Oracle8*i*

Getting to Know Oracle8*i*

Release 8.1.5

February 1999

Part No. A68020-01

ORACLE

Getting to Know Oracle8*i* , Release 8.1.5

Part No. A68020-01

Primary Author: Ruth Baylis, Paul Lane

Contributors: Lance Ashdown, Mark Bauer, D. Yitzik Brenman, Dave Colello, Mary Ann Davidson, Sandy Dreskin, John Frazzini, Joe Garcia, Michael Hartstein, Thomas Kurian, Lefty Leverenz, Diana Lorentz, David Mathews, Jack Melnick, Mark Niebur, Tom Portfolio, Den Raphaely, Nasser Shariatpanahy, Deborah Steiner, Mark Townsend, Randy Urbano, V. Vijayanandan, Steven Wertheimer

Graphic Designer: Valarie Moore

Contents

3 Release 8.0 New Features and Options

4 Oracle8*i* Feature and Option Availability

5 Oracle8*i* Documentation

6 Deprecated and Desupported Features

Glossary

Index

Send Us Your Comments

Getting to Know Oracle8*i* , Release 8.1.5

Part No. A68020-01

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 the Information Development department in the following ways:

- Electronic mail - infodev@us.oracle.com
- FAX - (650) 506-7228 Attn: Server Technologies Documentation Manager
- Postal service:
  Oracle Corporation
  Server Technologies Documentation Manager
  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 book is intended to introduce Oracle8*i*, and to present the new functionality that it brings. It provides descriptions of new features, options, and enhancements; and includes feature and option factoring matrixes to identify those features and options that are available with each specific product configuration (Oracle8*i*, Oracle8*i* Enterprise Edition, or Oracle8*i* Workstation). It references the documentation that is available for Oracle8*i* and identifies deprecated or desupported features.

This preface contains the following topics:

Intended Audience

This book is addressed to anyone who would like to become familiar with Oracle8*i*. While it provides an overview of Oracle8*i*, it is intended to be of particular interest to those who desire to identify and develop an understanding of its enhancements, new features, and options.

How this Book is Organized

This book is organized as follows:

Chapter 1, "Overview"

Contains an introduction to Oracle8*i*.

Chapter 2, "Oracle8i New Features"

Contains summaries of the new features and options, offered in Oracle8*i*, release 8.1.5. Some additional products that compliment or enhance the functionality of Oracle8*i* are also discussed.

Chapter 3, "Release 8.0 New Features and Options"

Because many of the Oracle8*i* features enhance or extend functionality introduced in Oracle8, Chapter 3 maintains the summary of Oracle8 features.

Chapter 4, "Oracle8i Feature and Option Availability"

Contains the feature factoring matrixes that present the availability and packaging of features and options contained in Oracle8*i*.

Chapter 5, "Oracle8i Documentation"

Identifies Oracle8*i* documentation.

Chapter 6, "Deprecated and Desupported Features"

Lists the deprecated and desupported features that a user should be aware of when migrating to Oracle8*i*.

Glossary

Contains definitions of terms, most of which are new to Oracle8*i*.

How to Use This Book

Chapter 2, "Oracle8i New Features", can be used to identify a new option, feature, or enhancement and to read its description. For more information, you may refer to the documentation that is cross-referenced in the description.

Oracle8*i* contains a subset of the features and options that are available in Oracle8*i* Enterprise Edition and Oracle8*i* Workstation. You can refer to the matrixes in Chapter 4, "Oracle8i Feature and Option Availability", to identify if a particular option or feature is available for your product configuration.

Related Documents

While this book identifies enhancements, new features, and options for Oracle8*i*, release 8.1.5, it does not attempt to identify every new or changed initialization parameter, data dictionary table, view, command, package, or the likes. Such information should be available from the cross-referenced documentation.

However, some general references for finding such information are:

Oracle8i Migration.	For a complete list of all new Oracle8*i* initialization parameters, static data dictionary tables, and dynamic performance views (V$ views). Many initialization parameters have been added, changed, or dropped in this release.
Oracle8i Supplied Packages Reference	Any new packages are included in this book or cross-referenced by it.
Oracle8i Reference	This is the complete Oracle8*i* reference for: ■ Initialization Parameters ■ Static Data Dictionary Views ■ Dynamic Performance (V$) Views ■ Database Limits ■ SQL Scripts ■ Oracle Wait Events ■ Enqueue and Lock Names ■ Statistics Descriptions
Oracle8i SQL Reference	This is the complete reference of SQL commands. It identifies all new Oracle8*i* commands.

Your Comments Are Welcome

We value and appreciate your comments as an Oracle user and reader of our references. As we write, revise, and evaluate, your opinions are the most important input we receive. At the front of this reference is a Reader's Comment Form that we encourage you to use to tell us both what you like and what you dislike about this (or other) Oracle manuals. If the form is missing, or you would like to contact us, please use the following address or fax number:

Server Technologies Information Development Manager
Oracle Corporation
500 Oracle Parkway
Redwood Shores, CA 94065
FAX: 650-506-7228

You can also e-mail your comments to the Information Development department at the following e-mail address: infodev@us.oracle.com

1

Overview

This section presents an overview of the Oracle8*i* database server and includes the following topics:

- Introducing Oracle8i
- A Family of Database Products

Introducing Oracle8*i*

Oracle8*i*, the database for Internet computing, changes the way information is managed and accessed to meet the demands of the Internet age, while providing significant new features for traditional online transaction processing (OLTP) and data warehouse applications. It provides advanced tools to manage all types of data in Web sites, but it also delivers the performance, scalability, and availability needed to support very large database (VLDB) and mission-critical applications.

Oracle8*i* is much more than just a simple relational data store. It introduces *i*FS, the Internet File System that allows users to easily move all of their data into an Oracle8*i* database where it can be stored and managed more efficiently in an integrated fashion. A new option, Oracle8*i* *inter*Media, allows businesses to manage and access multi-media data, including image, text, audio, video, and spatial (locator) data. The *inter*Media Clipboard and Web Agent work together to Web-enable *inter*Media. WebDB is a new HTML-based development tool for building HTML Web pages with content based on data stored in Oracle databases.

Oracle8*i* introduces new support for Java by including a robust, integrated, and scalable Java Virtual Machine within the server. This expands Oracle's support for Java into all tiers of applications, allowing Java programs to be deployed where they perform best—in the client, server, or middle tier—without recompiling or modifying the Java code.

Not only does Oracle8*i* introduce significant breakthrough technology, such as a Java VM and *i*FS, it also has significant new features and functionality for traditional OLTP and data warehouse applications. For data warehouses, Oracle8*i* now provides sophisticated summary management features to store aggregates that are commonly queried, reducing query processing significantly.

For OLTP applications, Oracle8*i* introduces the optimizer plan stability feature allowing the user to ensure that the optimizer selects the same tuned plan every time the same query is executed. Advanced Queuing has been enhanced to support a publish/subscribe paradigm. OLTP applications benefit from the many new and enhanced features that improve the recoverability and availability of the database during routine operations, such as index rebuilds, and in disaster situations where a standby database may be activated.

Oracle8*i* introduces database resource management where the DBA now has the ability to control the processing resources allocated to a user or group of users. Two new partitioning methods, hash and composite, complement the established range partitioning method to provide a rich set of partitioning methods from which the DBA may choose the best method to fit an application's profile and workload.

The Oracle Parallel Server features a new architecture in Oracle8*i*. Cache fusion is a new "diskless" ping architecture that greatly improves inter-instance communication. New networking features improve the ease of use for OPS and system administration is made easier through the enhancement of Oracle Parallel Server Management, the new Oracle Universal Installer, and the Oracle Database Configuration Assistant.

Oracle8*i* extends the functionality of advanced replication, focusing on mass-deployment applications. Data can be replicated to servers that are closer to users and have only the data those users need, providing better performance. Security has been improved. The replication manager has been rewritten in Java and is no longer constrained to run on a Windows-based machine.

While Oracle has expanded its support of multimedia data through *inter*Media, and other complex data types through the separately packaged Visual Information Retrieval, Time Series, and Spatial options, it has also provided a new extensible architecture for independent software vendors (ISVs) and other developers to exploit when developing support for new types of data or specialized processing. There is significant enhancement to Oracle's object relational technology, and Oracle8*i* makes it available in both the standard and Enterprise Edition.

Language improvements for PL/SQL, Pro*C/C++, Pro*COBOL., and the Oracle Call Interface (OCI) include significant new functionality. National Language Support also undergoes substantial change, and a new book provides information on its use.

Fine-grained access control and application context features build row-level application security into the database, rather than leaving it at the application level. Stronger security is provided throughout the enterprise by expanding or enhancing methods of authentication and authorization, centralizing user management, and supporting standards.

There is greater Windows NT integration in Oracle8*i*. It provides full, native integration with MTS in the Windows NT environment. Application development is made simpler by the Oracle Application Wizard (AppWizard) for Visual Studio, which provides developers with a GUI tool for creating a Visual C++, Visual InterDev, or Visual Basic application accessing data in an Oracle database. The generated code framework will include Oracle Objects for OLE (OO4O) classes for connectivity and data access to Oracle databases.

Finally, Oracle8*i* includes Oracle Enterprise Manager, which is the comprehensive management framework for managing the Oracle database and application environment. Oracle Enterprise Manager presents an easy-to-use centralized console, a rich set of management tools, and the extensibility to detect and solve any

problems that may arise. It also includes several administrative applications for performing day-to-day tasks for databases and applications, such as scheduling backup routines.

In summary, Oracle8*i* is designed to access and manage all your data using the style and infrastructure of the Internet. Oracle8i is the most complete and comprehensive platform for building, deploying and managing Internet and traditional applications.

- It simplifies the development of applications
- It simplifies the management of Internet content
- It simplifies the deployment of applications

Oracle8i provides the lowest cost platform for developing and deploying applications on the Internet.

A Family of Database Products

Oracle8*i* is available in four standalone versions:

- Oracle8*i*, also referred to as the standard edition of Oracle8*i*, is the basic version and includes core features for most Windows NT and Unix applications.
- Oracle8*i* Enterprise Edition adds several high-end features and options for mission-critical OLTP and data warehousing applications.
- Oracle8*i* Workstation is a single-user development database for Windows NT, Windows95, or Windows98, that now includes all Enterprise Edition options and features.
- Oracle8*i* Lite is a lightweight mobile Java database that easily synchronizes laptops and hand held devices with corporate databases. It is not discussed in this book.

2

Oracle8*i* New Features

This chapter describes new features, options, and enhancements available in Oracle8*i*.

The following topics are included.

Content Management for the Internet

The Web and the Internet bring many new challenges to organizations. Today's Web sites are often a series of Web pages that take too much time and too many human resources to develop, deploy, and manage. In addition, their infrastructure may not be stable or robust enough to support mission-critical applications. Everything necessary to transform data into powerful Internet applications and content-driven Web sites is included with Oracle8*i*. It provides a Web development environment that allows software developers to easily build dynamic, data-driven Web sites with a standard Web browser and the Oracle8*i* database.

WebDB

WebDB is an HTML-based development tool for building HTML Web pages with content based on data stored in the Oracle database. Using a browser-based interface, Web pages can be created containing reports, charts, calendars, menus, and forms. WebDB provides build wizards for step by step guidance through the process of creating these components. The wizards include steps for building the PL/SQL statement that selects data used in the components, and steps for setting look and feel attributes of the component such as color and size.

There is no need to know PL/SQL to create a component. Selecting options in the build wizard guides WebDB to automatically write the PL/SQL code for building the component.

WebDB also provides a wizard to create a Web site within the Oracle database and assign a site administrator to control its development of the site. The Web site administrator in turn assigns owners who are responsible for providing content to the site. Content can include Web pages containing WebDB components, links to other URLs, and files uploaded to the Web site.

Once a site is created, it is maintained using options within the site itself. Content owners and end users never have to see the WebDB user interface. For example, the look and feel of the site can be updated using options provided within the site. End users perform searches of the site's content using options within the site.

For more information, see *Oracle WebDB Release 2.0: Getting Started - Installation and Tutorial*.

Oracle Internet File System (*i*FS)

Oracle *i*FS combines the power of Oracle8*i* with the ease of use of a file system. Completely integrated with Oracle8*i*, the Internet File System is a Java application

that runs within the Oracle8*i* Java Virtual Machine. It enables the database to become and Internet development and deployment platform.

Oracle iFS makes it possible to treat the database as if it were simply a shared network drive. Users can store and retrieve files managed by the database as if they were files managed by a file server. Because *i*FS supports a variety of standard protocols, users have universal access to their data. Whether a user accesses the contents of *i*FS through Windows Explorer, a Web browser, an FTP client, or an e-mail client, the files appear the same.

Not only can relational data appear as files, but hybrid documents that combine relational and non-relational data into single objects can be created and viewed. Files and folders in *i*Fs are stored in a repository. They are indexed on content and file properties, allowing for intelligent text searches and queries.

From a developer's standpoint, *i*FS is the single data store containing the data for many different applications. For the system administrator, *i*FS provides a single system for file storage and messaging, rather than several separate systems to maintain and administer. In short, *i*FS is a simpler way of storing different types of files in the database.

Note: For Oracle8*i*, release 8.1.5, an SDK version of *i*FS is available to selected sites, including online documentation. Oracle *i*FS is planned for general availability with the next production release of Oracle8*i*.

---

*inter*Media and the Web

Web applications require advanced data management services that support the rich datatypes used in Web repository, e-commerce, and other Internet applications. Oracle8*i* *inter*Media adds support that enables the database server to manage multimedia content, both for Internet and traditional applications that require access to image, audio, video, text, and location information. Oracle8*i* *inter*Media includes the following:

- Oracle's ConText text services, to deliver the powerful text retrieval capabilities fundamental to Web applications. These services let users query and analyze documents stored in document archives, online news feeds, customer call reports, and other online text information sources. Users can query for data in the most common formats, including HTML, Word, Excel, and Acrobat / PDF formats.

- Audio, video, and image services to support integrated management of audio, video, and image information within an Oracle8*i* database. These services enable access to audio, video, and image data in dozens of Internet formats from a variety of sources, both within Oracle8*i* and from external locations such as Web URL sites or specialized servers.
- Geometric locator services to support the development of Internet applications that help users locate information, such as stores, distribution points, and events, based on their location or distance from a given address. These *inter*Media locator services enable Oracle8*i* to perform location queries, and support the Internet's leading online geocoding services, including Centrus from QMSoft and MapXtreme from MapInfo.

Integral to *inter*Media is the *inter*Media Clipboard and Web Agent. It provides a clipboard and Web agent working together to Web-enable *inter*Media. The clipboard can be used to:

- Capture multimedia objects from files or external sources, such as cameras, audio input devices, and scanners
- Retrieve multimedia objects from an Oracle8*i* database
- Edit objects using other tools such as Paint Shop Pro or Word
- "Visualize" and store objects into the database.
- Drag and drop to multimedia objects from the database to Web applications and Web authoring tools such as Oracle WebDB, Symantec Visual Page, and Microsoft FrontPage
- Construct the simple or complex URLs for accessing objects through the *inter*Media Web agent.

The Oracle8*i* *inter*Media Web agent decodes URLs to retrieve multimedia objects for display by Web clients, such as Web browsers. The Oracle8*i* *inter*Media Web Agent supports Netscape Enterprise Server, Netscape FastTrack Server, Microsoft Internet Information Server, and Oracle Application Server.

Note: For release 8.1.5, the Clipboard and Web Agent Components of Oracle8*i* *inter*Media are not available on the Oracle8*i* distribution media. Instead, you can download these components from the "Free Software" download area of the Oracle Corporation Web site:

http://www.oracle.com/products/free_software/index.html

The documentation, which includes readme files and the book *Using Oracle8i interMedia with the Web*, is included in the download.

For a more detailed summary of *inter*Media, and references to specific documentation, see below.

Oracle8*i* *inter*Media, Spatial, Time Series, and Visual Image Retrieval

Oracle8*i* allows you to integrate all your data into its datastore. The Oracle8*i* extensibility framework and object relational technology have been exploited to extend the reliability, availability, and data management capabilities of the database server to multimedia data. A new multimedia option, Oracle8*i* *inter*Media, services text, document, image, audio, video, and locational data in a single integrated package. The Spatial, Time Series, and Visual Image Retrieval options, available separately, also have been designed to utilize the Oracle8*i* extensibility architecture.

For each of the media types, customers and partners can easily create "plug-ins" to support additional formats, new digital compression and decompression schemes (codecs), specialized indexes, custom query optimization and methods, external media data sources and even specialized data processing algorithms.

The following is a summary of functionality available with each option.

Oracle8*i* *inter*Media

Oracle8*i* *inter*Media allows multimedia data to be managed in an integral fashion with other enterprise data. Applications can access interMedia through both object and relational interfaces. Database applications written in Java, C++, or traditional 3GLs can interface to interMedia through class libraries, PL/SQL, and the Oracle Call Interface (OCI).

For a more general description of *inter*Media, refer to "interMedia and the Web" on page 2-3. This section presents specific features of the *inter*Media components and points you to supporting documentation.

Text

Oracle8*i* *inter*Media Text delivers powerful text management and retrieval capabilities. It indexes any documents or textual content to deliver fast, accurate retrieval of information from document archives, online product catalogs, news services, media asset management systems, job postings, customer call reports, and other text information sources in multiple languages. Several techniques are used to ensure fast and accurate text searching, from traditional full-text search to document theme analysis.

The base for *inter*Media Text is Oracle's ConText Cartridge. Several new features have been added, in addition to its integration into *inter*Media. Discussions of these new features follow.

For more information, see *Oracle8i ConText to interMedia Text Migration* and *Oracle8i interMedia Text Reference*.

Improved Ease of Use This is made possible through the tight integration of *inter*Media Text with SQL, PL/SQL, Oracle Enterprise Manager, SQL*Loader, and other components of the Oracle8*i* database engine. Some examples:

- Most queries can be performed using SQL, and text searches can be combined with regular database searches in a single SQL statement. Query performance is improved.
- SQL*Loader can be used to load data into LOBs.
- The administrative tool has been largely redesigned and is now a Java Applet integrated with the Oracle Enterprise Manager (OEM).
- New out-of-box default settings make indexing easy. By default, *inter*Media Text autodetects language settings, text column datatypes, and document formats, among other things.

Extensible Index Creating an index is now much simpler because it is created and maintained using standard SQL. This is made possible through the Oracle8*i* extensible indexing framework as described in "User-Defined Operators and Extensible Indexing" on page 2-31.

Extensible Query Optimizer Through new functionality provided by the Oracle8*i* extensible optimizer, described in "Extensible Optimizer" on page 2-32, the optimizer can be used to select the best plan for executing a CONTAINS query. It does so by analyzing collected statistics of all the tables and indexes affected by a CONTAINS query.

Extensible Knowledge Base (KB) Application developers and users can import a thesaurus to augment the built-in KB. User-specific concepts, categories, words, or phrases can be added to improve theme capabilities. Information in the thesaurus takes precedence over information in the built-in KB when there is a conflict.

Filters By default during indexing, *inter*Media Text uses the Inso Corporation filtering technology, which can filter most document formats. This filtering technology automatically recognizes document formats. Thus, this filter can be used to index single or mixed column formats. A custom or other filter may also be specified to filter documents during indexing.

For document presentation, the Inso filtering technology is used to create plain text or HTML versions of formatted documents.

Hierarchical Query Feedback Given a query expression, *inter*Media Text returns related query term information (broader term, narrower term, related term) that can help refine queries.

Theme Highlighting The behavior of theme highlighting procedures has been changed. For theme queries, *inter*Media Text procedures highlight and mark up words or phrases that best represent the theme query. This behavior is different from ConText, where paragraphs are highlighted for theme queries.

Alternate Spelling Certain languages use alternate spellings or representations for characters that cannot be stored as single characters: for instance, the letter "ö" in German is alternately represented as "oe". *inter*Media Text now checks for these alternate spellings during indexing and queries.

Stoplists ConText allows stoplists to specify words that are not to be indexed (*stopwords*). The following additional stoplists are introduced in *inter*Media Text.

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

Also, stoplists can now be changed dynamically, after an index has already been created.

Field Sections A new section type (*field section)*, allows text within the field section to be indexed as a subdocument separate from the rest of the document.

Audio, Image, and Video

Oracle8*i* *inter*Media provides foundational support for audio, image, and video digital information. Services provided by *inter*Media facilitate the integral storage of information in these forms into the Oracle8*i* datastore as BLOBs, or as references to external BFILEs. These *inter*Media services also include methods for retrieval of this multimedia digital information, inserting new information, and for limited manipulation.

