pre-8.1), 4-4  
CONTAINS operator, 6-3  
ConText query  
    migrating, 6-2  
ConText Workbench (pre-8.1), 3-7  
COUNT\_HITS procedure, 6-14  
COUNT\_LAST procedure (pre-8.1), 6-5, 6-14  
counting hits, 6-14  
CREATE INDEX command, 5-7  
CREATE\_INDEX procedure (pre-8.1), 5-7  
CREATE\_PREFERENCE procedure, 5-6  
CREATE\_SECTION\_GROUP procedure, 5-17  
CREATE\_STOPLIST procedure, 5-15  
CTX\_ADM package, A-2  
CTX\_DDL package, A-4  
CTX\_DML package (pre-8.1), A-3  
CTX\_DOC package, 7-2, 7-5  
CTX\_INFO package (pre-8.1), A-6  
CTX\_LING output tables (pre-8.1)  
    creating, 7-6  
CTX\_LING package (pre-8.1), 7-5, A-7  
CTX\_QUERY package, A-8  
CTX\_SERVERS view, 3-2

- CTX\_SVC package (pre-8.1), A-10
- CTXADMIN role (pre-8.1), 3-3
- CTXAPP role, 3-3
- ctxctl (pre-8.1), 1-13, 3-2
- ctxkbtc compiler, 1-4, 1-13
- ctxload, 1-4, 3-4
  - changes, 1-13
- ctxsrv, 5-13
  - changes, 1-13, 3-2
  - starting and shutting down, 3-7
  - viewing status of, 3-2
- CTXSYS role, 3-3
- CTXSYS user
  - role in migration, 2-8
- CTXUSER role (pre-8.1), 3-3
- cursor query
  - migrating, 6-5
- customer support
  - contacting, xiii

## D

---

- D server, 5-14
- Danish alternate spelling, 1-6
- data migration
  - about, 2-5
  - when to perform, 2-8
- data storage
  - about, 5-5
  - preference example, 5-6
  - user, 1-4
  - USER\_DATASTORE object, 1-11
- datastore objects
  - migrating, 4-3
- DDL pipe (pre-8.1), 3-5
- default index behavior, 1-3
- DIRECT (pre 8.1), 4-3
- DIRECTORY READER (pre-8.1), 4-4
- disk requirements, 2-6
- DML processing
  - background, 1-13, 3-2, 5-13
  - batch, 5-14
- document
  - synthesizing at index-time, 1-4
- document filtering, 1-3

- document loading, 1-13, 3-4
- document presentation
  - migrating, 7-1
- document sections, 5-17
- drminst script
  - about, 4-7
  - when to run, 2-7
- drmrn script
  - about, 4-7
  - when to run, 2-7
- DROP INDEX command, 5-9
- DROP\_INDEX procedure (pre-8.1), 5-9
- DROP\_PREFERENCE procedure, 5-8
- DROP\_STOPLIST procedure, 5-16
- dropping an index, 5-9

## E

---

- engine objects
  - migrating, 4-4
- enhancements in 8.1, 1-3
- executables
  - changed and new, 1-13
- execute operator
  - migrating, 6-13
- explain information
  - obtaining, 6-19
- explain plan, 6-19
- extensible knowledge base, 1-4
  - compiling, 1-13
- extensible query optimizer, 1-3

## F

---

- features
  - new, 1-3
- feedback
  - query, 1-4
- FETCH\_HIT procedure(pre-8.1), 6-5
- field section
  - about, 5-18
  - new feature, 1-5
- filter class
  - about, 5-5
- filter objects

- migrating, 4-3
- FILTER procedure, 7-4
- filtering
  - about, 1-3
  - to HTML and plain text, 7-4
- FIRST\_ROWS hint, 6-6
- first/next operator, 6-11
  - migrating, 6-12
- full themes
  - obtaining, 7-8

## G

---

- GENERIC ENGINE (pre-8.1), 4-4
- GENERIC LOADER (pre-8.1), 4-4
- GENERIC STOP LIST (pre-8.1), 4-4
- GENERIC WORD LIST (pre 8.1), 4-4
- German alternate spelling, 1-6
- GIST procedure, 7-5, 7-9
  - example, 7-9
- Gist table, 7-9
- Gists
  - 8.1, 7-8
  - overview of differences, 7-5
  - pre-8.1, 7-6
- granting roles, 3-3

## H

---

- hardware requirements, 2-6
- hierarchical query feedback, 1-4
- HIGHLIGHT procedure
  - 8.1, 7-3
  - pre-8.1, 7-2
- highlighting
  - overview of differences, 7-2
- highlighting text, 7-3
- highlighting themes, 7-2, 7-3
- hits counting, 6-14
- HTML
  - filtering to, 1-4, 7-4
- HTML FILTER (pre-8.1), 4-3

## I

---

- index
  - about, 5-2
  - background DML, 5-13
  - creating, 5-7
  - default behavior, 1-3
  - dropping, 5-9
  - optimizing, 5-12
  - procedure for creating, 5-4
  - rebuilding, 5-11
  - synchronizing, 3-2, 5-14
  - theme component, 6-9
- index creation
  - migration script warnings, 4-10
- index migration, 4-7
  - manual, 2-4
  - plan, 2-3
  - steps, 2-8
  - with migration scripts, 2-4
- index objects
  - changes, 4-2
  - new, 1-11, 4-5
- indexes
  - multiple, 5-2
- indexing
  - resuming failed, 5-10
- indexing views, 5-2
  - warning, 4-10
- initsid.ora file, 3-6
- in-memory query, see cursor query
- Inso filter
  - about, 1-3
- installing Oracle8i, 2-7
- integration with Oracle
  - new feature, 1-3
- interMedia Text
  - related publications, xii

## J

---

- JAPANESE V-GRAM LEXER (pre-8.1), 4-3

## K

---

- knowledge base

- extending, 1-4
- knowledge base compiler, 1-13
- KOREAN LEXER (pre-8.1), 4-3

## L

---

- lexer class
  - about, 5-5
- lexer objects
  - migrating, 4-3
- list of themes
  - obtaining, 7-8
  - overview of differences, 7-5
  - pre-8.1, 7-6
- loader objects (pre-8.1), 4-4
- loading documents, 1-13, 3-4
- LOB columns
  - loading, 3-4
- location of migration scripts, 4-7
- LONG columns
  - loading, 3-4
  - migrating, 2-5

## M

---

- M personality of ctxsrv, 3-2, 5-13
- machine requirements, 2-6
- marked-up document
  - obtaining, 7-3
- MARKUP procedure, 7-3
- MASTER DETAIL (pre 8.1), 4-3
- MASTER DETAIL NEW (pre-8.1), 4-3
- max operator, 6-11
- migrate.sql, 2-4
  - editing, 4-7, 4-8
  - index creation, 4-10
  - location and function, 4-7
  - using, 4-7
  - warnings, 4-7, 4-8, 4-10
  - when to run, 2-7
- migrating
  - application code, 2-8
  - data, 2-5
  - document presentation, 7-1
  - Gists, 7-8

- highlighting, 7-2
- index, 2-3, 2-8
- index preference objects, 4-2
- list of themes, 7-8
- policies, 4-10
- queries, 6-2
- section groups, 5-17
- system-defined preferences, 4-8
- theme summary, 7-8
- user-defined preferences, 4-7

- migration
  - definition, 2-3
  - disk requirements, 2-6
  - Oracle7, 2-2
  - path, 2-2
  - plan, 2-3
  - steps, 2-7
- migration scripts
  - about, 2-4
  - using, 4-7
- mixed-format columns
  - filtering, 1-3
- multiple indexes, 5-2
- multiple policies, 4-10