Summary of Audio, Image, and Video Support The following table summarizes the support for audio, image, and video data provided by Oracle8*i* *inter*Media.

Media Type	Description of Support
Audio	■ Client access via Java Media Framework (JMF)--this allows any JMF player and *inter*Media to access and play audio files within an application ■ Audio delivery through any streaming server such as Oracle Video Server or RealNetworks RealAudio Server ■ Basic parsing of AUFF, AIFF, AIFF-C, and WAVE formats ■ Management of audio data from a variety of sources and formats external or internal to Oracle8*i*; e.g., BFILEs, LOBs, URLs, or specialized servers ■ Client-side access components in Java
Image (Note that Oracle's Image Cartridge has been subsumed by Oracle8*i* *inter*Media.)	■ Support for popular image formats--such as BMP, TIFF, GIF--and compression formats--such as CCITT, JPEG ■ Support for Live Picture FlashPix format server ■ Simplified Java and C++ access to images stored in Oracle8*i* ■ Management of image data from a variety of sources and formats external or internal to Oracle8*i*; e.g., BFILEs, LOBs, URLs, or specialized servers ■ Metadata extraction from image header information, conversion among image and compression formats, and support for nonstandard and custom image formats through a raw pixel format ■ Direct access to pixel data in an image ■ Support for basic manipulation functions including scaling and cropping

Media Type	Description of Support
Video	■ Client access via Java Media Framework (JMF)--this allows any JMF player and *inter*Media to access and play video files within an application ■ Video delivery through any streaming server such as Oracle Video Server or RealNetworks RealVideo Server ■ Support for popular video formats such as AVI, QuickTime, and MPEG. ■ Management of video data from a variety of sources and formats external or internal to Oracle8*i*; e.g., BFILEs, LOBs, URLs, or specialized servers ■ Client-side access components in Java

For more information, see the *Oracle8i interMedia Audio, Image, and Video User's Guide and Reference*.

Oracle8*i* interMedia Audio, Image, and Video Java Client Where a greater degree of manipulation or modification of image, audio, or video data is required, Oracle provides the *Oracle8i interMedia Audio, Image, and Video Java Client*.

The Oracle8*i* *inter*Media Audio, Image, and Video Java Client allows the user to use local (client side) applications to manipulate or modify multimedia data stored in a network-accessible (server side) database. It allows the user to connect to a server side multimedia object, copy that object from the server side to the client side, perform various operations on the client side object, and transfer the new multimedia object back to the server side.

For situations where the user does not have permission to modify the server side object, Oracle8*i* *inter*Media Audio, Image, and Video Java Client can retrieve the multimedia data from the server side for display purposes only.

In addition, Oracle8*i* *inter*Media Audio, Image, and Video Java Client gives the user the ability to integrate multimedia objects with various media frameworks, such as the Java Media Framework (JMF), RealPlayer, or Oracle Video Client (OVC).

For more information, see *Oracle8i interMedia Audio, Image, and Video Java Client User's Guide and Reference.*

Locator

Oracle8*i* *inter*Media Locator is the component of Oracle8*i* *inter*Media that supports online *geocoding* facilities for locator applications and proximity queries. It supports

geocoding of spatial point data of interest, stores this geocoded data locally in the Oracle8*i* database server, and performs within-distance queries on the geocoded data.

Geocoding represents addresses and locations of interest (postal codes, demographic regions, etc.) as geometric features (points). Geocoding services can be used to add the exact location (latitude and longitude) associated with points of interest to existing data files stored in Oracle8*i*. Once data is geocoded, Oracle8*i* *inter*Media Locator enables distances to be calculated and sites to be represented graphically in Web, data warehousing, customer information system, and enterprise resource planning applications.

A sample application might use *inter*Media Locator to locate restaurants and hotels within a given point-to-point distance of a specified location, such as a business or tourist information kiosk.

Locator is not designed to be an end-user application in itself. It consists of an *inter*Media locator object datatype, a geocode result object datatype, geocoding functions, a spatial locator index, and a spatial operator for performing within-distance queries. It supports only point geometry types. Web application developers can use the basic function provided here to build specialized Web-enabled *inter*Media Locator applications. Oracle8*i* *inter*Media Locator also supports server-based geocoding and data scrubbing operations for data warehouse applications.

For more information, see *Oracle8i interMedia Locator User's Guide and Reference*.

Oracle8*i* Spatial

Spatial data is any data with a location component. The location component could be the geocoded addresses of customers or suppliers, the course of a river and the outline of its floodplain, the locations of thousands of utility poles, or X and Y coordinates on a blueprint. Spatial data is not limited to the land surface but includes the subsurface, aquatic, marine, and lower atmospheric regions.

Oracle8*i* Spatial is designed for two groups of users:

- It enables traditional database customers to add useful spatial queries to their applications.
- It supports geographic information system (GIS) vendors who must store, retrieve, and manage very large spatial databases containing hundreds of gigabytes of geodata.

This section describes the enhancements to Oracle8*i* Spatial, formerly known as the Oracle8 Spatial Cartridge, for the current release. Additionally, this release completes the implementation of the OpenGIS Consortium RFP1 - Simple Features guidelines. Oracle8*i* Spatial conforms to the specifications associated with SQL92 representation of points, lines, and polygons.

For more information about Oracle8*i* Spatial, see *Oracle8i Spatial User's Guide and Reference*.

VARRAY Storage Model

The structure of the spatial data model has changed such that ordinates are stored using the VARRAY storage model. A spatial object type can be created that represents single element or multi-element geometries. Where in release 8.0, the cartridge had individual columns for geometry attributes, Oracle8*i* combines these into a single column of type SDO_GEOMETRY, similar to the way Oracle8*i* *inter*Media defines specialized objects for images, text, audio, and video.

Dynamic Window Queries

For dynamic window queries, the release 8.0 cartridge required that users create and maintain a window layer that stored the definition of an area-of-interest and its corresponding spatial index entries. In Oracle8*i*, the spatial operators take a bind variable of type SDO_GEOMETRY, and the index creation and usage are handled internally.

Spatial Index Mechanism

The spatial index mechanism now uses Oracle8*i* extensible indexing features, including support for maintaining the index on insert, update, and delete. As a result, it is no longer necessary to explicitly call SDO_ADMIN.POPULATE_INDEX() or SDO_ADMIN.UPDATE_INDEX().

RELATE() Operator

The RELATE() operator has been enhanced such that it can perform both primary and secondary filter operations. At query time, the Oracle8*i* extensible optimizer then determines which process to use based upon information in the query plan. The new syntax for the RELATE() operator greatly simplifies queries.

Extended Functionality

Two new spatial operators have been added.

- SDO_FILTER() implements a primary filter by determining whether the minimum bounding rectangles of the objects interact.
- WITHIN_DISTANCE() determines whether two geometries are within a specified Euclidean distance from each other.

Circle and *circular arc* geometric datatypes have been added, and several new spatial functions further extend the capabilities of this product.

Oracle8*i* Visual Information Retrieval

Oracle8*i* Visual Information Retrieval provides content-based retrieval for images stored in Oracle8*i*. Content-based retrieval--where the query takes the form "find me objects that look like this one"--reduces the time and effort required to obtain image-based information.

Users can tap into image content with self-service applications using flexible query criteria. They can locate similar images in large databases by searching visual criteria, such as color, pattern, and texture. Examples of database applications where content-based retrieval is useful include:

- Tracking images in stock photo houses, ad agencies, libraries, museums, and online networks
- Finding product information in design, manufacturing, and online shopping

Oracle8*i* Visual Information Retrieval includes foundational datatype support for images in Oracle8*i*, that complements and is completely compatible with Oracle8*i* *inter*Media. Oracle8*i* Visual Information Retrieval is based on the VIR™ technology from Virage, Inc., a leader in content-based retrieval software.

This section describes the enhancements to the Oracle8 Visual Information Retrieval Cartridge, now repackaged as Oracle8*i* Visual Information Retrieval. Also available with this release is the Oracle8*i* Visual Information Retrieval Java Client, similar in function to that described in "Oracle8i interMedia Audio, Image, and Video Java Client" on page 2-9.

For more information, see *Oracle8i Visual Information Retrieval User's Guide and Reference* and *Oracle8i Visual Information Retrieval Java Client User's Guide and Reference.*

New Extensible Indexing

Oracle8*i* Visual Information Retrieval allows users to create an index to support image objects. Oracle8*i* and Visual Information Retrieval cooperate to define, build, and maintain an index for image data. Once created, the index automatically

updates every time an image is inserted or removed from the database table. This feature significantly speeds up query time and improves performance.

Temporary LOB Usage

Consider the case where a thin client requests a scaled-down image from the database. Rather than storing the scaled-down image in a table, it is now possible to use temporary LOBs, described in "Temporary LOBs" on page 2-33. Methods are provided which allocate and operate on temporary LOBs. Additional performance benefits are achieved by utilizing OCI LOB buffering routines provided in Oracle8*i*, instead of local buffering mechanisms.

Scaling Enhancements

Scaling is now allowed to absolute dimensions. The process() method in release 8.0 only allowed scaling by a floating point number. The new scaling mechanism allows, for example, a much easier method to create a column of thumbnails for a table with images of various sizes in it.

Proprietary Image Formats

Because it is not practical to provide native support for all of the image formats in use today, Visual Information Retrieval provides a canonical raw image format gateway to ease the import of proprietary image data. This benefits the many image processing programs that can directly support input and output of uncompressed image data.

Oracle8*i* Time Series

Oracle8*i* Time Series enables time-stamped data to be stored efficiently in an Oracle database. It supports a basic set of functions--calendar, time series, and time scaling--to retrieve and process data. Its collection-based interface enables product developers and third-party providers to add functions to the base cartridge. By offering basic functions and datatypes, it allows customers and independent software vendors (ISVs) to develop data provisioning, data warehousing, and full-analysis time series applications using the power of Oracle8*i*.

Oracle8*i* Time Series provides the following kinds of functions:

- Calendar functions provide a convenient mechanism for defining time-related operations and ensuring the validity of time-related data. For example, arbitrary calendars can be defined over a fixed interval, such as a calendar of business days in the week or a calendar of quarterly dividend payment dates.

- Time series functions provide analysis of time series data and include support for complex aggregation (such as moving average), mathematical operations (such as cumulative sum and cell-by-cell arithmetic operations), and data verification.
- Time scaling functions allow a time series function to be transformed from one time scale to another, such as from aggregate daily data into quarterly summaries. Time scaling is flexible because the source and target scales are determined by calendars, which can be customized.

A summary of the new features of Oracle8*i* Time Series, formerly called the Oracle8 Time Series Cartridge, is presented below.

For more information on Oracle8*i* Time Series, see the *Oracle8i Time Series User's Guide*.

Irregular Time Series

Support is provided for irregular time series, that is, time series without an associated calendar. Irregular time series provide a means to conveniently handle high volumes of unpredictable data, as well as many applications with predictable data where timestamp validation is not needed.

Additional Calendar Frequencies

Added are *week*, *10-day*, *semimonthly*, *quarter*, and *semiannual* calendar frequencies.

Flexible Timestamp Precisions

Flexible timestamp precisions in Oracle8*i* Time Series relax the precision requirements of the previous release. For example, a monthly calendar now can be defined where all timestamps are on almost any specific day of the month (such as the 15th), whereas before they had to be on the first of the month.

Flexible Patterns

The calendar pattern has been extended to provide flexible support of a repeated pattern of included and omitted timestamps. The pattern specification extensions allow for more semantic (behavioral) options with scaleup and scaledown functions.

Time Scaling Enhancements

Time scaling enhancements in Oracle8*i* include the implementation of scaledown functions and additional overloadings of scaling functions to provide an alternative to SQL null semantics.

Nested Table Support

Previously, the Time Series Cartridge supported a storage model for time series data based on a flat index-organized table (IOT). Oracle8*i* Time Series provides alternative support for a nested IOT model, enabling time series data to be stored in a nested IOT. Either method has advantages and disadvantages that must be weighed by the user to select the method providing the better performance and scalability.

Ease of Use

Administrative tools and procedures are provided to simplify the creation and use of time series schema objects. For most users, these procedures are a convenient alternative to explicitly defining the tables, views, and triggers used by Oracle8*i* Time Series.

Java

Java is becoming the standard language of the Internet. In Oracle8*i*, Oracle delivers an enterprise-class Java platform to develop and deploy Internet applications. Additionally, Oracle is making a significant strategic commitment to Java and is integrating it in a large part of its product offering.

Oracle8*i*'s Java offering includes:

- Oracle JServer Option--this is the Java Virtual Machine (Java VM), which runs within the Oracle8*i* database server address space
- Oracle JServer Accelerator Option--a native code compiler that speeds up the execution of Java code by eliminating interpreter overhead (scheduled for Oracle8*i*, release 8.1.6)
- Programmatic interfaces--JDBC drivers and SQLJ
- Utilities and Development Tools--some are separate products

Each of these is summarized separately below.

For more information about Oracle's Java support, refer to the following books.

- *Oracle8i Java Developer's Guide*
- *Oracle8i Java Stored Procedures Developer's Guide*
- *Oracle8i Enterprise JavaBeans and CORBA Developer's Guide*
- *Oracle8i JDBC Developer's Guide and Reference*
- *Oracle8i SQLJ Developer's Guide and Reference*

Oracle8*i*'s Java Virtual Machine

Oracle's Java VM is a complete JDK 1.1.6-compliant Java execution environment. The Java VM runs in the same process space and address space as the database server, sharing its memory heaps and directly accessing its relational data. This design optimizes memory use, increases throughput, and delivers an open, highly available, secure, and manageable Java server.

The Java VM provides a runtime environment for Java objects. It fully supports Java data structures, method dispatch, exception handling, language-level threads, and all the core Java class libraries. It embeds the standard Java namespace in database server schemas allowing Java programs access to Java objects stored in Oracle databases and application servers across the enterprise.

Components of the Java VM include a library manager, bytecode compiler, interpreter, class loader, verifier, native compiler (separate option), server-side JDBC driver, and server-side SQLJ translator. Only the native compiler (JServer Accelerator), JDBC driver, and SQLJ translator components are discussed in this document.

A complete description of Oracle's Java Virtual Machine can be found in *Oracle8i Java Developer's Guide* .

JServer Accelerator

JServer Accelerator is a native code compiler that speeds up the execution of Java code by eliminating interpreter overhead. It translates standard Java binaries into C programs that are processed by a platform-dependent C compiler into native libraries that the Oracle Java VM can load dynamically. Unlike just-in-time (JIT) compilers, JServer Accelerator is portable to all OS and hardware platforms. To speed up applications, the Java VM is supplied with natively compiled versions of the core Java class libraries, ORB, and JDBC.

Note: The JServer Accelerator is not available in release 8.1.5. It is planned for Oracle8*i*, release 8.1.6. However, Oracle JServer enjoys the performance advantage of having all of the JDK libraries shipped with release 8.1.5 compiled with an internal version of JServer Accelerator.

Programmatic Interfaces

Oracle offers two programmatic interfaces (Application Programmer Interfaces, or APIs) for Java Developers: JDBC (Java Database Connectivity) and SQLJ (embedded SQL in Java).

JDBC Drivers

JDBC is a standard Java interface for connecting to relational databases from Java. The JDBC standard was defined by Sun Microsystems, allowing individual providers to implement and extend the standard with their own JDBC drivers. JDBC is based on the X/Open SQL Call Level Interface, and complies with the SQL92 Entry Level standard. In addition to the standard JDBC API, Oracle drivers have extensions to properties, types, and performance.

Oracle offers three versions of its JDBC driver.

Server JDBC Driver Using low-level entry points, a specially tuned JDBC driver runs directly inside the database server. As a result, it provides the fastest access to Oracle data from Java stored procedures. The Server JDBC Driver complies fully with the Sun JDBC specification. Tightly integrated with the database server, it supports Oracle-specific datatypes, NLS character sets, and stored procedures. Also, the client-side and server-side JDBC APIs are the same, allowing for flexibility in where the application is deployed.

OCI JDBC Driver This driver is meant for developers of client/server Java applications and Java-based middle tiers. It converts JDBC invocations into OCI calls, which are sent via Net8 to the Oracle database server.

The OCI JDBC Driver supports various configurations including stateful, multi-tier configurations in which browser-based applets communicate with middle-tier servlets. However, it is not downloadable and requires the installation of OCI and Net8 libraries on each client machine or on the middle-tier Java application server.

Thin JDBC Driver This driver is meant for developers of Java applications and applets. It establishes a direct connection to Oracle over Java sockets and implements a lightweight TCP/IP version of Net8 protocol. So, unlike the OCI JDBC Driver, it works only with TCP/IP-based networks.

Written in Java, the Thin JDBC Driver is about 150 K (300 K uncompressed), so it can be downloaded with Java applets into a Web browser or Network Computer. No preinstallation of software on the client is required.

SQLJ Translator

SQLJ enables SQL statements to be embedded in Java programs. SQLJ code is much more concise and easier to write than JDBC and features static analysis and type checking.

The SQLJ translator, itself a Java program, takes as input a file of Java source code in which SQLJ clauses are embedded. Then, it translates the SQLJ clauses into Java class definitions that implement the specified SQL statements. The Java type system ensures that objects of those classes are called with the correct arguments. Oracle provides both a server-side and client-side SQLJ translator.

The server-side SQLJ translator is a highly optimized SQLJ translator that runs directly inside the database server, where it provides runtime access to Oracle data via the Server JDBC Driver. SQLJ forms can include queries, DML, DDL, transaction control statements, and calls to stored procedures. The client-side and server-side SQLJ APIs are the same, allowing for flexibility in where the application is deployed.

Development Models

Java VM supports a variety of development models, including:

- Java stored procedures--to support traditional database programmers and SQL-oriented clients.
- Enterprise JavaBeans (EJB)--Oracle8*i* Java VM provides a transaction server platform for distributed EJBs.
- CORBA Services--Oracle8*i* also allows distributed systems developers to implement CORBA services in Java. It provides a number of standard CORBA and EJB services.

These are discussed in more detail in the following sections,

Java Stored Procedures

Stored procedures are Java procedures or methods published to SQL and stored in an Oracle database for general use. With stored procedures, you can implement business logic at the server level, thereby improving application performance, scalability, and security.

Except for GUI (Graphical User Interface) methods, any Java method can run in the database server as a stored procedure. The runtime contexts are:

- Functions and procedures
- Database triggers
- Object-relational methods

To publish Java methods, call specifications (*call specs* for short) are written, which map Java method names, parameter types, and return types to their SQL counterparts. A call spec is *not* a wrapper. Without adding a layer of execution, a call spec simply publishes the existence of a Java method. So, when the method is called (through its call spec), the runtime system dispatches the call with minimal overhead. And, because the stored procedure is executing in the database, there is efficient access to SQL data.

Anything that can be written with PL/SQL can now be written with Java and run in the database server in the same contexts as PL/SQL. The database server allows a high degree of interoperability between Java and PL/SQL. Java applications can call PL/SQL stored procedures using an embedded JDBC driver. Conversely, PL/SQL applications can call Java stored procedures directly.

Enterprise JavaBeans

Enterprise JavaBeans (EJB) is a component model introduced by JavaSoft that enables developers to create custom component applications. These applications consist of server-side beans developed by yourself or by third parties. The beans provide the business logic in EJB applications.

The EJB model is flexible in that you can use different components from different vendors. For example, you can use configuration and management software from one company, bean containers from a second company, and business logic beans from a third company that specializes in providing beans for just that purpose.

CORBA

CORBA is a standards-based distributed component model proposed by the Object Management Group (OMG). It supports a development environment for building,

deploying, and managing distributed object applications that are interoperable across platforms.

CORBA objects communicate using OMGs Internet Inter-ORB Protocol (IIOP), the standard for communication between and among distributed objects running on the Internet, intranets, and in enterprise computing environments. Oracle8*i* integrates a Java-based CORBA 2.0 compliant Object Request Broker (ORB) that provides users with the ability to call into and out of the database server using CORBA's IIOP. CORBA components written in different languages, running on different platforms, can transparently communicate and interoperate.

Utilities and Development Tools

Oracle provides utilities and tools for the development and deployment of Java applications. Some of these are discussed here.

JPublisher

JPublisher is a utility, written entirely in Java, that translates user-defined types to Java wrapper classes. It is similar to the Object Type Translator (OTT) utility used in C/C++ environments. Developers are required to have Java classes that correspond to user-defined types for developing:

- SQLJ applications that use object types, collection types (VARRAY and nested table types), or REF types
- JDBC applications that use object types, collection types, or REF types

JPublisher helps create Java language applications that use user-defined types in the Oracle server. For a Java-language application to access object data, it is necessary to represent the data in a host language format. JPublisher helps do this by creating the mapping between object types and Java classes, and between object attribute types and their corresponding Java types.

Using JPublisher to automatically generate Java wrapper classes for user-defined types is less time consuming and error prone than manually coding them.

Note: Portions of JPublisher are still considered beta, and a beta version of the JPublisher User's Guide will be made available. The full functionality of this utility is planned to be available in Oracle8*i*, release 8.1.6.

JDeveloper

With the release of Oracle8*i*, Oracle offers a new version of the JDeveloper product. JDeveloper 2.0 provides complete support for building and deploying Java applications on Oracle8*i*. It provides a standard GUI based Java development environment that is well integrated with Oracle's Application Server and Database.

The following are new features included in JDeveloper 2.0:

- Advanced Enterprise JavaBeans support
- Java database programming with Oracle8*i*
- Servlet support for transactional Web content
- Revamped interface for true multi-tier application development

JDeveloper runs on Windows NT. General availability will follow the production Oracle8*i* release. For more information on JDeveloper 2.0, refer to the product specific documentation.

loadjava and dropjava

Provided with Java VM, the command-line utilities loadjava and dropjava manage Java library units. Java stored procedures can be written, compiled, and partially tested and debugged on the client side in JDeveloper, or other popular Java IDE. Then, the command-line utility loadjava can be used to upload the resulting Java sources, binaries, and resources into the database server. The companion utility dropjava can be used to drop given Java sources, binaries, and resources from a schema.

Data Warehousing and Very Large Data Bases (VLDB)

Oracle8*i* offers many new features and enhancements for the data warehousing and VLDB environment. The performance of queries is dramatically increased by materialized views and summary management, which allow frequently requested summaries to be stored and maintained. Parallelism is improved with more operations offered in parallel. The new database resource management facility, discussed in "Database Resource Management" on page 2-48, provides an effective way of managing CPU usage.

Moving data from one database to another is made simpler and faster with transportable tablespaces. New data mining and relational online analytical processing (ROLAP) functions are introduced. If you are using the partitioning option, the "Partitioning Enhancements" section on page 2-45 discusses two new

methods of partitioning, providing more partitioning options for the data warehouse environment.

Only a few of the many Oracle8*i* changes relating to data warehousing and the VLDB environment have been mentioned above. Read on for descriptions of these and allied new features and enhancements introduced in Oracle8*i*.

Summary Management Using Materialized Views

A *materialized view* is a stored summary containing precomputed results. Materialized views allow for significantly faster data warehouse query processing. With the Oracle8*i* summary management feature, the Oracle database server automatically rewrites queries to use the summary data, rather than retrieving data from detail tables by doing expensive joins and aggregate operations. This *query rewrite* facility is totally transparent to the application, which is not aware of the existence of the materialized view.

The DBA's first step in creating materialized views is to define *dimensions*. These represent the hierarchies that are present in the real world; for instance, multiple months make up a quarter, multiple districts make up a region, etc. The CREATE MATERIALIZED VIEW statement is used to create a materialized view. This statement includes a subquery, typically a join or a data aggregation (GROUP BY), the results of which comprise the materialized view.

A materialized view is maintained by a refresh process. The refresh process can be done automatically when a commit is done on a detail table, or it can be controlled manually by the DBA. A refresh is specified as *complete* or *incremental*. A complete refresh truncates existing data, then repopulates the summary with new data from the detail tables. An incremental refresh updates only changed data.

To create a materialized view in a users own schema, the user must have the CREATE SNAPSHOT or CREATE MATERIALIZED VIEW, CREATE TABLE, CREATE INDEX, and CREATE VIEW system privileges. To create a materialized view in another user's schema, a user must have the CREATE ANY SNAPSHOT or CREATE ANY MATERIALIZED VIEW system privilege.

To enable a materialized view for query rewrite:

- If all the master tables in the materialized view are in a user's schema, that user must have the QUERY REWRITE privilege.
- If any of the master tables are in another schema, a user must have the GLOBAL QUERY REWRITE privilege.

- If the materialized view is in another user's schema, both users must have the appropriate QUERY REWRITE privilege described in the preceding two items.

The schema that contains the materialized view must have sufficient quota in the target tablespace to store the materialized view's base table and index or have the UNLIMITED TABLESPACE system privilege.

To create and refresh a materialized view, both the creator and materialized view owner must be able to issue the defining query of the materialized view. This capability depends directly on the database link that the materialized view's defining query uses.

For ease in managing materialized views, Oracle provides a collection of materialized view analysis and advisory functions that are callable from any PL/SQL program. These are packaged in the DBMS_OLAP package, which provides recommendations and supporting data such as the following:

- Summary usage--such as number of times a query rewrite was done to use a summary, the space used by a summary, and a cost-benefit ratio for each summary
- Summary recommendations--such as creation, retention, and dropping of summaries

For more information on summary management, see *Oracle8i Concepts*, *Oracle8i SQL Reference*, and *Oracle8i Supplied Packages Reference*.

Transportable Tablespaces

This feature allows a user to move a subset of an Oracle database into another Oracle database. It is a lot like unplugging the subset from the original database and plugging it into another one. It is also possible to clone a tablespace in one database and plug it into another, thereby copying the tablespace between databases.

Moving data using the transportable tablespace feature can be orders of magnitude faster than either export/import or unload/load of the same data, because it involves only copying data files and integrating the tablespace metadata. When transporting tablespaces, index data can also moved, eliminating the need to rebuild indexes after importing or loading the table data.

In the data warehouse environment, where huge amounts of data flow from the initial OLTP databases into the enterprise data warehouse and on to data marts, this feature presents an opportunity for a faster and more innovative means of data movement. And, its use can be extended to the archiving of data or publishing data to other databases.

For more information, see *Oracle8i Concepts* and the *Oracle8i Administrator's Guide*.

Direct Path Load API

The direct path load API is a set of OCI interfaces that provides an application access to the direct path load engine in the Oracle server. Previously, the only client of the direct path load engine has been Oracle's SQL*Loader utility. Now, it is possible for ISVs to develop applications which can take advantage of the performance benefits provided by the direct load engine, including parallel load. There is no longer a need to write SQL*Loader control and data files in order to access the direct path load engine.

For more information, see *Oracle8i Utilities* and the *Oracle Call Interface Programmer's Guide*.

ROLAP Enhancement: Extended Aggregate Operations

The GROUP BY clause of SQL queries can now specify additional OLAP aggregate operations. These new operations are called ROLLUP and CUBE, and are effective for computing subtotals and cross-tabulations across multiple dimensions.

While these same operations could be done using previous SQL syntax, the SQL was difficult to generate, and the execution was inefficient because the optimizer had no hints about what the user wanted to accomplish. These new operations allow client reporting and analysis tools to perform aggregate operations with less effort and greater efficiently. They are useful for creating materialized views.

For more information, see *Oracle8i Concepts* and *Oracle8i SQL Reference*.

Data Mining: SAMPLE Function

The FROM clause in a query has been modified with a SAMPLE clause to allow the user to specify that the results are to be based on a random (row or block) sample of a whole table. This feature can be a useful for data mining tools, and in a data warehousing environment can be used to avoid full table scans.

For more information, see *Oracle8i Concepts* and *Oracle8i SQL Reference*.

Function-Based Indexes

Indexes can now be created on functions and expressions that involve one or more columns in the table being indexed. A *function-based index* precomputes the value of the function or expression and stores it in the index. In a data warehouse, query

performance is dramatically improved when a computationally intensive expression is used to build the index, rather than using that expression in the WHERE clause of a query.

The function used for building the index can be:

- An arithmetic expression
- An expression that contains a PL/SQL or Java function, package function, C callout, or SQL function
- A method inside an object, if building an index on that object

For more information, see *Oracle8i Concepts*.

Descending Indexes

The DESC keyword on the CREATE INDEX statement is no longer ignored. It specifies that the index should be created in descending order. Indexes on character data are created in descending order of the character values in the database character set. Neither this, nor the ASC keyword, may be specified for a domain index. DESC cannot be specified for a bitmapped index.

For more information, see *Oracle8i SQL Reference*.

Enhancements for Bitmap Indexes

Several changes enhance query performance when bitmap indexes are used. One of these changes lifts a restriction whereby the use of the ALTER TABLE statement could invalidate a bitmap index.

A new feature introduces two new ALTER TABLE statement options:

- MINIMIZE RECORDS PER BLOCK
- NOMINIMIZE RECORDS PER BLOCK

These options allow for tuning of the rowid-to-bitmap mapping. The MINIMIZE option can be used to optimize bitmap indexes for a query-only environment by requesting the most efficient possible mapping of bits to rowids. The NOMINIMIZE option disables this optimization.

For more information, see *Oracle8i SQL Reference* and *Oracle8i Tuning*.

Parallelization of Aggregate Distinct Queries

The parallelization of aggregate distinct queries is now possible even if they do not contain a GROUP BY clause. This is done through new hints for specifying distribution methods.

For more information, see *Oracle8i Tuning*.

Sort Improvements

The sort routines used throughout the Oracle database server have been rewritten to eliminate performance problems. Sort now performs more predictably and consistently, given the same input, system configuration, and parameter settings. Memory is utilized more efficiently, enabling more rapid sorts when additional memory is specified. The following table shows initialization parameters that have been dropped or added as a result of this rewrite:

Dropped Initialization Parameters	Added Initialization Parameter
SORT_DIRECT_WRITES	SORT_MULTIBLOCK_READ_COUNT
SORT_WRITE_BUFFERS	
SORT_WRITE_BUFFER_SIZE	
SORT_READ_FAC	
SORT_SPACEMAP_SIZE	

For more information, see *Oracle8i Tuning*.

Single-Table Hash Cluster

Using the new SINGLE TABLE option when creating a hash cluster specifies the creation of a hash cluster with only one table. The benefit is faster access to the rows in the table.

For more information, see *Oracle8i Concepts*, or for details about the SINGLE TABLE HASHKEYS clause of the CREATE CLUSTER statement, see *Oracle8i SQL Reference*.

Data Warehouse Statistics Gathering

To support the speed requirements of queries running in a data warehouse environment, it is critical that the optimizer selects the best plan of execution out of all those that are available for consideration. The optimizer selects the best plan

based on cost, as determined by the statistics which have been gathered on the data. Therefore, it is vitally important that the optimizer has good and recent statistics on the data. Oracle8*i* introduces new options for the gathering and manipulation of statistics.

A new package, DBMS_STATS, is provided for:

- External manipulation of statistics
- Gathering statistics utilizing parallel execution
- Copying of statistics between databases

The copying of statistics between databases is useful for tuning studies. Statistics can be copied from the production database into a test database. DBMS_STATS can also be used to back up statistics. The backup may be restored later, if necessary, or used to study the change in data characteristics over time.

Additionally, the syntax of the CREATE and ALTER INDEX statements has been changed to include a COMPUTE STATISTICS clause that allows statistics to be gathered incidentally, and at minimum cost, while the index is being built.

For more information, see *Oracle8i Tuning*, *Oracle8i Concepts*, *Oracle8i SQL Reference*, and *Oracle8i Supplied Packages Reference*.

Automated Parallel Query Tuning

This enhancement simplifies tuning of parallel query (PQ) execution and provides better system utilization. PQ users can simply specify that a query or table is to be parallelized, and the system will determine the optimal single user setting, then reduce it based on the system utilization at the time the query starts execution. There are four parts to this enhancement.

- Default settings are improved for degree and instances.
- There are improved defaults for initialization parameters when the user intends to use parallel query.
- A multi-user adaptive degree algorithm is used to select an optimal degree of parallelism based on system load.
- A load balancing algorithm evenly distributes the load across nodes in an inter-node parallel query environment.

For more information, see *Oracle8i Parallel Server Concepts and Administration* and *Oracle8i Tuning*.

Statistics Generation for Long-Running Operations

In large databases, operations such as complex queries, index builds, and backup operations can take a long time to complete. The dynamic V$SESSIONS_LONGOPS view provides a way for users and DBAs to monitor the progress of such operations and to estimate their completion times. In Oracle8*i*, this view is enhanced by more internal server components collecting a greater number of statistics. Additionally, the package DBMS_APPLICATION_INFO contains a procedure SET_SESSION_LONGOPS to populate the view from an application.

For more information, see *Oracle8i Reference* and *Oracle8i Supplied Packages Reference*.

New Constraint Functionality

The following new features are implemented through syntax changes to the ALTER TABLE and CREATE TABLE statements. For additional information on either of these enhancements, see the *Oracle8i Administrator's Guide* and *Oracle8i SQL Reference*.

Constraint Modification

It is now possible to modify the attributes of an existing constraint through the ALTER TABLE...MODIFY CONSTRAINT clause. This enhancement also introduces a new constraint property, RELY / NORELY. This flag can be used by the DBA to indicate validity of data without enabling or validating a constraint. It is used by the query rewrite function of materialized views.

DISABLE VALIDATE Constraint State

Data warehouses may benefit from the DISABLE VALIDATE constraint state. This state saves space because it requires no index on a unique or primary key, yet it guarantees the validity of all existing data in the table. This is at the expense of disallowing DML and index lookups, but those may be less important considerations in some situations.

The DISABLE VALIDATE constraint state also enables efficient loading of data from a nonpartitioned table into a partitioned table using the EXCHANGE PARTITION clause of the ALTER TABLE statement.

To enable this feature, DISABLE / ENABLE and VALIDATE / NOVALIDATE have been made independent properties of constraints, where all four combinations are now allowed.

Index-Organized Tables

There are numerous enhancements for index-organized tables intended to make them usable for very large databases and mission-critical applications. These enhancements can be broadly classified into two areas:

1. Provide full-table functionality for index-organized tables.
2. Provide the ability to do key compression.

General Enhancements

Some of the key enhancements include:

- LOB column support provides the ability to store large objects, such as text documents and images required by various data cartridges and Web-based applications.
- Secondary index support allows for efficient access using non-primary key columns.
- Index-organized tables (IOTs) can be rebuilt with the new MOVE clause for ALTER TABLE, which also supports an ONLINE keyword. The ONLINE keyword may only be used for IOTs and allows DML operations on the IOT while the primary key index is being built.
- CREATE TABLE... AS SELECT enables parallel loading of an index-organized table.

For more information, see *Oracle8i Concepts*.

Logical ROWIDs

ROWIDs provide the fastest possible access to a given row in a given table. They contain the physical address of a row down to the specific block and allow retrieval of the row in a single block access. To expand this functionality to index-organized tables (IOTS), whose rows do not have permanent physical addresses, Oracle8*i* introduces primary key-based logical identifiers called logical ROWIDs.

Rows in index-organized tables do not have permanent physical addresses--they are stored in the index leaves and can move within the block or to a different block as a result of insertions--therefore their row identifiers cannot be based on physical addresses. Instead, Oracle provides index-organized tables with logical row identifiers, called logical ROWIDs, that are based on the table's primary key. Oracle uses these logical ROWIDs for the construction of secondary indexes on index-organized tables.

Also introduced is the UROWID datatype, which makes logical ROWIDs usable by applications in the same sense that ROWIDs are currently used (e.g., selecting ROWIDs for later update or as part of a cursor). UROWID can be used to store ROWIDs from other databases accessed via gateways, and it can be used to reference physical ROWIDs. PL/SQL fully supports the UROWID datatype.

For more information, see *Oracle8i Concepts*.

Related Features

Other features are also useful in the data warehouse environment, but are discussed elsewhere in this document.

- "Partitioning Enhancements" on page 2-45
- "Database Resource Management" on page 2-48
- "Oracle Parallel Server" on page 2-61
- "Optimizer Plan Stability" on page 2-37
- "User-Defined Operators and Extensible Indexing" on page 2-31

Database Features

With Oracle8*i*, the Oracle database server provides a new extensibility framework, making it easier for ISVs and other developers to extend the database to meet their needs. Many object relational enhancements have been incorporated into this release, and this functionality has been extended to all Oracle8*i* product configurations.

Online transaction processing (OLTP) systems are usually concerned with entering and retrieving mission-critical data from day-to-day operations. Availability, reliability, and performance are extremely important for these systems. Oracle8*i* delivers significant enhancements in these areas, along with improvements in scalability, serviceability, and security.

Advance Queuing has been significantly enhanced, continuing to develop the publish/subscribe messaging paradigm. National Language Support now has a new book to help users better learn how to use its new and already existing functionality.

These and other new database features are discussed in this section.

Extensibility Framework

The extensibility framework in Oracle8*i* extends the Oracle database server to support specialized datatypes and to provide core services for external use. Developers are able to provide their own implementations of database server services and register them with the server. The framework is safe and protects the integrity of Oracle8*i*.

ISVs and other developers can exploit this extensibility framework to more easily build user-defined software components, also referred to as data cartridges, that are easier to use, and that integrate seamlessly with each other and the Oracle database server. Multimedia users especially can benefit from the greater flexibility and fuctionality provided by this framework as it enables independent software components to interoperate in a seamless manner. For example, a query can now be based on both image and spatial data.

This section summarizes the features and enhancements that comprise the Oracle8*i* extensibility framework. For more information on the functionality described below, see the *Oracle8i Data Cartridge Developer's Guide* and *Oracle8i Concepts*.

User-Defined Operators and Extensible Indexing

Increasingly, databases are being used to store more varied types of data, such as spatial, audio, or video data. This brings a need for indexing complex datatypes and for specialized indexing techniques. Oracle8*i* provides an interface that enables developers to define domain-specific operators and indexing schemes and integrate them into the Oracle database server.

Oracle provides a set of built-in operators, for use in SQL statements, which include arithmetic operators (+, -, *, /), comparison operators (=, >, <), logical operators (NOT, AND, OR), and set operators (UNION). These operators take as input one or more arguments (operands) and return a result. The extensibility framework of Oracle8*i* allows developers to define new operators. Their implementation is provided by the user, but the Oracle database server allows these user-defined operators to be used in SQL statements in the same manner as any of the predefined operators provided by Oracle.

The framework to develop new index types is based on the concept of cooperative indexing, where an application and the Oracle database server cooperate to build and maintain indexes for complex datatypes. The application software is responsible for defining the index structure, maintaining the index content during load and update operations, and searching the index during query processing. The index structure itself can be stored either in the Oracle database as tables or

externally as files. Indexes created using these new index types are referred to as *domain indexes*.

For specific information about the extensions to SQL that provide this support, see *Oracle8i SQL Reference*.

Extensible Optimizer

The optimizer functionality has been extended to allow authors of user-defined functions and domain indexes to create statistics collection, selectivity, and cost functions that will be used by the optimizer in choosing a query plan.

Statistics Collection Functions The ANALYZE command can now make a call to a user-specified statistics collection function whenever a domain index is analyzed. User-defined statistics collection functions can also be defined for individual columns of a table and for user-defined datatypes.

Selectivity Functions Users can specify user-defined selectivity functions for predicates containing user-defined operators, standalone functions, or type methods.

Cost Functions Users can define costs for domain indexes and for user-defined standalone functions, package functions, and type methods. The user-defined costs can be in the form of default costs that the optimizer simply looks up, or they can be computed by functions called by the optimizer.

Cartridge Services

In addition to the above, other basic services have been packaged for use by user-defined components. These, and the other extensibility and object enhancements in Oracle8*i*, have allowed the uniform integration of Oracle's former data cartridge options (ConText, Image, Visual, Time Series, Spatial) into the Oracle database server.

The services include:

- Memory management
- Context management
- Parameter management
- String and numbers manipulation
- File I/O

- Internationalization
- Error reporting
- Thread management

Additionally, the Oracle Software Packager, discussed in "Oracle Software Packager and Oracle Universal Installer" on page 2-58, allows cartridges to be packaged and installed into Oracle8*i*.

Object Relational Enhancements

The object relational paradigm introduced in Oracle8 gains new functionality in Oracle8*i*. Additionally, object relational technology is now available in all configurations of the Oracle database server. Enhancements target more flexible storage and retrieval of objects, and there is greater transparency between object views and object tables. Specifics of these changes are described below.

For more information on object relational enhancements, see the *Oracle8i Concepts*, *Oracle8i Application Developer's Guide - Fundamentals*, and *PL/SQL User's Guide and Reference*. Additional references may be provided below in specific descriptions.

LOB Enhancements

Several LOB-related enhancements are included in this release. Additional references for these include the *Oracle Call Interface Programmer's Guide*, the *Oracle8i Application Developer's Guide - Large Objects (LOBs)*, and *Oracle8i SQL Reference*.

Temporary LOBs Oracle8 provided support for permanently storing large unstructured data by means of large object (LOB) datatypes. But many applications have a need for temporary LOBs that act like local variables. These LOBs do not exist permanently in the database, but exist mainly for the purpose of performing transformations on LOB data.