## N

---

- new features, 1-3
- news group documents
  - support for sections, 1-6
- NEWS\_SECTION\_GROUP object, 1-6, 1-11
- NLS\_LEXER (pre-8.1), 4-4
- non-indexed policies
  - warnings, 4-10
- notational conventions, xiii
- NULL COMPRESSOR (pre-8.1), 4-4
- NULL FILTER (pre-8.1), 4-3
- NULL TRANSLATOR (pre 8.1), 4-4

## O

---

- obsolete
  - index objects, 4-2, 4-7
  - operators, 1-9
  - PL/SQL packages and procedures, A-1



- system-defined preferences, 4-8
- utility (ctxctl), 1-13
- offset information
  - highlight, 7-3
- one-step queries (pre-8.1), 3-6
- OPEN\_CON procedure (pre-8.1), 6-5
- operators
  - changes in 8.1, 1-9
  - obsolete, 6-11
  - SQE, 6-16
- OPTIMIZE\_INDEX procedure (pre-8.1), 5-12
- optimizing index, 5-12
- optimizing queries, 1-3
- Oracle Corporation
  - customer support, xiii
- Oracle Data Migration Assistant, 2-8
- Oracle Enterprise Manager, 3-2, 3-7
- oracle homes
  - required for migration, 2-6
- Oracle7 users
  - migration path, 2-2
- Oracle8 users
  - migration path, 2-2
- Oracle8i
  - installing, 2-7
- Oracle8i interMedia Text Manager, 3-2, 3-7
- ORDER BY clause, 6-3
- OSFILE (pre-8.1), 4-3

## P

---

- paragraph section, 5-18
- paramstring for CREATE INDEX, 5-7
- path
  - migration, 2-2
- personality of server, 1-13, 3-2
- plain-text
  - filtering to, 1-4, 7-4
- plan for migration, 2-3
- PL/SQL operator
  - migrating, 6-13
- PL/SQL packages
  - changes, A-1
  - new, 1-12
- policies, 2-3

- migrating, 4-10
- policy warnings, 4-10
- preference objects, See index objects
- preferences, 2-3
  - creating (example), 5-6
  - creating with admin tool, 3-7
  - dropping, 5-8
  - migrating system-defined, 4-8
  - migrating user-defined, 4-7
  - new system-defined, 4-6
- PURGE\_SQE procedure (pre-8.1), 6-16

## Q

---

- query
  - 8.1 PL/SQL example, 6-4
  - 8.1 SQL example, 6-3
  - composite textkey, 6-10
  - counting hits, 6-14
  - cursor (in-memory), 6-5
  - migration, 6-2
  - structured, 6-7
  - theme, 6-9
- query explain plan
  - migrating, 6-19
- query expression
  - changes, 6-2
- query feedback
  - hierarchical, 1-4
- query feedback, see query explain plan
- query methods
  - changes, 6-2
- query optimizer, 1-3
- query pipe (pre-8.1), 3-5

## R

---

- rebuilding an index, 5-11
- REFRESH\_SQE procedure (pre-8.1), 6-16
- REMOVE\_SQE procedure
  - 8.1, 6-17
- REMOVE\_STOPCLASS procedure, 5-16
- REMOVE\_STOPTHEME procedure, 5-16
- REMOVE\_STOPWORD procedure, 5-15
- REQUEST\_GIST procedure (pre-8.1), 7-5, 7-6

- REQUEST\_THEMES procedure (pre-8.1), 7-5, 7-6
- RESUME\_FAILED\_INDEX procedure
  - (pre-8.1), 5-10
- resuming failed index, 5-10
- roles
  - changes, 3-3
  - granting, 3-3