Oracle8*i* supports the creation, freeing, access, and update of temporary LOBs. Their default lifetime is a session, but they may be explicitly freed sooner by the application. They are ideal as transient workspaces for data manipulation, and because no logging is done or redo records generated, they have better performance than persistent LOBs.

Other Enhancements ■A new conversion function, TO_LOB, can be used to convert a LONG to a LOB.

> **Note:** LONG column support will be discontinued in a future Oracle release. Users are advised to migrate LONG data into LOB columns.

- Usable LOB chunk size is exposed to users via an OCI interface and in DBMS_LOB.
- Open / Close / IsOpen APIs (OCI or DBMS_LOB) for internal LOBs have been added.
- Variable-width character sets are supported for CLOBs and NCLOBs.
- A DBMS_LOB and OCI API are provided to allow users to append data to the end of the LOB value.

User-Defined Object Identifiers

The system-generated object identifiers (OIDs) provide globally unique identifiers that are important for many applications, especially distributed processing and replication. However, they carry an overhead of increased storage requirements, and increased loading time. For applications that do not require the advantages of system-generated OIDs and which must save on overhead, Oracle8*i* allows users to specify their own identifier, such as a primary key, as an OID.

A related enhancement is referential integrity constraint support on REF columns.

Nested Table Enhancements

These enhancements target making the handling and storing of large nested tables more efficient. Additional references for nested table enhancements are the *Oracle Call Interface Programmer's Guide*, *Oracle8i SQL Reference*, and *Oracle8i Supplied Packages Reference*.

- DDL is enhanced to allow specifying that the rows of a nested table be stored in an index-organized table. Constraints may be specified on the table. Storing the nested table data in an index-organized table provides clustering of the nested table data and is an efficient storage organization, especially when the data is always retrieved based on the primary key.
- The user may specify at the time of table creation that a collection locator is to be returned when a nested table column or attribute is fetched. An OCI call (OCICollIsLocator()) and a UTL _COLL package (IS_LOCATOR function) are provided to indicate whether a collection is locator based or not. An application or PL / SQL program can now check whether a nested table attribute or variable

is locator based, before performing certain collection operations that could cause a large nested table value to be materialized in memory.

Varray Enhancements

Oracle8*i* provides for more efficient handling and storage of large varrays. Varray storage can now be specified. If specified, it is stored out of line in a LOB and the user can specify the storage characteristics of the LOB.

Collection Unnesting

The notion of collection datatypes (nested tables and varrays) was introduced in Oracle8. Users are allowed to create columns of collection types, and functions and procedures with parameters of collection types. When it is necessary to view these collection types in a flat (relational) form, Oracle provides a way of unnesting them.

This unnesting is implemented using the TABLE() operator found in the *table_collection_expression* of the SQL statement. When introduced in release 8.0, the TABLE() expression took an argument of a set of rows and allowed that set to be treated like a table. Oracle8*i* extends this functionality so that a collection column of a table in a query may be used like an independent table.

Note: TABLE() subsumes the THE(subquery) expression, thereby deprecating its usage.

Enhancement of Object Views

Object views are an extension to the basic relational view mechanism to provide an object abstraction over existing relational and object data. They allow for the retrieval, update, insertion, and deletion of relational data as if such data were stored as object types. In this release, object views are enhanced to provide more features than are available in object tables, such as, updatability of nested tables and creation of new objects in the OCI object cache. The performance of object views is improved to be comparable with that of object or relational tables.

Triggers on Nested Table View Columns

Triggers are PL/SQL procedures which are fired when various events occur in the system, such as updates to a row. In release 8.0, the trigger mechanism was defined over rows of tables or as an INSTEAD OF mechanism over views. Oracle8*i* expands the application of triggers to allow them to be defined over nested table view

columns. As a result, it is now possible to update an element of a collection synthesized using the CAST...MULTISET operation.

Object Cache Enhancements

The following enhancements are options that can be set for the object cache:

- The ability to detect whether an object in the cache has been changed in the server by another committed transaction
- The ability to set all of the attributes of an object to NOT NULL at the time of creation
- To prevent a user from blocking while trying to lock an object, NOWAIT can be specified when trying to lock an object
- Object cache extensibility -- allow cache clients to use their own object representations instead of C structs

For more information, also see the *Oracle Call Interface Programmer's Guide*.

ANALYZE Command, Dangling REFs

The ANALYZE command can now set dangling REFs to NULL. In this case, these REFs no longer raise an error condition and terminate execution.

For more information, also see *Oracle8i SQL Reference*.

Subqueries in VALUES Clause of INSERT Statement

In release 8.0, if a relational table or an object table contained REF type columns, there was no simple way of inserting a relational row or a row object into the table. First NULLs were required to be inserted into the columns and then an update statement was issued to set their values. This inefficiency is resolved in Oracle8*i*, by allowing a value subquery expression in the INSERT...VALUES list.

A value subquery expression may also be used as an argument to a function or a type constructor.

SQL*Loader Support for Objects, Collections, and LOBs

The SQL*Loader, used to move data from external files into tables in an Oracle database, has been extended to support the loading of objects, collections, and LOBs. For a summary of SQL*Loader enhancements, see "SQL*Loader Enhancements" on page 2-56.

For more information, see *Oracle8i Utilities*.

Other Object Relational Enhancements

Partitioning of Tables Containing Objects The partitioning of tables with columns containing objects, collections (varrays and nested tables), and REFs is provided in Oracle8*i*. The partitioning of object tables is also supported.

Parallel Query Support Oracle8*i* provides parallel query support on object tables.

Optimizer Plan Stability

After carefully tuning an application, a user may wish to ensure that the optimizer generates the same execution plan whenever the same SQL statements are executed. Oracle8*i* provides a method for stabilizing execution plans, regardless of changes to the system configuration, parameters, statistics, or even changes to the optimizer itself. This also benefits high-end OLTP sites by having SQL execute without having to invoke the cost-based optimizer at each SQL invocation.

This feature is especially useful for ISVs, who can distribute the execution plans of the SQL in their package applications so that all installed sites will be executing the same execution plans.

Through a new SQL statement, CREATE OUTLINE, the user is able to create a *stored outline* containing a set of attributes used by the optimizer to create an execution plan. Stored outlines can also be created automatically by setting the system parameter CREATE_STORED_OUTLINES to TRUE. The system parameter USE_STORED_OUTLINES can be set to TRUE, FALSE, or to a category name to indicate whether to make use of existing stored outlines for queries that are being executed. The OUTLN_PKG package provides procedures used for managing stored outlines.

For more information, see *Oracle8i SQL Reference*, *Oracle8i Tuning*, and *Oracle8i Supplied Packages Reference*.

Advanced Queuing

Advanced Queuing (AQ) was introduced in release 8.0 of Oracle8 Enterprise Edition. It integrated a message queuing system with the Oracle database through a set of queue tables and queue functions, and allowed for either synchronous or asynchronous communication between programs. Oracle8*i* provides enhancements to the functionality of Advanced Queuing. These are discussed in this section.

For more information, see the *Oracle8i Application Developer's Guide - Advanced Queuing*.

AQ-Based Publish/Subscribe

Additional features are introduced in Oracle8*i* Advanced Queuing to support a publish/subscribe messaging paradigm. In this model, database queues serve as a repository for the messages, rules, and control information. There is a decoupling of addressing between message senders and message recipients. It is the recipients who specify which messages they wish to receive. Sending applications anonymously publish messages and receiving applications independently and autonomously subscribe to such messages.

The components of this publish/subscribe model are:

- Publishers

  Publishers are entities that publish information (as messages) to queues. They do not know or care about the interest of other applications in the messages; they only know that they are to publish the information. There are two types of publishing: database event publishing where triggers cause messages to be enqueued when certain database or system events occur, and publishing from an application. Database event publishing is new in Oracle8*i* and is discussed separately later.

- Subscribers

  Subscribers are entities that receive information by expressing interest in certain types of messages. They do not care about the origins of the messages. They specify certain filtering rules for messages, allowing them to receive only the messages whose content satisfies the rules.

- Rules Engine

  The rules engine, new in Oracle8*i*, provides rules-based subscriptions by evaluating incoming messages from publishers against a set of rules specified by subscribers. Only if the message satisfies a rule is it delivered to the subscriber. Subscribers can express rules using the full power of SQL. Since AQ messages are normal database objects, any SQL operation that applies to data objects can also be applied to messages.

An application (subscriber/client) may receive messages that are published in one of three ways: explicit dequeuing, dequeuing with listen, or by using OCI functions to register a callback. For more specific information, also see the *Oracle Call Interface Programmer's Guide*.

Database Event Publication

The database event publication feature allows applications to subscribe to database events just as they subscribe to messages from other applications. Trigger syntax is extended to support system and other data events on DATABASE and SCHEMA, Trigger syntax also supports a CALL to a procedure as the trigger body.

Users can enable the publication of the following events.

- DML events (DELETE, INSERT, UPDATE)
- DDL Events (CREATE, ALTER, DROP)
- Database events (SERVERERROR, LOGON, LOGOFF, STARTUP, SHUTDOWN)

The database event publication subsystem tightly integrates with the AQ publish/subscribe engine. For a complete description of triggers for data and system events, see *Oracle8i SQL Reference*.

Enhanced Propagation

For increased scalability, the performance of basic queue operations has been enhanced and changes have been made to underlying data structures. Message propagation scheduling is improved. AQ propagation uses message streaming and eliminates the current two-phase commit mechanism. Additionally, the identity of the sender is now made part of message properties. This meets a proposed standard for messaging services and enables messaging services such as TIBCO's RVCM to distribute messages in near real time to a large number of applications.

Statistics are now maintained on the scheduled propagation, enhancing the administrator's ability to manage and tune propagation. Also, a single view is provided for the administrator that can be used to determine the propagation schedule, its status, and last error.

There is better history management for multi-consumer queues, and the history itself will now contain a new message state and property for distinguishing between a message being propagated and its being processed.

Miscellaneous Enhancements

Other Advanced Queuing enhancements are as follows:

- For multiple consumer queues, a user is allowed the option of enabling or disabling the collection of statistics on the number of messages per recipient.

- System and object level security control are provided. The AQ security model is now a natural extension of the Oracle8*i* security model.
- Performance is improved when running in parallel server mode.
- The creation of nonpersistent queues is allowed.

DBMS_REPAIR Package

Oracle8*i* provides enhanced block corruption repair capability through the new DBMS_REPAIR package. It provides the DBA with a three-stage approach to addressing corruptions.

1. Detect and report corruptions.
2. Make the object usable.
3. Repair corruptions and rebuild lost data.

Depending upon the nature of the repair, data may be lost and logical inconsistencies may be introduced. The DBA must determine whether the potential loss warrants the use of this feature, and to help the DBA, this package also provides a report mechanism that aids in the determination of whether to use this feature or not.

For information on this feature, see the *Oracle8i Administrator's Guide* and *Oracle8i Supplied Packages Reference.*

Redo Log Analysis Using LogMiner

Log files contain a wealth of useful information about the activities and history of an Oracle database, but until Oracle8*i* there has been no easy tool that could tap into this information. LogMiner allows online and archived redo log files to be read, analyzed, and interpreted by the user using SQL. Analysis of the log files with LogMiner can be used for the following:

- It can be used to track specific sets of changes based on transaction, user, table, time, etc. It is possible to determine who modified a database object and what the before and after data was. Understanding and being able to trace database changes back to their source and being able to undo them are valuable security and management tools.
- It can help pinpoint when a logical corruption was introduced into the database. This is important for determining how to initiate recovery for restoring the database to a consistent state prior to corruption.

- This data can also be used as a supplemental source of data for tuning and capacity planning. Performing various forms of historical analysis to determine trends and data address patterns can be done.

No additional collection overhead is incurred to obtain data for LogMiner.

For more information about LogMiner, see *Oracle8i Backup and Recovery Guide*.

Drop Column

A new drop column capability allows the DBA to easily remove unused columns in the database. Previously, it was necessary to use the export/import utilities. New syntax for the ALTER TABLE statement allows a column to be marked as unusable, without freeing up space in the table, or to be dropped from the table with the data deleted.

This feature complies with Transitional SQL92.

For more information, see the *Oracle8i Administrator's Guide* and *Oracle8i SQL Reference*.

Locally Managed Tablespaces

Prior to Oracle8*i*, management of free and used extents within a tablespace relied heavily on data dictionary tables. Now, Oracle introduces a new mechanism for managing space within a tablespace: locally managed tablespaces. All extent information is tracked in the tablespace itself, using bitmaps.

Bitmaps manage space allocation very efficiently, and require no dictionary access to allocate or update extents for the tablespace. The result is better space management, reduced fragmentation, and increased reliability

For more information, see *Oracle8i Concepts* and *Oracle8i SQL Reference*.

Online Index Creation, Rebuild, and Defragmentation

Creating or rebuilding an index on a table improves query performance when completed, but can be a very time-consuming operation. For large tables, it can require several hours of downtime because it forces a lock on the table and prevents concurrent DML. This is called offline index build.

Oracle8*i* supports the online creation or rebuilding of an index that works for partitioned or nonpartitioned B*-tree indexes, including index-organized tables. Since the table is not locked, DML operations and queries can execute on the base table while the index is being built.

New SQL syntax specifies the online index creation or rebuild. Specifically, the ONLINE keyword may be specified on the CREATE INDEX or ALTER INDEX statement.

Alternatively, if it is not desired to rebuild the index because of space or other considerations, the new COALESCE keyword may be specified to defragment the index. In either case, performance will be improved and space recovered.

For more information, see the *Oracle8i Administrator's Guide* and *Oracle8i SQL Reference*.

Nonpartitioned Table Reorganization

A new MOVE clause used in an ALTER TABLE statement provides a means to easily reorganize a nonpartitioned table by allowing the user to move data into a new segment while preserving all views, privileges, etc. defined on the table. The operation is performed offline.

For more information, see the *Oracle8i Administrator's Guide* and *Oracle8i SQL Reference*.

Online Read-Only Tablespaces

Oracle8*i* improves the performance of the operation that places a tablespace in read-only mode. A tablespace in Oracle8*i* can be placed in read-only mode when there are no outstanding transactions in that tablespace alone, unlike previous versions of Oracle where the operation completed only when there were no outstanding transactions in the entire database.

Now, an ALTER TABLESPACE...READ ONLY statement will make the tablespace read only as soon as all transactions on that tablespace have completed. This capability is especially important for the transportable tablespaces feature described in "Transportable Tablespaces" on page 2-23.

For more information, see *Oracle8i Concepts*.

Temporary Tables

A temporary table is a table with session-specific or transaction-specific data. It is empty when the session or transaction begins, and discarded at the end of the session or transaction. Its definition is visible to all sessions but the data is visible to, and can be queried by, only the session that inserts the data into the table. It is created in the user's temporary tablespace. Undo (rollback to savepoint) is supported, but not redo (crash recovery). Temporary tables can be useful for saving

intermediate results that can be joined back into another table. They also might facilitate migration from other databases.

For more information about temporary tables, see *Oracle8i Concepts*.

National Language Support (NLS)

Oracle8*i* National Language Support includes significant new enhancements that make it easier to deploy applications requiring local language support, or that require country-specific formatting.

General Enhancements

These are some general enhancements to NLS.

- Programming interfaces (OCI) provide cartridge developers and application developers access to international information and services.
- Function-based indexes, as described in "Function-Based Indexes" on page 2-24, allow for the creation of indexes based on linguistically sensitive NLS_SORT orders.
- Fixed-width Unicode (UCS2) character support is provided in the following client interfaces: OCI, Pro*C / C++, ODBC.
- Expanded NLS data has been included. New territories have been added for Europe (Belgium, Ireland, Luxembourg), Asia Pacific (Australia, New Zealand, Singapore), Eastern Europe (Kazakhstan, Uzbekistan), and Africa (South Africa).
- Support for the Euro currency symbol is included.

For more information, refer to the new *Oracle8i National Language Support Guide*.

Changing Character Sets with ALTER DATABASE

Oracle8*i* makes it possible to change both the character set the database uses to store data, and the national character set used to store data in columns specifically defined as NCHAR, NCLOB, or NVARCHAR2, *after* the database has been created. The old character set must be a strict subset of the new one.

A user with SYSDBA system privilege, may issue the following command statements:

- ALTER DATABASE CHARACTER SET
- ALTER DATABASE NATIONAL CHARACTER SET

For more informations, see *Oracle8i SQL Reference*.

TRIM Function

This enhancement implements the ANSI standard TRIM function. It combines the functionality of the existing LTRIM and RTRIM functions, allowing the user to trim leading or trailing characters, or both, from a character string.

For more informations, see *Oracle8i SQL Reference*.

External Routines

An *external routine*, previously referred to as an external procedure, is a routine written in another language and stored in a dynamic link library (DLL), or libunit in the case of a Java class method. The routine is registered with the base language, and can be called to perform special-purpose processing.

First introduced in Oracle8, external procedures allowed the writing of C functions as PL/SQL bodies. These C functions are callable from PL/SQL and SQL (via PL/SQL). In Oracle8*i*, a special-purpose interface, the *call specification*, is provided that lets external routines be called from other languages.

While this service is designed for intercommunication between SQL, PL/SQL, C, and Java, it is accessible from any base language that can call these languages. For example, a routine can be written in a language other than Java or C and still be usable by SQL or PL/SQL, provided that is callable by C. Therefore, a C++ routine, would use a C++ extern "C" statement in that routine to make it callable by C.

Below is a brief description of the new call specification, and a discussion of other changes targeting enhancement of external routine performance.

For information about external routines, see *Oracle8i Application Developer's Guide - Fundamentals*.

The Call Specification

Until now, an external routine has been published to Oracle via an AS EXTERNAL clause in a PL/SQL wrapper. This wrapper defines the mapping to, and allows the calling of, external C routines. Oracle8*i* introduces call specifications, which include the AS EXTERNAL wrapper as a subset of a new AS LANGUAGE clause. AS LANGUAGE call specifications allow the publishing of external C routines, as before, but also Java class methods.

Call specifications also allow publishing with the AS EXTERNAL clause, introduced in Oracle8. For new applications, however, the AS LANGUAGE clause should be used.

External Routine DLL Caching

A DLL caching mechanism has been developed to allow reuse of already loaded libraries for subsequent external routine invocation. This avoids costly repetitive operations, thus improving the performance, usability, and scalability of external routines. DLL caching applies to both server-side external routines, invoked through the extproc agent, and to client-side external routines.

External Routine Object Support

Certain object support, previously available to trusted external routines only, is now available to untrusted and distributed routines. Specifically, all external routines now support object types as argument types, and clients are allowed to issue object-related OCI callbacks.

Database Security

For a discussion of the *virtual private database* feature, whose components are fine-grained access control and application context, see "Securing Data in the Oracle Database Server" on page 2-68. This feature secures data in the database by providing security at the row-level, across all applications, by attaching a security policy directly to a table or view.

Partitioning Enhancements

Range partitioning was introduced in Oracle8, release 8.0. Oracle8*i* provides enhancements for range partitioning, but more significant is the expansion of Oracle's repertoire of partitioning methods by introducing two new methods: *hash* and *composite*.

For more information about partitioning and the new features discussed in this section, see *Oracle8i Concepts* and *Oracle8i SQL Reference*.

Range Partitioning Enhancements

Merging Partitions

A new statement, ALTER TABLE...MERGE PARTITIONS, may be used to merge the contents of two adjacent partitions of a table partitioned using the range method. This is the inverse of a SPLIT PARTITION operation.

Updatable Partition Keys

A new clause, ENABLE ROW MOVEMENT, of the ALTER TABLE or CREATE TABLE statement allows rows to be moved between partitions. When row movement is enabled, updates affecting partitioning keys in such a way that a row no longer belongs in its current partition will proceed, resulting in a row migrating to the appropriate partition. The default behavior (DISABLE ROW MOVEMENT) is that row movement is disabled and such updates are disallowed, resulting in an error being returned to the user.

New Partitioning Methods

The new partitioning methods introduced in Oracle8*i* are hash and composite. Along with range partitioning, these present a rich set of partitioning methods that allow the DBA to choose the partitioning method that will offer the best performance depending on the database workload and the application's profile. As with range partitioning, all partitioning techniques are transparent to applications and standard DML statements run against the partitioned tables.

Hash partitioning provides a very simple way to break data into evenly sized containers to be spread across multiple I/O devices, or even multiple machines in a shared-nothing cluster. Query performance is improved by spreading I/O across multiple devices, and the performance of parallel DML may also be improved. Note, however, that hash partitioning is inappropriate for rolling change windows of historical data because the data for any period of time can be spread across multiple partitions.

Composite (range/hash) partitioning provides the manageability and availability benefits of range partitioning with the data distribution advantages of hash partitioning. The user specifies ranges of values for the primary partitions of the table or index, then specifies a number of hash subpartitions. Data skew is unlikely, because the user can always add or drop subpartitions within a partition to maintain even distribution of each container. Rolling change windows of historical data are easily maintained by adding or dropping primary partitions with no effect on subpartitions in other primary partitions.

Local indexes are supported with hash and composite partitioning. All existing SQL query and DML statements work with both new partitioning methods and new or extended DDL statements are provided for maintenance.

Additional Performance Gains

Enhanced partitioning includes many performance enhancements, some of which have already been discussed. Not yet discussed are the following two.

Enhanced Partition Elimination

Partition elimination is the skipping of unnecessary index and data partitions (or subpartitions) in a query. Because Oracle8*i* enhanced partitioning provides the database with greater knowledge of data placement, it can perform more advanced partition elimination. Particular focus is on queries using disjunctive OR or IN predicates.

Partition-Wise Join

In Oracle8, as the first step when two tables are being joined in parallel, each table is split into some number of separate pieces, scanned, then redistributed on the join column. In Oracle8*i*, the redistribution step can be skipped if the data is already partitioned on the join key. This reduces memory and temporary storage requirements and increases overall performance.

Partitioned Table LOB Support

Full support for partitioned tables with LOB columns is now provided. The LOB value can be stored inline in the row or out of line in a separate segment. All LOB types are supported: BLOB, CLOB, NCLOB, BFILE.

Partitioning of Index-Organized Tables

Very large databases, historical databases, and OLTP databases with special availability requirements can benefit from the partitioning of index-organized tables (IOTs). Oracle8*i* delivers this support for partitioning of IOTs and the secondary indexes defined on IOTs. The bulk loading of partitioned IOTs with secondary indexes via SQL*Loader direct path is supported and provides the most efficient way of loading data.

System Management

This section is concerned with the new functionality and tools that greatly enhance your ability to control resource usage, enable better recoverability and availability in the database, and make your database a lot easier to manage.

Database Resource Management

Traditionally, resource management decisions have been left to the operating system, rather than the database server. This results in the following problems.

- There is no way to partition machine resources appropriately among database tasks of varying importance.
- When the number of servers is high, there is excessive overhead due to OS context switching between database servers.
- Scheduling is inefficient--Oracle database servers are descheduled while they hold latches.

The database resource manager feature addresses these issues by giving the database more control over the management of resources. Oracle8*i*'s database resource manager feature provides the DBA with the ability to control and limit the total amount of processing resources available to a given user or set of users.

For example, suppose that a data warehouse is shared by the marketing department and the sales department. Using the database resource manager, a warehouse administrator could specify that the marketing department receives at least 60 percent of the CPU resources of the machines, while the sales department receives 40 percent of the CPU resources. The warehouse administrator could further specify limits on the total number of active sessions, and the degree of parallelism of individual queries for each department.

Specifically, using the database resource manager facility, the DBA can:

- Guarantee certain users a minimum amount of processing resources regardless of the load on the system and the number of users.
- Distribute available processing resources by allocating percentages of CPU time to different users and applications. In a data warehouse, a higher priority may be given to ROLAP applications than to batch jobs.
- Limit the degree of parallelism that a set of users can use.

- Configure an instance to use a particular method of allocating resources. A DBA can dynamically change the method, for example, from a daytime setup to a nighttime setup, without having to shut down and restart the instance.

In addition, a user can select the priority of a session from a given set of priorities that the DBA has assigned to that user.

DBAs use *resource consumer groups*, *resource plans*, *resource allocation methods,* and *resource plan directives* to implement database resource management. There functions are:

resource consumer group	They allow the administrator to group user sessions together based on their requirements for processing resources.
resource plan	It contains resource plan directives that specify the resources that are to be given to each resource consumer group.
resource allocation method	It determines what method (or policy) to use when allocating for any particular resource. Resource allocation methods are used by both plans and consumer groups.
Resource plan directives	There is one of these for each consumer group in the plan. They allow the administrator to assign consumer groups to particular plans and partition resources among consumer groups by specifying parameters for each resource allocation method.

Resource plans, resource consumer groups, and resource plan directives are created using the PL/SQL package DBMS_RESOURCE_MANAGER. Users are assigned to consumer groups using the DBMS_RESOURCE_MANAGER_PRIVS package. The initialization parameter RESOURCE_MANAGER_PLAN specifies which top plan to use for a given instance. The database resource manager loads this top plan as well as all its descendants (subplans, directives, and consumer groups).

For more information, see *Oracle8i Concepts*, the *Oracle8i Administrator's Guide*, and *Oracle8i Supplied Packages Reference*.

Recoverability and Availability in the Database

The Oracle database server's industry leading availability and reliability are further extended in Oracle8*i*. Significant enhancements and new features expand our

customers' ability to provide highly available data center solutions and meet service level objectives.

Unless otherwise noted, information about the new recovery and availability features discussed here can be found in the *Oracle8i Backup and Recovery Guide*.

Multiple Remote Archive Destinations

Oracle8*i* provides the means to allow the background archive process (ARCH), or its foreground equivalent, to archive online redo log files to multiple destinations. Up to five destinations can be specified with the initialization parameter: LOG_ ARCHIVE_DEST_n. A destination can be a local disk-based file or it can be a user-specified standby database that is either local or remote to the primary database.

It has been observed that two archival destinations is not enough, while more than five is excessive and provides no additional benefit.

Multiple Archive Processes

Oracle8*i* allows the user to specify multiple archive processes to be invoked at instance startup. This is intended to alleviate bottlenecks that can occur when the LGWR (log writer) process is able to write to the redo log file faster than ARCH can read from the log file and write to one or more destination archivelogs. It will no longer be necessary for users to attempt manual foreground archival operations in order to recover, as multiple archive processes will automatically handle the situation.

Automated Standby Databases

Automated standby databases provide the means to create and maintain multiple remote copies of a production database for disaster recovery scenarios. This is one of the key technologies available to protect against a disaster to the primary database system or data center. The backup or automated standby database can reside in the same or different data center and take over the processing from the primary production database providing near continuous database availability. Oracle has provided the standby database feature for a number of years, but the technology is significantly extended in Oracle8*i*.

Automatic Archival A standby database is initially created by copying or duplicating the production database. As archived redo logs are generated on the production database, they are applied to the standby database. This allows the standby database to remain synchronized with the production database. In releases prior to

Oracle8*i*, the redo logs were manually transported or copied to the standby database and manually applied.

With the new automated standby database feature, the archived redo logs can be automatically transferred and applied. This eliminates the need for manual procedures to copy and transmit the redo logs and the need for the operator at the backup site to manually specify which logs to apply. The standby database is placed in constant recovery mode by the user, causing it to wait for archived log files from the primary database. When a primary archived log file arrives at the standby site, it is verified and then applied to the standby database.

Read-only Databases Another new capability is the ability to use a standby database as a read-only database. All databases, not just standby databases, benefit from this new feature, but it is especially useful for standby databases. This option can be used to make a standby database available for queries and reporting, even while archive logs are being copied from the primary database site.

Fast-Start Fault Recovery

Transparent application failover (TAF) was introduced in Oracle8 and masks system failures from applications. It is discussed in "Transparent Application Failover (TAF)" on page 3-8. To allow the Oracle database server to quickly recover from system faults and minimize the impact on users, Oracle8*i* introduces fast-start fault recovery. It consists of the following components.

Fast-Start Checkpointing As a means of controlling the time required for instance and crash recovery, this feature allows the DBA to specify an upper limit on the number of I/O operations that Oracle will need to perform during instance recovery. A new dynamic parameter, FAST_START_IO_TARGET is introduced to control Fast-Start checkpointing.

Faster recovery is achieved at the expense of writing out additional (meaning: more than would have been written if this feature were disabled) buffers during normal processing. The fewer the number of dirty buffers in the buffer cache at the time of the failure, the faster the recovery times. This is because recovering data blocks by applying redo records between the most recent checkpoint and the end of the log constitutes the majority of the work and time during recovery. Statistics are collected and are available to the DBA through dynamic performance views to tune the system and make tradeoffs between runtime performance and database availability.

For additional information on this feature, see *Oracle8i Tuning*.

Fast-Start On-Demand Rollback This feature enables dead transactions to be recovered, on demand, one block at a time. This improves the availability of the database for users accessing data that is locked by large dead transactions. In earlier releases, users are blocked until the entire dead transaction is recovered. With this feature, the user immediately recovers only the block under consideration and proceeds, leaving the rest of the dead transaction to be recovered in the background.

Fast-Start Parallel Rollback This Oracle8*i* feature is primarily targeted to solve the problem of recovering parallel transactions. Parallel transactions in release 8.0 exploited parallelism in the roll forward stage using parallel block recovery. But on a failure, the failed transactions were rolled back serially by the background process. This impacted the availability of the part of the database involved in the failed transactions.

Fast-start parallel recovery allows transactions to be recovered in parallel, resulting in significantly increased recovery throughput. Inter-transaction recovery is performed like parallel block recovery, where each slave rolls back a separate transaction. But if one of those transactions is extremely long, then intra-transaction recovery is invoked, where that transaction is divided up among the slaves.

Fast-start parallel recovery is started when SMON discovers that the amount of recovery work that is required is above a certain threshold. The threshold is the amount of work at which the time it takes to recover in parallel becomes less than the time it takes to recover serially.

Database SUSPEND/RESUME

This new enhancement provides a mechanism by which all I/O in the database can be stopped, enabling copies of the database to be made without I/O interference. Two new statements are introduced. The ALTER SYSTEM SUSPEND statement stops new lock and I/O activity from being initiated. The creation of backups or archived logs is not affected. The database remains suspended until an ALTER SYSTEM RESUME statement is issued.

Control File Character Set

This enhancement enables the Recovery Manager to correctly interpret tablespace names for recovery when the character set is other than the default. This is done by storing a character set ID in the control file. When the database is first opened, this character set ID is propagated from the data dictionary into the control file database information record. The information is then available from the control file, before database open, for any subsequent recovery attempts.

If the control file is lost, recovery is still possible. The DBA may specify the character set as an argument on the CREATE CONTROLFILE statement.

Recovery Manager (RMAN)

Changes to the Recovery Manager are presented below. Additionally, the *Oracle8i Backup and Recovery Guide* has been extensively rewritten. Most notably, it now contains an exhaustive reference for RMAN command syntax, and a chapter on recovery catalog views.

Recovery Catalog No Longer Required

The use of a recovery catalog is now optional. If you choose not to use a recovery catalog, you can still use RMAN very effectively. RMAN obtains the information it needs from the control file of the target database. When using a recovery catalog, RMAN can perform a wider variety of automated backup and recovery functions. For this reason, Oracle recommends using a recovery catalog with RMAN whenever possible.

A related enhancement is discussed above in "Control File Character Set".

Media Management API, Version 2.0

RMAN includes changes to support version 2.0 of the Oracle Media Management API, commonly referred to as System Backup to Tape API, or SBT. Support for the version 1.1 Media Management API is maintained.

In order to utilize different types of tertiary storage for Oracle database backups, RMAN requires that third party media management software be installed. This software interfaces to Oracle8*i* and RMAN using SBT. It is the responsibility of this vendor supplied software to load, label, and unload sequential media such as tape drives for the purpose of backing up and recovering data.

Proxy Copy Through an enhancement called *proxy copy*, the media management vendor software is able to take over the entire data movement involved in a backup or restore. RMAN merely provides a list of files requiring backup or recovery to the media manager, which in turn makes all decisions regarding how and when to move the data. The PROXY option of the BACKUP command invokes this facility. The facility must be specifically supported by the media management software--not all media management software provides this support.

Media Pool Selection Some media management software allows backup media to be arranged into storage pools based on media type, retention period, or other criteria. The new POOL operand of the BACKUP command provides tight integration between such products and RMAN.

Vendor Identification and Improved Error Messages When a channel is allocated, RMAN will display in its log a text message identifying the media management product that will be used to take backups on that channel. When the media manager encounters an error, it will return a text error message explaining the error, which will be displayed in the RMAN log.

Send Command The new SEND command allows commands to be sent directly from an RMAN session to the media management software. Typically, this command is used to control runtime options of the media manager that can not otherwise be controlled using RMAN.

Crosscheck Catalog

RMAN can detect when backups expire from the media management software catalog or on disk, and it can update its own catalog accordingly. Two new commands, CROSSCHECK BACKUP and DELETE EXPIRED BACKUP, allow for the synchronization of the recovery catalog with the media manager's catalog.

The new CHANGE...CROSSCHECK command can perform crosschecks on both backup sets and image copies.

List and Report Commands

The output of the LIST BACKUP command is significantly improved. It now prints the list of backups belonging to a backup set in a separate section of the report from the list of data files or archived logs included in the backup set.

A new command, REPORT NEED BACKUP REDUNDANCY, has been implemented. When fewer than a user-specified number of backups of a data file exist, this report alerts the user that a new backup is required.

Recovery Catalog Maintenance Commands

The following new commands make it easier to create, upgrade, and drop the recovery catalog.

CREATE CATALOG This command creates the recovery catalog. It replaces *catrman.sql* and associated scripts in the admin directory, which had to be run manually to create the recovery catalog schema.

UPGRADE CATALOG When RMAN detects that the version of the recovery catalog is too old, it will issue an error message indicating that the recovery catalog requires upgrading. Previously it was necessary to run a SQL script to perform the upgrade. Now the upgrade is done with the UPGRADE CATALOG command. This command upgrades the recovery catalog from any prior version. No data is lost; the backups which were taken with the recovery catalog remain intact.

DROP CATALOG This command removes the recovery catalog schema.The backups themselves are not removed, so if there is a duplicated or replicated catalog available, or the information is available in the control file, the backups can still be used for restore.

Database Maintenance Commands

These new commands allow for more types of recovery with a single RMAN session. It is no longer necessary to exit RMAN to shutdown, startup, mount, or open the database.

STARTUP/SHUTDOWN These commands have the same syntax as the corresponding SQL*Plus commands.

ALTER DATABASE The following two forms of the ALTER DATABASE command are supported.

- ALTER DATABASE...MOUNT
- ALTER DATABASE...OPEN

DUPLICATE Command

The DUPLICATE command allows creation of a replicated database using the backups of another database. Typically, a database is duplicated in order to create a test database that is separate from the production database.

Node Affinity Detection

When backing up on multiple nodes of a parallel server, it is possible that some disks have 'affinity' to certain nodes in the cluster such that access to those disks is faster from those nodes than from other nodes in the cluster. RMAN recognizes node affinity, if it exists, and will attempt to schedule data file backups on channels allocated at nodes that have affinity to those files. To use this feature, channels must be allocated at more than one node in a parallel server cluster.

Backups

The following are RMAN changes affecting backups.

Duplexed Backup Sets Recovery Manager can now create up to four concurrent copies of each backup piece. This feature is particularly useful for archivelog backups. If a data file backup is bad, an earlier backup of the same data file can usually still be used for recovery, but if an archivelog backup is bad, then no recovery beyond the point in time when that log was created is possible. Because archivelogs are critical to recovery, customers often require that archivelogs be backed up twice before being deleted from disk.

Naming RMAN no longer requires that backup piece names be explicitly specified using the FORMAT parameter. By default, RMAN chooses a unique name for each backup piece.

Will No Longer Be Overwritten A backup piece will no longer be overwritten if an attempt is made to create a backup piece with the same name as an existing one. RMAN now issues an error.

Performing TSPITR Without A Recovery Catalog

It is now possible to perform TSPITR (Tablespace Point in Time Recovery) without a recovery catalog, however some restrictions apply.

Backup Performance Views

Two new views are available to monitor the progress and performance of Recovery Manager backups. These are V$BACKUP_SYNC_IO and V$BACKUP_ASYNC_IO. These views can be used to monitor the progress of individual files in a backup or restore, and to identify performance bottlenecks. A complete description of how to use these views to tune backups is in *Oracle8i Tuning*.

SQL*Loader Enhancements

For Oracle8*i* provides the following the SQL*Loader enhancements:

- SQL*Loader now includes support for the loading of objects, collections, and LOBs.
- There is no longer a 64K physical record size limit.

- A new keyword, FILLER, can be used to specify a filler field: *a data file mapped field which corresponds to no database column*. The filler field is assigned values from the data field to which it is mapped.
- New datatypes VARCHARC, VARRAW, LONG VARRAW, and VARRAWC are length-value pair datatypes similar to VARCHAR.
- The user can now specify a custom record separator in the OS-file-processing string.
- DEFAULTIF and NULLIF predicates can now be ANDed together.
- Field delimiters can now be one *or more* characters in length.

For more informations, see *Oracle8i Utilities*.

Export and Import Utilities

Many of the changes for the Import and Export utilities in Oracle8*i* are in support of other features. However, there are a few enhancements in these utilities which stand on their own.

- The ability to specify a query for the select statements that export uses to unload the tables.
- The ability to specify multiple dump files for an export command. (This allows users to circumvent the previous 2Gb limit for export dump files.)
- The ability to export tables containing LOBs and objects, even if direct path is specified on the command line.
- The ability to export and import precalculated optimizer statistics instead of recomputing the statistics at import time. (This capability is only applicable to certain exports and tables.)
- The ability to selectively inhibit the validation of type object identifiers by import.
- Support for larger dump files and dump tapes.

For more information on any of the above, see *Oracle8i Utilities*.

Operational Simplicity

Oracle is committed to making the Oracle database server simpler to use and administer. Reducing the number and duration of administrative tasks reduces the total cost of ownership for a system. Fewer administrators are required for

day-to-day operations, and they can spend their time on more challenging issues such as performance tuning and database design. For very low end systems in particular, Oracle is working toward making them as self-administering and self-tuning as possible.

The following Oracle8*i* components support Oracle's goal toward achieving operational simplicity.

Oracle Software Packager and Oracle Universal Installer

The Oracle Software Packager and Oracle Universal Installer work together to provide a robust method for packaging and installing Oracle8*i* software, whether for simple installations or those with complex interdependencies.

Oracle Software Packager The new Oracle Software Packager is a Java-based, object-oriented tool used for packaging products into components that can later be installed by the Oracle Universal Installer. Using wizards, it provides a visual development environment for generating installation packages in a format that can be interpreted by Oracle Universal Installer.

A component is the basic object to be installed by Oracle Universal Installer. The component wizard guides the install developer through the specification of the installation properties of the component, like component name, version number, operating systems and languages, the files or groups of files to be installed as part of the component, dependencies, variables, and the sequence of *dialogs* and *actions* that make up the installation plan. These properties are stored in a component definition file.

Dialogs represent user interface screens that are displayed to the installing user at run time and are used to assign values to variables. Action blocks are the state-change portions of the installation, and perform such tasks as copying files, setting environment variables, creating users, etc. The Oracle Software Packager is extensible and customizable.The install developer can take advantage of the packager's Java class libraries implementing predefined actions or dialogs, or request that the packager import custom Java classes.

Once the developer has defined the installation characteristics, Oracle Software Packager can verify the component definition and stage it.The verify wizard checks the component's integrity, while the staging wizard moves the component from the development area to a distribution medium, such as CD-ROM, Internet Server, or NFS system.

The Oracle Software Packager does not require access to a component to allow it to be defined as a dependent of the component being packaged. From simple

installation definitions to very complex bundled installations, Oracle Software Packager enables packaging of Oracle software, as well as easy integration of ISV and value-added reseller (VAR) software into a common installation.

For more information, see the *Oracle Software Packager User's Guide*.

Oracle Universal Installer Oracle8*i* is installed using the new Java-based Oracle Universal Installer, presenting the same look and feel across all platforms. Oracle Universal Installer interprets the staging area prepared by Oracle Software Packager and performs the appropriate installation actions. Its features include:

- It has the ability to read predefined answers to installation dialogs from response files defined by the administrator using the response file editor. This allows Oracle Universal Installer to run in silent mode.
- Installs can be from a CD, or they can be Web-based.
- Multiple Oracle_Homes are allowed, enabling multiple versions (different release levels) of the same component to be installed.
- Dependencies are automatically resolved.
- Postinstallation tools, such as the Database Configuration Assistant, are automatically launched at the end of installation, if specified by the software packager.
- Logging

The user is guided through the installation process by wizards and online help.

Configuration Improvements

The Oracle Database Configuration and Data Migration Assistants have been enhanced to further simplify the creation, deletion, modification, and migration of Oracle databases.

For information on either of these products, refer to your operating system specific Oracle installation documentation. Some features may not be available on all operating systems.