## S

---

- section group
  - creating with admin tool, 3-7
  - migrating, 5-17
  - new objects, 1-11
- section group class
  - about, 5-5
- section searching enhancements, 1-5
- SECTION\_GROUP attribute (pre-8.1), 4-4
- sections, 5-17
  - field, 1-5, 5-18
  - special, 5-18
  - zone, 5-17
- sentence section, 5-18
- server, See ctxsrv
- services queue (pre-8.1), 3-5, 7-5
- SET\_ATTRIBUTE procedure, 5-6
- single themes
  - obtaining, 7-8
- SOUNDEX\_AT\_INDEX attribute (pre-8.1), 4-4
- special sections, 5-18
- spelling
  - alternate, 1-6
- SQE operator, 6-16
- SQL
  - changes and new commands, 1-8
- SQL\*Loader, 3-4, 4-4
- STCLAUSE attribute (pre-8.1), 4-4
- steps for migration, 2-7
- stopclass, 1-5, 5-15
- stoplist
  - creating with admin tool, 3-7
  - default, 5-15
  - migration, 4-4, 5-15
  - new procedures, 5-15
- stoplist class

- about, 5-5
- stoplist enhancements, 1-5
- stoptheme, 1-5, 5-15
- stopword, 5-15
- storage class
  - about, 5-5
- storage objects
  - migrating, 4-4
- STORE\_SQE procedure, 6-16
  - 8.1, 6-17
- stored query expressions
  - migrating, 6-16
- structured query
  - 8.1 method, 6-8
  - migrating, 6-7
  - pre-8.1, 6-7
- SUBMIT procedure (pre-8.1), 7-7
- Swedish alternate spelling, 1-6
- SYNC procedure (pre-8.1), 5-14
- synchronizing index, 3-2, 5-14
- system-defined preferences
  - migrating, 4-8
  - new, 4-6

## T

---

- table
  - CTX\_LING pre-8.1, 7-6
  - Gist, 7-9
  - themes, 7-8
- template policies
  - warnings, 4-10
- terminology, 1-2
- test environment
  - creating, 2-7
- text highlighting, 7-3
- Text index
  - about, 1-3
  - definition, 1-2
- Text Manager tool, 3-7
- Text query
  - definition, 1-2
- text request queue (pre-8.1), 3-5
- text\_enable variable, 3-6
- text-only index

- about, 5-2
- theme base
  - extending, 1-4
- theme highlighting, 7-2, 7-3
- THEME LEXER (pre-8.1), 4-4
- theme query
  - migrating, 6-9
- theme summary
  - 8.1, 7-8
  - overview of differences, 7-5
  - pre-8.1, 7-6
- theme table
  - 8.1, 7-8
  - pre-8.1, 7-6
- theme-only index about, 5-2
- THEMES procedure, 7-5, 7-8
- thesaurus
  - compiling, 1-4
  - importing, 3-4
- two-step query
  - migrating, 6-2

## U

---

- unsupported operators, 1-9, 6-13
- updating index
  - background DML, 5-13
  - batch DML, 5-14
- upgrading
  - Oracle8 to Oracel8i, 2-5
- upgrading Oracle8, 2-2
- URL (pre-8.1), 4-3
- user data storage, 1-4
- USER FILTER (pre-8.1), 4-3
- USER TRANSLATOR (pre-8.1), 4-4
- USER\_DATASTORE object, 1-4, 1-11, 5-2
- user-defined preferences
  - migrating, 4-7
- users
  - changes, 3-3

## V

---

- views
  - indexing, 5-2

- new and changed system-supplied, 1-14

## W

---

- warnings in migration script
  - non-migratable policies and structures, 4-10
  - obsolete preference object, 4-7
  - obsolete system-defined preference, 4-8
- WHERE clause, 6-3
- WITHIN operator, 5-17
- word query
  - definition, 1-2
- wordlist class
  - about, 5-5
- wordlist object
  - migrating, 4-4

## X

---

- XML documents
  - support for sections, 1-5
- XML\_SECTION\_GROUP object, 1-5, 1-11

## Z

---

- zone sections, 5-17