Oracle Database Configuration Assistant The Oracle Database Configuration Assistant enables the creation, modification, or deletion of an Oracle database. In Oracle8*i*, it can be invoked as a standalone Java application from the Oracle Universal Installer, or as an applet from the Java-based Oracle Enterprise Manager.

A wizard interviews the user for information relevant to the use and environment of the database. At the end of this process a pretuned starter database, based on a seed database distributed on the installation CD-ROM, can be created immediately, or a SQL script is generated for creating the database at a later time. The actual database generation runs in quiet mode from parameters generated or selected based on the interview. The creation of remote databases is supported using the Oracle Enterprise Manager's intelligent agent.

The Database Configuration Assistant provides the ability to create an Oracle OFA (Optimal Flexible Architecture) compliant instance. The OFA-like file layout provides a standardized file layout and eases management and maintenance.

If the user selects to modify a database, installed cartridges that have not previously been configured for use with the database can be configured. Also, multi-threaded server support for the database can be enabled or disabled.

When deleting a database, all database files except the initialization parameter file are deleted.

Oracle Data Migration Assistant Oracle Data Migration Assistant is a GUI tool that seamlessly transforms the data of an existing Oracle database to the current Oracle database release. It changes data file headers but leaves actual data unchanged. It does not copy data.

The Oracle Data Migration Assistant can migrate an Oracle7 database to Oracle8*i*. See your operating system specific Oracle documentation for information about the earliest release that the Oracle Data Migration Assistant can migrate on your operating system.

Oracle Enterprise Manager (OEM)

Version 2.0 of the Oracle Enterprise Manager delivers greater scalability and ease of use through the following enhancements.

- It uses a new three-tier model, allowing flexible deployment options.
- It is Java-based, allowing it to be accessed from any operating system, not just Windows NT.
- A shared repository allows administrator access from multiple accounts and consoles.

The first tier, of Oracle Enterprise Manager's three-tier model, consists of a Java-based console and integrated applications which can be installed or run from a Web browser. Administrators can access the console from virtually anywhere, and the 100% Java framework provides heterogeneous platform support.

The middle-tier component is the Oracle Management Server (OMS). Its main function is to provide centralized intelligence and distributed control between clients and managed nodes, thereby processing and administering all system management tasks. As the number of system management tasks increases, the architecture scales through the addition of more OMSs. Failover and load balancing are also automated within the OMSs, providing much greater reliability in notification processing.

The third tier is comprised of managed services or targets, such as databases, nodes, applications or application servers. An intelligent agent functions as the executor of jobs and events sent by the OMS. The agent works independently from the Console, the OMS, or network connections. Once an Agent is assigned a task from the OMS, it will perform the work at the time scheduled, and report status back to the OMS. An example of one such task might be performing weekly database backups.

Oracle Enterprise Manager, Version 2.0, is a multi-user system. Multiple administrators with separate accounts and permissions can log into their own consoles. A shared repository allows them access to the same information from their separate consoles.

For more information, refer to the Oracle Enterprise Manager documentation set.

Oracle Parallel Server

Numerous enhancements to Oracle Parallel Server (OPS) deliver major advances in performance, scalability, and manageability. Additionally, by extending commonly used single-instance diagnostic tools and functionality to be OPS-aware, DBAs, development, and support personnel are better able to diagnose, tune, and monitor systems. Summaries of these OPS enhancements are presented below.

For more information see *Oracle8i Parallel Server Concepts and Administration*.

New Architecture

In Oracle8*i*, new architecture dramatically decreases the need for workload partitioning and allows all applications to obtain the performance and scalability benefits from an OPS implementation. Changes include:

- A new "diskless" ping architecture, called cache fusion, that provides copies of blocks directly from the holding instance’s memory cache to the requesting instance’s memory cache. This functionality greatly improves inter-instance communication. Cache fusion is particularly useful for databases where updates and queries on the same data tend to occur simultaneously and where,

for whatever reason, the data and users have not been isolated to specific nodes so that all activity can take place on a single instance. With cache fusion, there is less need to concentrate on data or user partitioning by instance.

- Deeper integration of the distributed lock manager (DLM) into the database engine, increasing the efficiency of parallel cache management.
- A new, inter-process communication protocol that is highly efficient on modern interconnects.

Easier System Administration

Oracle 8*i* improves the diagnosis of system performance problems and the administration of the OPS environment with improved tools.

Diagnostic Enhancements

Oradebug is a utility used by consulting and support personnel to diagnose and troubleshoot poorly behaving systems at run time. Previously, it was best suited for single-instance situations, but now that functionality is extended for the Oracle Parallel Server.

Oracle Parallel Server Management (OPSM)

Enhancements to OPSM provide for greater ease in administration of the Oracle Parallel Server. A single generic interface is presented for administering parallel servers on any platform, helping administrators to easily manage the complexities of the Oracle Parallel Server product. Here are some of the features.

- A new Oracle Enterprise Manager applet, Parallel Server Manager, has been introduced for managing parallel servers.
- Installation and creation of OPS databases is handled by a separate tool called OPS Assistant.
- Group Membership Services (GMS) has been moved to the vendor-specific Cluster Managers (CM) and the Oracle database engine. There is no longer a discrete GMS module visible to the user that must be explicitly started before starting instances.

For more information about enhancements to OPSM, see the *Oracle Parallel Server Management Users Guide* and the *Oracle8i Parallel Server Setup and Configuration Guide*.

Parallel Server Installation and Database Configuration

The Oracle Universal Installer (see "Oracle Universal Installer" on page 2-59) and Oracle Database Configuration Assistant (see "Oracle Database Configuration Assistant" on page 2-59) are both cluster aware.

In Oracle8*i*, only a single installer session is required to install an OPS database. The installer collects node information from the user, distributes the required Oracle products to the specified nodes, and then invokes the OPS Assistant to set up the instances and create the database. When OPS Assistant is finished, the parallel server is up and available on all nodes, and the parallel server configuration information is saved so that OPSM can use it to manage the new parallel server.

For more information, see the *Oracle8i Parallel Server Setup and Configuration Guide*.

Instance Affinity for Jobs

Instance affinity for jobs is the association of jobs to an instance. Through the new DBMS_JOB package, the user can indicate whether a particular instance, or any instance, can execute a user-submitted job in the Oracle Parallel Server environment.

This Oracle8*i* feature can be used to improve load balancing and limit block pinging. For instance, using Oracle Parallel Server and replication at the same time will result in block pinging problems on the deferred transactions queue if all instances in a clustered environment decide to propagate transactions from the deferred transaction queue. By limiting the activity against tables to only one instance within the parallel server cluster, pinging can be minimized.

For more information, see *Oracle8i Supplied Packages Reference* and the *Oracle8i Administrator's Guide*.

Recoverability and Availability Improvements

There are recoverability and availability improvements which enhance the reliability of the Oracle Parallel Server. These are covered in more detail elsewhere in this chapter, but are mentioned briefly here.

- Parallel transaction recovery is performed on each node. See "Fast-Start Parallel Rollback" on page 2-52.
- Node affinity for Recovery Manager increases backup and restore throughput. See "Node Affinity Detection" on page 2-55.

- Establishing bounds on recovery time reduces false pinging. See "Fast-Start Checkpointing" on page 2-51.

Networking Improvements

The following new Net8 enhancements, described elsewhere, improve ease of use and scalability for the parallel server.

- A database can now have more than one service associated with it. See "Service Naming" on page 2-75.
- Connection load balancing balances the number of active connections among various instances and dispatchers for the same service. See "Connection Load Balancing" on page 2-76.

Distributed Systems

Oracle8*i* extends the functionality of advanced replication, focusing on mass-deployment applications. Heterogeneous services, introduced in release 8.0 and which implemented an extensibility framework for accessing non-Oracle systems, has also been further enhanced.

Advanced Replication Enhancements

The Oracle8*i* features and enhancements described below comprise the overall effort to optimize replication performance and make snapshot environment distribution and security more effective. All are included in advanced replication.

For more information, see *Oracle8i Replication* and *Oracle8i Replication API Reference*.

Performance Improvements

Significant performance gains are realized by the internalization of PL/SQL replication packages and by optimizations to snapshot refresh.

Internal Apply Packages Continuing the trend started with release 8.0, more replication code has been moved into the database engine. The PL/SQL generated packages used to apply replicated transactions at a remote site have been internalized. This allows replicated transactions to be more efficiently applied at remote sites and because packages are not generated, a site can be more quickly instantiated. Internal packages are also more secure because they are tamper proof.

Faster Snapshot Refresh Snapshot refresh has been optimized to support large refresh groups. There is improved support for subquery snapshots, and for null refresh (no

changes to the master tables since the last refresh). A single refresh group can now contain 400 snapshots, and the number of roundtrips required to refresh snapshots in a refresh group has been reduced. (This feature was first added in release 8.0.5.)

Improved Mass Deployment Support

In Oracle8*i*, Oracle adds new functionality to better support mass deployment and front office applications.

Parameterized Snapshot Deployment Templates These facilitate the mass deployment of information to support such applications as field service and sales force automation. These templates represent a grouping together of snapshots and other database objects to be instantiated at a node. They allow a DBA to centrally package a snapshot environment for easy, custom, and secure distribution to one or multiple sites. The goal is to create the environment once, then deploy the snapshot deployment template as often as necessary. Template parameters allow data subsetting at a remote site without redefining the template, and a template may be defined as public or private. Public templates may be instantiated at any site, whereas private ones can be instantiated only at predefined, authorized sites. An Oracle Replication Manager deployment wizard guides the DBA through the selection of schema objects to add to the template, the selection of parameters, and defining authorizations.

Column Level Snapshot Subsetting Updatable snapshots can now be subsetted horizontally (selected rows) or vertically (selected columns). Previously, only horizontal subsetting was allowed. Vertical partitioning allows the deployment of the minimum amount of data needed by a remote site, thus reducing connection time. It also protects snapshot sites from changes to their associated masters. A column can be added to a master site without impacting the snapshot site, or a column can be deleted and not impact the snapshot site, if the snapshot site does not currently reference that column.

Offline Instantiation Snapshot deployment templates can be instantiated online or offline. Online instantiation allows a snapshot site to instantiate the template while connected to the target master site. The advantage here is that the data will be current. However, this is at the cost of requiring a live connection, possibly of long duration and having the potential of generating extreme network traffic that could degrade other network services.

Offline instantiation allows the DBA to package the deployment group templates and required data onto some type of storage media (tape, CD-ROM, etc.) for distribution to a snapshot site. Instead of connecting to the master site, instantiation

can be done by pulling the template and data from the storage media. Users can fast refresh immediately after completing an offline instantiation; a full refresh is not required. Offline instantiation is an ideal solution for mass deployment situations where many disconnected laptops will be instantiating the target template.

Improved Security

While the scripts used to instantiate a snapshot site are generated at the master site and can control access to data, it is still necessary to connect to a receiver and proxy snapshot administrator to propagate replicated transactions and to refresh snapshots. Oracle8*i* enhancements to the replication security model eliminate certain security deficiencies regarding the granting of privileges to untrusted sites.

- Object privileges, as required by a receiver of remote procedure calls (RPCs) at a master site, are now automatically managed. Only required object privileges are granted to untrusted sites.
- Proxy snapshot administrators now have a way of accessing objects in object groups without being granted excessive privileges.

Replication Manager

A number of improvements have been made to the Oracle Replication Manager; some have already been mentioned. A major and noticeable enhancement is that it has been rewritten to conform to the new Oracle Enterprise Manager (OEM) Java interface. Oracle Replication Manager can now be run from anywhere in the network, and it is not constrained to a Windows-based machine. There is also a replication class of events in OEM which, for example, can be used to monitor errors or delinquent snapshot refreshes.

Improved Oracle Lite Integration

The integration between the Oracle8*i* Enterprise Edition and Oracle Lite has been improved to provide better performance and increased functionality. All of the replication changes previously described are supported, and Oracle Lite users with laptops at remote sites will especially benefit from reduced connection time.

Heterogeneous Services

Heterogeneous services (HS) implements an extensibility framework for accessing non-Oracle systems. Introduced in Oracle8, HS integrates the core of Oracle's gateway technology directly into the Oracle database server by extending the Oracle SQL engine to optimize and rewrite SQL for non-Oracle datastores. This closer integration has improved performance. For instance, the Transparent

Gateway for DB2 has provided a substantial performance improvement, particularly where large numbers of rows are retrieved.

Oracle8*i* enhances HS with the following changes. For more information, see *Oracle8i Distributed Database Systems*.

Multi-Threaded Service Agent

Heterogeneous service (HS) agents have been made multi-threaded. The architecture chosen for this implementation is similar to the Oracle MTS architecture. Specifically, there is a set of dispatcher threads to receive requests from Oracle database server processes and return results to them, and a pool of task threads to process the requests and compute results.

This enhancement will reduce the amount of system resources consumed when there are large numbers of user sessions concurrently accessing the same non-Oracle system. This more efficient use of system resources allows a greater number of concurrent user sessions.

Improved Management Interface: Fixed Views

A set of fixed views is now available to provide heterogeneous services status and to monitor information. The following views are added:

V$HS_AGENT	Identifies agent processes
GV$HS_AGENT	Identifies agent process used by all instances of an Oracle Parallel Server configuration
V$HS_SESSION	Identifies HS connections
GV$HS_SESSION	Identifies HS connections used by all instances of an Oracle parallel server

The views are present at all times in the Oracle database server and can be accessed directly through SQL queries. An OEM GUI applet also displays the information. A full description of each of these views is available in *Oracle8i Reference*.

Miscellaneous Improvements

Agent self-registration was introduced in a previous release. It reduces or eliminates the need for DBA intervention in configuring heterogeneous services. In Oracle8*i*, code has been rewritten to make the self-registration process more efficient.

New in Oracle8*i* is an agent-specific shared library (DLL) for HS object files, other than drivers, that is substituted when linking agent executables. While the benefits

from this change are platform-specific, it can improve scalability by using a single agent library for all types of agents (extproc, hsalloci, hssqlpss, hsdepxa, and hsots). Also, memory requirements may be reduced because agent executables become quite small.

Remote Join Enhancements

In distributed database systems where it may be necessary to join tables across systems, poor data access techniques can cause user data flow between database servers to be a major factor in overall performance. In order to determine the best access technique for an optimum execution plan, the optimizer must first be able to determine a sufficient number of alternate paths. The remote join enhancements in Oracle8*i* present more options, allowing for better execution plans to be generated with a corresponding performance increase.

Networking, Security, and Oracle Advanced Security

Oracle8*i* security and networking enhancements support standards, provide stronger security, and lower the cost of ownership. The support of standards provides for more choice, and provides for greater ease of system integration between Oracle and non-Oracle components. When allowing remote access and extending networks to the Internet, you now have a variety of standards-based and proven methods for data encryption and authenticating users, databases, and Web servers.

The *virtual private database* feature secures data in the database by providing security at the row-level, making it easier to handle sophisticated security requirements. It also moves the definition of the security policy out of the application and into the database, thereby eliminating the potential to circumvent the security enforcement.

Centralized user management, through integrated security and directory services, will lower your cost of ownership. Fewer accounts means less administrative overhead, and single signon increases productivity by reducing the number of logins. It can also reduce lockouts resulting from a failure to remember all of the necessary passwords.

Securing Data in the Oracle Database Server

Oracle 8i introduces the virtual private database feature. With so many means of accessing the database available, both locally and remotely, and most notably through the Internet, server-enforced and granular access control becomes crucial. Security implemented in application code is insufficient, because to bypass the

security one only needs to access the data from outside the application (through SQL*Plus, for example). The virtual private database feature addresses this problem by attaching the security policy directly to a table or view.

An important benefit of the virtual private database feature is that security is built only once, and not into every application that accesses the data. Then, no matter how a user gets to the data, the same security policy is enforced.

The virtual private database feature has two components: fine-grained access control and application context. Here is how they work together to implement a safe and consistent security policy across applications.

Fine-Grained Access Control

Fine-grained access control works using a dynamic predicate, or WHERE statement. This dynamic predicate allows the security rules to be acquired at SQL statement parse time when the base table or view is referenced in a select or DML statement.

Implementing fine-grained access control consists of creating a function to implement a security policy and associating the function with a table or view. The Oracle supplied DBMS_RLS package is used to associate security policy functions with the table or view. When a query or subquery which affects that table or view is executed, the server dynamically rewrites the query appending a WHERE condition generated by the function implementing the security policy.

For example, fine-grained access control can be used to enforce a security policy "customers can see only their own orders". An application might submit a SQL statement to select "all" orders. The security policy would append an additional predicate on the WHERE clause restricting the statement to retrieve only those orders submitted by that customer. This provides row-level, server enforced, access control, that needs to be built only once to protect your data no matter how the data is accessed.

For more information, see the *Oracle8i Application Developer's Guide - Fundamentals* and *Oracle8i Supplied Packages Reference*.

Application Context

An application context functions as a secure data cache for storing information used to make access control decisions. It provides more flexibility and space than using CLIENT_INFO, and represents a new approach.

Application contexts are extensible. Each application can have its own context, with its own attributes. For example, an order entry context could have customer number and sales region attributes, while a human resources application might

have employee number, position, and country code attributes. These attributes are used by fine-grain access control in enforcing a security policy.

An application context can be set and reset only by a designated package. The CREATE CONTEXT statement specifies the namespace of the context and the package to which it is bound. A secure and consistent API is provided to manage and access the application context.

For more information, refer to the *Oracle8i Application Developer's Guide - Fundamentals* and *Oracle8i SQL Reference*.

Secure Network Computing Through the Oracle Advanced Security Option

With the continuing growth of distributed systems involving numerous databases and applications comes an ever increasing challenge of user authentication and user management. To meet this challenge of *enterprise user management*, Oracle8*i* provides methods of encryption, authentication, and authorization, along with integrated security and directory services, packaging them as Oracle Advanced Security.

The Advanced Networking Option (ANO) has been renamed to Oracle Advanced Security to better describe its focus and new functionality. Oracle Advanced Security runs on top of Net8 to deliver security solutions to the Oracle Network and beyond through the integration of industry standards for encryption, authentication, and remote access. Oracle8*i* expands the choice of security solutions by offering SSL (Secure Sockets Layer) and RADIUS (Remote Authentication Dial-In User Service) protocol adapters.

Oracle8*i* also enhances enterprise user management by offering integrated security and directory services through Oracle Internet Directory (O*i*D). User information is stored and managed in a directory and is accessed by an LDAP (Lightweight Directory Access Protocol) version 3 compliant server.

Note: The Oracle Security Server (OSS) product no longer exists in Oracle8*i*, but its functionality has been integrated into the standards-based products: Oracle Advanced Security and Oracle Internet Directory. Users of OSS must migrate their applications to use these Oracle8*i* components.

For more information, see *Oracle Advanced Security Administrator's Guide*.

SSL

Oracle Advanced Security includes implementation of a SSL protocol adapter. SSL is an industry standard protocol for securing network connections and is widely

used over the Internet. It secures Net8 connections and other protocols as well, including IIOP connections used with thin clients and Enterprise JavaBeans.

SSL uses digital certificates and a public key infrastructure (PKI) to provide the major pieces of security:

- authentication of people and machines
- encryption techniques for privatizing
- checksums for protecting against data modification or snooping

The importance of verifying the identity not only of users, but also of machines, becomes crucial when an organization opens its doors to the Internet.

Most organizations have chosen to protect their databases by keeping them inside of the network and placing a Web server outside of the network. Used in combination with a firewall or other security measures, the data can remain safe. The data server's ability to authenticate the Web server, therefore, is extremely important. You must verify that the Web server is what it claims to be, so that you can trust it to pass data--encrypted or unencrypted--to the outside world. SSL provides the critical authentication piece, along with the data privacy piece, of the security solution.

Oracle's implementation of SSL uses industry-standard X509 version 3 digital certificates for authentication. Briefly, the authentication works as follows:

- A client is issued an X.509v3 certificate and a private key by a *certificate authority* (CA), and stores the certificate in an *Oracle wallet*.
- When the client initiates a Net8 connection to the server, SSL uses the private key and certificate to perform a handshake (transparent to the user) between the two processes.
- If it is successful, the client is granted access.

SSL supports single signon in that the user only requires one password: that which opens the wallet. The certificate and private key are used to authenticate the user to multiple services. The complete package for this authorization requires components and functionality as described in "Integrated Security and Directory Services" below.

SSL does not provide authorizations (Oracle roles), that must be provided by the application. SSL provides only public key-based authentication; this is distinct from other authentication methods provided by Oracle Advanced Security, which include tokens, smart cards, and biometric devices.

RADIUS

The RADIUS (Remote Authentication Dial-In User Service) protocol is an industry standard for remote authentication and controlled access to networks. It is widely accepted because of its flexibility, its ability to handle many devices, and its ability to provide user authentications, authorization, and accounting between a network client and an authentication server.

RADIUS support in Oracle Advanced Security offers two major benefits.

- It readily integrates into existing systems by making the Oracle8*i* database server a RADIUS client and thereby capitalizing on the infrastructure and investment that organizations have already made.
- It extends the solutions for authenticating users to Oracle by enabling support for new authentication technologies such as token cards, smartcards, and challenge-response mechanisms.

Your choice of authentication mechanisms is extensively expanded, with little development effort required.

Integrated Security and Directory Services

> **Note:** The features discussed here for integrated security and directory services are available only as a beta release. They are not an integral part of this release (8.1.5), but are planned to be available in the next production release.

Enterprise user management and security presents many challenges. Users often have too many passwords, consequently they write them down or choose the same password for all accounts. They have multiple accounts that can be cumbersome and time-consuming for an organization to manage. Another challenge is that common application information is often fragmented across the enterprise, leading to data that is redundant, inconsistent, and expensive to manage.

Oracle has recognized these challenges and has continued to evolve solutions. The Advanced Networking Option was introduced in Oracle7, providing data encryption and supporting multiple authentication mechanisms, including biometrics, DCE (Distributed Computing Environment), and Kerberos. Oracle8 introduced Oracle Security Server (OSS), supporting single signon using X.509 version 1 certificates and proprietary authentication protocol and components. Oracle8*i* enhances the security architecture of Oracle8 by providing an open, standards-based approach for enterprise user management.

Oracle8*i* addresses the enterprise's need for strong security and centralized management of application information by offering integrated security and directory services; specifically, by storing and managing user information in a directory. Multiple Oracle applications can rely on a common centralized definition of a user to determine which applications, services, and data servers a user may access, and with what privileges. The benefits of this approach include:

- Single signon to multiple services throughout the enterprise
- Single enterprise user, instead of multiple accounts per user
- Reduced total cost of ownership through single station user administration
- Well integrated, standards-based public key infrastructure (PKI)

Oracle8*i*'s integrated security and directory services incorporates multiple components, described below.

Oracle Wallet Manager Oracle Wallet Manager is a tool that is used to manage an Oracle wallet. An Oracle wallet contains a user's credentials used for authenticating the user to multiple services, such as data servers and application servers. Specific wallet contents include:

- A user's X.509v3 certificate. This certificate is analogous to a driver's license containing a user's identifying information, and it allows a user to be uniquely identified within an organization, thus enabling strong authentication. Oracle requires this certificate be issued by an X509v3-compliant certificate authority.
- A user's private key. This is issued by the certificate authority at the same time as the X.509v3 certificate, and they are both used by SSL in authenticating the user.
- Certificate trustpoints. This is a list of certificate authorities that the user trusts.

The Oracle Wallet Manager gives users (and administrators of data servers) complete control over the contents of their wallets. The contents of the wallet are encrypted with a key based on a user-specified password. Since the wallet is encrypted, it may be stored locally on a user's disk, or centrally in a directory accessible via LDAP.

The user needs only to remember one password, which opens the wallet, to access his or her credentials, which can then be used to authenticate the user to multiple services. And, those services need no longer store and manage local passwords for users.

Certificate Authority Users may obtain certificates from any X509v3 certificate authority, such as VeriSign, Inc. This is an offline and email process, and may take from a few days to weeks. Having a certificate issued in this manner guarantees its authenticity.

Certificates can be loaded into Oracle wallets and Oracle Internet Directory using Oracle Wallet Manager. Publishing a wallet into a directory enables a user to access his or her credentials from anywhere in the organization. The wallet can be retrieved by the user as needed by sending a "certificate request" and password to the certificate authority, which authenticates the request using SSL.

Note: While OSS provided a means to create user certificates, Oracle currently has no such functionality. As mentioned above, users may obtain certificates from any X509v3 certificate authority

---

Oracle Internet Directory The Oracle Internet Directory is an LDAPv3 compliant directory. It combines a native implementation of the Internet Engineering Task Force's (IETF) LDAP v3 standard with an Oracle8 back-end datastore. This design provides the Oracle LDAP Directory Server with a high degree of scalability, reliability, and compatibility with standard interfaces. It is specifically designed to meet the needs of managing user and system configuration data, including security attributes and privileges, in an Internet environment.

The management of Oracle wallets and their contents has already been discussed, but Oracle8*i* also extends the benefits of *enterprise roles* by storing them in and retrieving them from the Oracle Internet Directory. Oracle8 introduced enterprise roles, but they were stored in a proprietary repository.

Enterprise roles enable centralized authorization of users. For example, a user may be granted the enterprise role "HR clerk", which contains the global role "HR user" on the human resources database, and the "employee" global role on the corporate information database. If a user changes jobs, his enterprise role assignment can be changed, altering his privileges in multiple databases throughout the enterprise. Also, an administrator can add capabilities to enterprise roles (granted to multiple users) without having to update the authorizations of each user independently.

Oracle8*i* enables LDAP access for applications running within the data server. For example, an HR application could query a directory server to retrieve information about what organization a user belongs to, or request a user’s certificate, and use the results of the query to limit or expand the data the user is allowed to see, or the functions the user is able to perform. Oracle8*i* provides a number of Java APIs to enable applications to access an LDAP directory form within the data server.

Directory-enabled Oracle Security Manager The Directory-enabled Oracle Security Manager is an administrative tool that provides single station administration of privileges throughout the enterprise. Security administrators can manage users and authorizations across multiple Oracle8*i* databases and Oracle Internet Directory from a single console. In the production release of Oracle's integrated directory and security services, this tool will be become Oracle Security Manager, which will be part of Oracle Enterprise Manager.

Net8 Enhancements

It is also now easier to set up, configure, and administer the Oracle network, using the new Net8 Easy Config and enhanced Net8 Assistant tools. These tools are not discussed in this book, but they are described in Net8 documentation.

For more information about Net8 enhancements, see the *Net8 Administrator's Guide*.

Service Naming

Up to this release, the client was configured with the system identifier (SID) of a database instance. This SID was then passed to the listener. The listener would then verify this information and permit or deny a connection. While a SID identified a database instance, it did not identify a database. This limitation did not permit a database to have more than one service or replication of data among databases. Because a database can serve multiple services, SID has been replaced with *service naming*. Service naming allows clients to access:

- a service as a whole through the service name
- an instance of a database (the way SID did) through the instance name

Service naming can include multiple services provided by a single database and services that span multiple instances. For example, a personnel system called HR and finance system called FINANCE could reside in the same physical database called SALES.US.ORACLE.WORLD. Administrators could define service names according to the name of the services being provided (and not the machine type, node name, or database name). The following service names might be created:

- SALES could be created to match the database name
- HR could be created to match the name of the personnel system
- FINANCE could be created to match the name of the finance system

A service can also be implemented as multiple database instances.

To support services that include multiple instances, use the following new parameters in connect descriptors:

SERVICE_NAME	Allows clients to access a service as a whole. It may span instances or nodes.
INSTANCE_NAME	Allows clients to access an instance of a database (the way SID did).

Further, in previous releases, an alias for a connect descriptor was *service_name*. Because SERVICE_NAME now is a parameter in the connect descriptor, *service_name* has been replaced with *net_service_name*.

Automatic Instance Registration

Database instances register themselves with the listener when started. Prior to this release, information about the instance was required to be manually configured in the *listener.ora* file. Instance registration is comprised of:

- service registration provides the listener with instance information, such as database service names and instance names.
- MTS dispatcher registration provides dispatcher information to the listener

This feature minimizes the system configuration since there is no need manually enter instance names in *listener.ora* files. Database instance registration enables dynamic load balancing and failover.

Connection Load Balancing

Database instance registration enables connection load balancing because of the registration that happens with remote listeners. Connection load balancing balances the number of active connections among various instances and dispatchers for the same service. This enables listeners to make their routing decisions based on how many connections each dispatcher has and on how loaded the nodes that the instances run.

The load of a dispatcher is determined by the number of connections to the dispatcher. The load of an instances takes into account the load of the node it is running.

Connection load balancing is only enabled for an MTS environment. It is not configurable by clients.

Native Authentication on Windows NT

The Net8 native authentication adapter on NT has been improved in the following ways:

- Authentication is faster. The server uses the same authentication mechanism that the client used to log into the NT system.
- The authentication adapter uses Security Support Provider Interface (SSPI) to authenticate.
- The client and server mutually authenticate each other. Previously, only the server authenticated the client

Backward compatibility with previous versions of Oracle is maintained, while integration with new security mechanisms in the future will be easier.

For more information about native authentication on Windows NT, refer to the Windows NT documentation "Getting Started" guides for Oracle8*i* and Net8.

Multi-Tier Authentication and Authorization

Oracle8*i* also offers security to multi-tier architectures. In some systems, the middle tier is super-privileged to perform any action on behalf of any user, and the identity of the real client is not preserved through the middle tier. Middle tiers, especially Web servers or application servers, often sit on or outside a firewall, so limiting their access and auditing their actions is especially important.

Oracle8*i* provides the ability to preserve the real client identity through the middle tier and limit the users on whose behalf a middle tier can connect. The server can also audit actions taken by the middle tier on behalf of a particular user.

To support this feature, syntax changes have been made to the ALTER USER and AUDIT statement. For more information, see *Oracle8i SQL Reference*.

Program Interfaces

This section covers the Oracle8*i* new features and enhancements for the various program interfaces used to access data in the Oracle database. PL/SQL, the Pro*C/C++ and Pro*COBOL precompilers, and the Oracle Call Interface (OCI) are included. The new features supporting Windows NT integration are also discussed.

PL/SQL

For more information on any of the PL/SQL features or enhancement described here, see the *PL/SQL User's Guide and Reference*. Additional references may be noted in the descriptions.

Autonomous PL/SQL Blocks

Autonomous PL/SQL blocks are PL/SQL blocks that have a transaction scope independent of the transaction scope of the calling PL/SQL block. They can perform operations, commit, and rollback independent of the transactions in the calling block, before returning to the calling block. Transactions within an autonomous PL/SQL block are referred to as autonomous transactions. They are not nested transactions in that they do not share resources with the calling transaction and committed changes are immediately visible to other transactions regardless of whether the calling PL/SQL block commits or rolls back. A PL/SQL stored procedure/function, local procedure/function, a package/type method, or a top-level anonymous block can be declared to be an autonomous block using the pragma specification:

```
pragma AUTONOMOUS_TRANSACTION
```

The above specification can appear anywhere in the declaration section of a PL/SQL block, but at most once.

The Oracle8*i* autonomous transactions feature provides a means of constructing reusable application subcomponents that perform specialized operations.

Invoker Rights

The invoker-rights model, introduced in this release, allows programs to be executed with the privileges of the calling user. Prior releases used a *definer-rights* model wherein programs execute with the privileges of the creating user. The syntax of the CREATE FUNCTION, CREATE PROCEDURE, CREATE PACKAGE, and CREATE TYPE statements has been altered to allow an *invoker_rights_clause*. If AUTHID CURRENT_USER is specified, the program becomes an invoker-rights program. If not specified, the default is AUTHID DEFINER and there is no change from past releases.

The invoker-rights model is another step toward the creation of reusable code--a single code base that can be used by multiple users to manage data stored in their schemas.

For more information, see *Oracle8i Concepts*.

PL/SQL Bulk Binds

For PL/SQL, Oracle8*i* provides functionality to send a collection of bind variables (bulk bind) to the SQL engine with a single SQL statement, rather than requiring the same statement to be sent multiple times, differently bound each time. This functionality is similar to that already provided with OCI and Pro*C/C++. It reduces communication overhead and can significantly improve performance.

Dynamic SQL in PL/SQL

Dynamic SQL adds power and flexibility to host languages by allowing an application to generate and submit SQL statements for execution at run time. Prior releases provided this capability through programmatic interfaces in the DBMS_SQL package. Oracle8*i* provides dynamic SQL processing support in PL/SQL that makes dynamic SQL processing extremely simple, efficient, and easy to use. It is seamlessly integrated with SQL, much like static SQL in PL/SQL, and its performance is comparable to that of static SQL.

Parameter Passing by Reference

In Oracle8*i*, PL/SQL supports three parameter passing modes: IN, IN OUT, and OUT. IN parameters are passed by reference; IN OUT parameters support copy-in and copy-out semantics; OUT parameters support copy-out semantics. Through a new syntax, using NOCOPY mode, Oracle8*i* allows all parameters to be passed efficiently by reference. This achieves a significant performance gain for any application, such as a data cartridge, that passes large data structures as IN OUT and OUT parameters.

PL/SQL Procedures for Supporting REF-based Operations

Oracle 8i provides the UTL_REF package, containing procedures to support reference-based operations. Specifically, given a REF, there are procedures for selecting, locking, updating, and deleting an object.

Unlike SQL, the UTL_REF procedures enable the writing of generic type methods without knowing the object table name.

For more information, see *Oracle8i Supplied Packages Reference*.

Monitoring and Analysis of Program Execution

Oracle8*i* provides three new APIs, in the form of packages, that facilitate the tracing, debugging, or profiling (including code coverage) of PL/SQL applications.

For more informations about any of these packages, see *Oracle8i Supplied Packages Reference*.

DBMS_TRACE This package provides a means for tracing the execution of PL/SQL programs on the server. It contains procedures to start and stop the tracing of calls to PL/SQL functions or procedures, and exceptions. Data is written to the database trace log file.

DBMS_DEBUG Third-party tool vendors can leverage this low-level API to provide debuggers to PL/SQL developers.

Debugging requires two database sessions. One session runs the target code, which invokes DBMS_DEBUG to cause the PL/SQL server to generate debug events. The other session, the supervising session or debugger, uses DBMS_DEBUG to communicate with and read events posted by the target session. Several debuggers are already available using this API, such as Oracle Procedure Builder.

DBMS_PROFILER This profiling API provides PL/SQL programs with services for collecting and persistently storing data, in database tables, that can help identify and isolate performance problems or provide code coverage information. It is possible to generate profiling data for all named library units in a single session. For this release, the data includes the total number of times each line of code is executed, the total amount of time spent executing the line, and minimum and maximum times spent on a particular execution of the line.

A simple sample suite of analysis and reporting tools is provided, and the use of database tables allows ad hoc querying. Third-party tool vendors can now provide more sophisticated analysis and reporting tools for profiling and code coverage.

Increased Package Body Size

Oracle8*i* relaxes the limit on the size of PL/SQL package bodies.

Purity Rules Relaxed

In earlier releases, users were expected to qualify functions and procedures with the RESTRICT_REFERENCES pragma, which has various options (WNDS, RNDS, WNPS, RNPS). This specifies the purity of a function or procedure if the user wishes to use the function for certain operations. This purity information, combined with additional computed information, is then used to determine whether it is safe to perform the following operations:

- calling a function from a query or DML statement

- parallelizing a query or DML statement which calls a function
- creating a snapshot with fast refresh

In Oracle8*i*, these rules have been relaxed to be more user-friendly.

More details about this, refer to the *Oracle8i Application Developer's Guide - Fundamentals*.

Precompiler Enhancements

The Pro*C/C++ and Pro*COBOL compilers, which allow the embedding of SQL statements in source code, have been significantly enhanced for Oracle8*i*.

Pro*C/C++

The following table lists the Oracle8*i* enhancements for the Pro*C/C++ Compiler.

Enhancement	Description
CALL Statement	The CALL embedded SQL statement invokes a stored procedure. It can be used instead of an embedded PL/SQL block in new applications.
LOB Support	A new embedded SQL interface allows the manipulation of LOBs (large objects).
Collections	A new embedded SQL interface to allows the manipulation of varray and nested table data.
ANSI Dynamic SQL	This is the implementation of ANSI dynamic SQL Method 4.
PREFETCH Option	This precompiler option speeds up database access by "prefetching" values, thus cutting down the number of network round-trips.
External Routines	External routines written in C can be called from PL/SQL blocks. The REGISTER CONNECT embedded SQL statement is used by the procedures.
HEADER Option	The precompiler option, HEADER, specifies that precompiled header files are to be created and used to reduce the time and computer resources needed for developing large projects.
Calling Java from PL/SQL	Stored procedures written in Java can be called from an application.
DML Returning Clause	This clause is allowed in INSERT, DELETE, and UPDATE statements.

Enhancement	Description
Universal ROWID	The universal ROWID datatype is supported. Index-organized tables use this concept.
SYSDBA / SYSOPER Privileges in CONNECT Statements	These privileges can now be set in the CONNECT statement.
CLOSE_ON_COMMIT Precompiler Option	The CLOSE_ON_COMMIT micro precompiler option allows a choice of whether or not to close all cursors when a COMMIT is executed and the macro option MODE=ANSI.

For more informations, see the *Pro*C/C++ Precompiler Programmer's Guide*.

Pro*COBOL

The Pro*COBOL precompiler was largely rewritten for release 8.0, introducing many new features, but with an emphasis on greater compatibility with DB2. Oracle8*i* continues Oracle's effort to enhance the functionality of the Pro*COBOL precompiler, providing support for new Oracle8*i* database server features. The following table summarizes the Oracle8*i* enhancements.

Enhancement	Description
CALL Statement	The CALL embedded SQL statement invokes a stored procedure. It can be used instead of embedded PL / SQL block in new applications.
LOB Support	An embedded SQL statement interface allows LOBs (large objects) to be used in precompiler applications.
ANSI Dynamic SQL	This is the implementation of ANSI dynamic SQL Method 4.
PREFETCH Option	This precompiler option speeds up database access by "prefetching" values, thus cutting down the number of network round-trips.
Calling Java from PL / SQL	Stored procedures written in Java can be called from a Pro*COBOL application.
DML Returning Clause	This clause, which can save round-trips to the database server, is now allowed in INSERT, DELETE, and UPDATE statements.
Universal ROWIDs	The support for universal ROWID datatype is presented. Index-organized tables use this concept.

Enhancement	Description
User-Specified Runtime Contexts	COBOL subprograms can now be called by C programs. The context can be passed by a Pro*COBOL program to a C subprogram
SYSDBA / SYSOPER Privileges in CONNECT Statements	These privileges may now be set using the CONNECT statement.
Tables of Group Items	Tables of group items are now allowed as host variables in Pro*COBOL.
WHENEVER Do Call Branch	The WHENEVER directive has a DO CALL action: a subprogram is called.
Decimal-Point is Comma	The DECIMAL-POINT IS COMMA clause is supported. This permits commas to be used instead of decimal points in numeric literals.
Optional Divisions	The following divisions and their contents are now optional: IDENTIFICATION, ENVIRONMENT, DATA.
NESTED Option	When set to NO, the NESTED precompiler option will prevent generation of the GLOBAL clause for non-nested programs.

For more informations, see the *Pro*COBOL Precompiler Programmer's Guide*.

Java Interfaces

The Java interfaces, JDBC and SQLJ, are discussed in the Java section of this document. See:

- "JDBC Drivers" on page 2-17
- "SQLJ Translator" on page 2-18

OCI Enhancements

For information about these OCI enhancements, see the *Oracle Call Interface Programmer's Guide*.

DML Returning Enhancements

Enhancements have been made to make the DML RETURNING clause more optimized for common usage patterns. Data transfer is minimized and the interface is simplified for these common cases. Specifically, OCI provides selective data

transfer extensions to the use of the RETURNING clause which returns data only when it has been changed in the server. Also, there is additional OCI functionality for single row and array DML operations in which each iteration returns more than one row.

Enhanced Array DMLs

OCI now provides an error-batching mode for an array DML statement execution. In prior releases, OCI aborted if an error was signalled during an array DLM operation. Oracle8*i* can allow the DML to complete, but trap any erroneous rows. The erroneous rows are returned at one time after inserting / deleting all correct rows. The user can then make necessary corrections, and do another update operation. Consequently, the number of round-trips a user might have to perform has been reduced to a maximum of two.

OCI Callback Registrations

OCI callbacks now provide a means of tracing OCI calls for debugging and performance measurements. Two types of callbacks can be registered, entry and exit, wherein additional pre or post processing can be done. They can also be used to substitute the body of the function with proprietary code to execute on a foreign data source. For licensed ISVs and internal Oracle groups, a dynamic callback registration mechanism is provided. This mechanism allows a core-compliant dynamically linked module to register callbacks at OCI environment initialization time. This is transparent to the OCI application and does not require any changes in a running application.

OCIDescribeAny Enhancements

OCIDescribeAny was introduced in Oracle8 as a simple and uniform OCI interface to describe schema-level database objects: tables, views, sequences, synonyms, procedures, functions, packages, and types. It has been expanded to describe attributes of sub-schema objects like columns of a table or fields in a type, and to include more attributes of currently described database objects.

Asynchronous Event Notification

The publish / subscribe and system event notification capabilities of Advanced Queuing were discussed in "AQ-Based Publish / Subscribe" on page 2-38. OCI provides a programmatic API through which an application or user can subscribe to be notified of a database or system event, and be notified asynchronously through a callback.

Non-Blocking OCI

Oracle8*i* provides non-blocking capability with OCI. Essentially, it allows a user to specify or set a non-blocking polling mode for OCI call execution. This mode can either be specified for a whole application or at the individual call level. OCI non-blocking mode allows applications to process other events while an OCI call is in progress.

Windows NT Integration

Microsoft Windows NT® is an increasingly popular environment for database development and deployment. Oracle8*i* contains several features to make it easier to develop Oracle-based applications with Microsoft products. The Oracle database server on Windows NT provides a highly available, scalable, and secure platform for application deployment. This section discusses the features supporting Windows NT integration offered in Oracle8*i*.

Oracle Application Generator for Microsoft Visual Studio

The Oracle Application Wizard (AppWizard) provides developers with a GUI tool with which to quickly and seamlessly create a Visual C++, Visual InterDev, or Visual Basic application that provides connectivity and data access to an Oracle database. A developer can use this tool to generate an Oracle database enabled application that compiles, links, and runs without writing a single line of code.

AppWizard is thoroughly integrated with the Visual Studio IDE, which is the most popular IDE for developing C++ and Visual Basic applications on the Windows NT/95 platform. It allows ISVs, VARs, and other users to easily build applications that leverage Oracle database technology in the Visual Studio IDE.

The wizard can be invoked whenever a developer decides to create a new project in Visual Studio. AppWizard guides the developer through a two-step process. First, the wizard prompts the user with questions about the database connection that is to be used: connection string, username, and password to the Oracle database. Then, the user is allowed to select the specific tables and columns from which the application will retrieve data.

Second, based on the developer's responses, AppWizard will generate a Visual Studio project and source code that provide the developer with a custom tailored application framework from which to start developing immediately. The generated code framework will consist of a mixture of Microsoft Foundation Classes (MFC) and Oracle Objects for OLE classes. The MFC code provides the basic GUI application code and the OO4O classes provide the connectivity and data access to Oracle databases.

Oracle Objects for OLE

Oracle Objects for OLE (OO4O) is a COM-based database connectivity tool that provides seamless and optimized access to Oracle databases. OO4O can be used in environments ranging from the typical two-tier client/server applications to application servers deployed in n-tier environments and Web servers such as Microsoft IIS or MTS. It can be used from virtually any programming or scripting language that supports the Microsoft COM Automation technology, such as Visual Basic, Visual C++, VBA in Excel, VBScript and JavaScript in IIS Active Server Pages. OO4O consists of an *in-process COM Automation Server*, a *C++ class library*, and the *Oracle Data Control*.

The Oracle8*i* version of OO4O will enable developers who utilize COM/DCOM based development tools to have seamless access to all Oracle specific features that are presently inaccessible from other ODBC or OLE DB-based components such as ADO. Significant enhancements in this release include:

- Efficient management of database connections and user sessions (connection pooling and multiplexing)
- Full support for Microsoft Transaction Server (MTS) coordinated transactions
- Seamless access to instances of new datatypes introduced in Oracle8 including:
  - Object References (REFs)
  - Object Instances (Objects)
  - Nested Tables
  - Varrays
  - BLOBs, CLOBs, NCLOBs and BFILEs
- Easy-to-use interface for describing schema objects
- Full support for accessing Oracle Advanced Queuing capabilities

Extensive online help is available with Oracle Objects for OLE, including a *Quick Tour* for getting started.

Microsoft Transaction Server Integration

Microsoft Transaction Server (MTS) is a middle-tier application server for managing distributed applications. It offers an ActiveX/DCOM based programming model to develop distributed applications and a runtime environment to deploy these applications. Oracle8*i* provides full, native integration with MTS. It allows a user to

create distributed transactions coordinated by MTS with Oracle databases on any platform as participants.

Prior to Oracle8*i*, customers used Microsoft's XA solution for MTS to support transactions involving Oracle data. The transactions are coordinated by MTS but are mapped to the XA protocol. But Oracle8*i* now provides much better integration by supporting the OLE transaction interfaces published by Microsoft. This solution will perform much better and with less configuration.

COM Cartridge

This cartridge allows PL/SQL developers to programmatically manipulate COM/DCOM objects using the OLE automation interface (IDispatch). By allowing COM manipulation in PL/SQL stored procedures, functions, and triggers, developers have access to the thousands of pre-built and reusable COM objects that exist today. This feature enables developers to leverage COM objects that they build or buy from third-party vendors. Furthermore, it allows customers to extend the functionality of PL/SQL to suit their requirements.

3

Release 8.0 New Features and Options

The following topics are included in this chapter:

Partitioned Tables and Indexes

Very large databases present significant challenges for administrators and application developers. Large amounts of data complicate administrative tasks and affect the availability of the database.

To improve availability, ease administration, and enhance query and DML performance, the Oracle8 Enterprise Edition allows tables and indexes to be partitioned, or broken up, into smaller parts based on a range of key values. Because partitions operate independently of each other, data in a partitioned table are available even if one or more partitions are unavailable. Partitions also make large tables easier to manage by breaking up administrative operations into smaller tasks, which in turn can be performed in parallel. Finally, partitioning a table or index can improve performance of operations on the data by eliminating unneeded partitions from the execution plan of the operation.

Partitioning is transparent to both applications and users, so standard SQL statements in existing applications can run against partitioned tables. The Oracle optimizer is partition-aware, and partitions which do not contain any data required by a query are eliminated from the search, often resulting in a substantial performance increase.

What is Partitioning?

A table or index can be partitioned or divided into smaller pieces. You define a table or index partitioning strategy when creating the structure. Pick a column or set of columns to act as a partition key, and this key will determine which data is placed into each partition. Data from insert operations is automatically placed into the appropriate partitions, so applications do not need to be rewritten to take advantage of partitioning.

All operations you perform on regular tables can be performed on individual partitions of a partitioned table. For example, you can export or back up single partitions of a table to avoid exporting or backing up the entire table in one operation. You can also perform the ANALYZE command on each partition concurrently to compute statistics needed for cost-based optimization more quickly.

Each partition of a partitioned table or index operates independently. Therefore, operations on one partition are not affected by the availability of other partitions. If one partition becomes unavailable because of a disk crash or administrative operations, both query and DML operations on data in other partitions can still continue.

Greater Control Over Data

Administrators can specify storage attributes for each partition and the placement of the partition within the host file system, increasing the granularity of control for very large databases. Partitions can be individually taken off-line or brought on-line, backed up, recovered, exported and imported, and loaded—thereby limiting the time required for management operations. An individual index partition can be built for one table partition, reducing the time required for index maintenance operations. Partition operations can be performed in parallel. Partitions increase availability by isolating media and application failures—applications not requiring data from an inaccessible partition continue to run without impact.

Easy-To-Use Administration

The Oracle8 Enterprise Edition provides a set of easy-to-use management commands for partitions. For example, you can implement a table containing a rolling time window of data with a partitioned table so that only one year's worth of data is ever contained in the table. To accomplish this, just add new partitions to the end of the table and drop partitions containing data more than a year old. You can also split and merge partitions easily to balance data among partitions. Also, stand-alone tables that use Oracle7 Release 7.3 UNION ALL views can be converted into partitions of a single table. This allows users with very large Oracle7 databases to quickly benefit from Oracle8 Enterprise Edition partitioning without having to rebuild large tables.

Improved Data Warehouse Performance

The Oracle8 Enterprise Edition introduces new features that improve data warehousing performance:

- Enhanced Star-Query Processing
- New Parallel Operations
- Increased Database Size

Enhanced Star-Query Processing

The Oracle8 Enterprise Edition introduces performance improvements to the processing of star queries, which are common in data warehouse applications. A star query, or star schema, occurs when one or more very large tables, often called fact tables, have relationships to multiple smaller tables called dimension tables.

Oracle7 introduced the functionality of star query optimization, which provides performance improvements for these types of queries. In the Oracle8 Enterprise Edition, however, star-query processing has been improved to provide even better optimization for star queries.

In the Oracle8 Enterprise Edition, a new method for executing star queries has been introduced. Using a more efficient algorithm, and utilizing bitmapped indexes, the new star-query processing provides a significant performance boost to data warehouse applications.

The Oracle8 Enterprise Edition has superior performance with several types of star queries, including star schemas with "sparse" fact tables where the criteria eliminate a great number of the fact table rows. Also, when a schema has multiple fact tables, the optimizer efficiently processes the query. Finally, the Oracle8 Enterprise Edition can efficiently process star queries with large or many dimension tables, unconstrained dimension tables, and dimension tables that have a "snowflake" schema design.

The Oracle8 Enterprise Edition's star-query optimization algorithm, unlike that of Oracle7, does not produce any Cartesian-product joins. Star queries are now processed in two basic phases. First, the Oracle8 Enterprise Edition retrieves exactly the necessary rows from the fact table. This retrieval is done via bit mapped indexes and is very efficient. The second phase joins this result set from the fact table to the relevant dimension tables. This allows for better optimizations of more complex star queries, such as those with multiple fact tables. The new algorithm uses bit-mapped indexes, which offer significant storage savings over previous methods that required concatenated column B-tree indexes. The new algorithm is also completely parallelized, including parallel index scans on both partitioned and non-partitioned tables.

For more information, see *Oracle8i Concepts*.

New Parallel Operations

Insert, update, and delete operations can now be run in parallel in the Oracle8 Enterprise Edition. These operations, known as parallel DML, are executed in parallel across multiple processes. By having these operations execute in parallel, the statement will be completed much more quickly than if the same statement were executed in a serial fashion. Parallel DML complements parallel query by providing parallel transaction execution as well as queries. Parallel DML is useful in a decision support (DSS) or data warehouse environment where bulk DML operations are common. However, parallel DML operations can also speed up batch jobs running in an OLTP database.

The Oracle8 Enterprise Edition supports parallel inserts, updates, and deletes into partitioned tables. It also supports parallel inserts into non-partitioned tables. The parallel insert operation on a non-partitioned table is similar to the direct path load operation that is available in Oracle7. It improves performance by formatting and writing disk blocks directly into the datafiles, bypassing the buffer cache and space management bottlenecks. In this case, each parallel insert process inserts data into a segment above the high watermark of the table. After the transaction commits, the high watermark is moved beyond the new segments.

To use parallel DML, it must be enabled prior to execution of the insert, update, or delete operation. Normally, parallel DML operations are done in batch programs or within an application that executes a bulk insert, update, or delete. New hints are available to specify the parallelism of DML statements.

For more information, see *Oracle8i Parallel Server Concepts and Administration*.

Increased Database Size

The Oracle8 Enterprise Edition can manage databases of hundreds of terabytes in size because of partitioning, administrative improvements, and internal enhancements. Many size limitations in earlier versions of Oracle have been raised, such as the number of columns per table, the maximum database size, and the number of files per database.

Improvements for OLTP Applications

Demanding OLTP applications benefit from a number of new features which improve scalability, performance, and manageability.

Extended Backup/Recovery Subsystem

Oracle8 and the Oracle8 Enterprise Edition's server-managed backup and recovery provides a better backup and recovery functionality integrated within the Oracle database server. Detailed information is maintained on when backups are performed, exactly which parts of the database are backed up, and where the files are stored. Should a recovery be necessary, Recovery Manager analyzes the state of the database and determines the operations necessary to repair the database. It then automatically performs those operations, greatly simplifying the recovery for the administrator and reducing the possibility of human error. A simple GUI interface within Oracle Enterprise Manager controls backup and recovery. An API is also available for third parties who may wish to provide an alternative interface. Media management layer interfaces to popular third-party tape management

products are available. Legato Storage Manager is provided free with both Oracle8 and the Oracle8 Enterprise Edition.

The Oracle8 Enterprise Edition provides multilevel, incremental backups that greatly reduce the size of the backups, because only the changed blocks are backed up. This can substantially reduce the time required to back up a datafile.

Tablespace point-in-time recovery allows one or more tablespaces to be recovered to an earlier time, while the remainder of the database is up and running. This allows many types of user errors to be easily corrected. For example, if a user runs a batch job that incorrectly updates many records in a table, the table can be restored to a time previous to the batch job. Also, if a table is accidentally dropped or truncated, it can be restored to a time before this operation.

For more information, see *Oracle8i Backup and Recovery Guide*.

Very Large User Populations

Numerous enhancements throughout the Oracle8 and Oracle8 Enterprise Edition database servers and Net8 increase the utilization of operating system and networking resources. Connection pooling temporarily drops the physical connection for idle users (and transparently re-establishes the connection when needed), thus increasing the number of users that can be supported. Oracle Connection Manager can be used to configure a middle tier that manages the connections of very large user populations. By configuring multiple connection managers, the Oracle8 Enterprise Edition can support tens of thousands of concurrent users. Shared database links multiplex many users into a database server with a single connection, reducing resource requirements, especially for multitier application architectures.

Advanced Queuing

The Oracle8 Enterprise Edition Advanced Queuing feature supports database messaging, or queuing, through a set of queue tables and queue functions.

Advanced Queuing adds direct support in the database for deferring transaction execution to a later time and executing transactions in a particular order. This capability allows you to decouple distributed applications and eliminates dependency on external systems for applications requiring high scalability. Enqueue and dequeue operations can be used to shift processing from within a transaction to a background process, thereby improving transaction response time. Also, queuing can be used to implement work flow applications that move data to a system as the state of the data changes. An example of this is moving orders from

an order-entry application to a shipping application, and then to a billing system during the life of an order. Advanced Queuing can also work in conjunction with popular TP monitor queuing systems.

Messages can be enqueued and dequeued by applications or other queues. The propagation feature enables applications to communicate with each other without having to be connected to the same database or to the same queue. Using the familiar database links and Net8, messages can be propagated from one queue to another, irrespective of whether these are local or remote.

For more information, see *Oracle8i Application Developer's Guide - Advanced Queuing*.

Parallel Server Improvements

Numerous enhancements in Oracle Parallel Server improve performance, scalability, memory usage, and availability. A common and integrated distributed lock manager replaces the lock managers provided by the different operating system vendors, improving performance and portability on most platforms. Several performance enhancements make the Oracle Parallel Server significantly faster. Also, new global V$ tables improve manageability.

Oracle Parallel Server now uses its own integrated distributed lock manager (DLM) for processing inter-node requests for resources. Previously, Oracle relied on operating system vendors to supply DLM capability for the Parallel Sever to run on a given platform. Integrating the DLM within the Oracle8 Enterprise Edition allows the Oracle Parallel Server to be available on platforms that previously did not support cross-node sharing of resources.

The Oracle8 Enterprise Edition introduces numerous improvements to Oracle Parallel Server performance. System change number (SCN) generation is now optimized for providing SCNs across instances more efficiently. This improvement alone can amount to a 10 to 15 percent improvement in Oracle Parallel Server performance. Also, the DLM now caches locks to avoid cross-node communication of lock information. Pinging, or contention for blocks across nodes, is also reduced by improving the algorithm for controlling access to contended blocks. Reverse-key indexes reduce "hot spots" in indexes, especially primary key indexes, by reversing the bytes of the leaf blocks and thus eliminating the contention for leaf blocks across instances. Partitions also help Parallel Server performance by allowing database administrators to map partitions to specific nodes, which can dramatically reduce pinging. Finally, the Oracle8 Enterprise Edition reduces the memory overhead associated with fine-grain locking in an Oracle Parallel Server.

The Oracle8 Enterprise Edition allows designation of groups of instances for parallel query or parallel DML processing. You can specify to which group an

instance belongs and then use the group to process statements from certain applications. This is especially useful for separating OLTP from data warehouse processing among your clustered servers. Using separate instances in an Oracle Parallel Server ensures that data warehouse queries do not affect the performance of OLTP applications.

The Oracle8 Enterprise Edition introduces global fixed views (GV$) for the Oracle Parallel Server. This allows administrators to log into one instance of a Parallel Server and view global views that contain data from all the V$ views in the cluster. This makes administration operations significantly simpler and improves the productivity of database administrators.

Transparent Application Failover (TAF)

Should a node in the Oracle Parallel Server fail, transparent application failover will migrate your connections and automatically re-establish their sessions on another node. Your applications will continue to run, and you may be unaware of the failure. This provides continuous availability in the event of scheduled and unscheduled outages. Even if you are not using the Oracle Parallel Server option, TAF will automatically reconnect and reestablish your session.

Depending on the amount of overhead you are willing to incur on the client and backup machines, the instance failure could result in a completely transparent migration of user connections to the failover node. You can specify that you want all queries to be cached on the client so that they can be reinstantiated on the failover node. Also, you can have a session pre-connected to the failover node, eliminating the time required to reconnect to a failover instance.

Transparent application failover is useful not only for availability, but also for manual load-balancing or orderly shutdown of the system. If too many users connect to an instance, you can terminate their sessions and have them transparently migrate to another node. You can also shut down a node and have users transparently migrate to a failover node after their current transaction completes. Shutdowns no longer need to interrupt users work.

Note: To take advantage of this functionality, applications must be written specifically using new Oracle8 OCI calls.

For more information, see *Oracle8i Administrator's Guide*.

Improved TP Monitor Support

Dynamic XA support improves performance for multitier applications with industry-standard XA-compliant Transaction Processing (TP) monitors. Oracle8 and

the Oracle8 Enterprise Edition improve support of the XA interface with support for dynamic registration and loosely-coupled transaction branches. They also offer better performance and recovery when used with Oracle Parallel Server.

For more information, see *Oracle8i Application Developer's Guide - Fundamentals* or *Oracle8i Parallel Server Concepts and Administration*.

Better Security Administration

Oracle8 and the Oracle8 Enterprise Edition include a Security Server which provides a single sign-on environment for centrally administering users and roles. Oracle Security Server is compliant with the X.509 certificate-based security standard for public/private key authentication. Mutual authentication between client and server is also supported for protection against "rogue" databases designed to capture client communication. Also, a digital signatures tool kit is provided for creating applications designed to identify unauthorized tampering with data.

Oracle8 and the Oracle8 Enterprise Edition also contain improved password maintenance and administration. You can now define a password profile to enforce a security scheme. Passwords can expire after a certain time, or be checked for complexity (e.g., minimum length). You can define your own policy or use the standard stored function for checking the length, content, or reuse of new passwords. Also, you can create user accounts so that the user must immediately change the password upon the first access to the system.

Oracle8 and the Oracle8 Enterprise Edition also add support for privileged database links, which make it unnecessary to embed a password in a database link, and data encryption services callable from OCI and PL/SQL.

Replication

Oracle8 and Oracle8 Enterprise Edition replication provides improved support for mass-deployment distributed systems, higher throughput failover configurations, and specialized data warehouse systems.

Increased replication performance is now possible. Changes to replica sites can be propagated and applied in parallel, effectively removing the throughput limit on replication, while maintaining transactional integrity. Also, much of the replication functionality has been rewritten and moved from PL/SQL triggers into C code inside the Oracle engine, providing a significant boost in performance. Finally, the amount of data sent over the network for each replicated transaction has been reduced, and network round-trips are kept to a minimum.

Salesforce automation and other mass deployment systems are now easier to administer and design with the subsetting capabilities of both Oracle8 and the Oracle8 Enterprise Edition. Complex subquery snapshots can now include a fast refresh clause which allows you to build a snapshot of select rows based on a query to another table. For example, you can build a snapshot for each salesman based on the ORDERS table. That snapshot can contain only the orders by selecting the orders based on criteria from an ASSIGNMENTS table. Each salesman sees only his orders and only has to interact with the master table when uploading new orders.

Oracle8 Enterprise Edition replication also provides numerous improvements in manageability and ease of use. Significant enhancements to Oracle Replication Manager, including wizards, make replicated environments easier to set up and maintain. Additional improvements to replication manageability include snapshot registration, which provides valuable information about which sites have associated snapshots; primary key snapshots, which allow faster snapshot refresh after a table reorganization; a new security algorithm, which is easier to administer; and fine-grain quiesce, which allows an administrator to make schema-level changes to one replication group while others continue to operate.

For further information, see *Oracle8i Replication*.

Object-Relational Technology

The Oracle8 Enterprise Edition makes a major leap in data management technology with the introduction of an object-relational paradigm. Database schemas and applications today are becoming increasingly complex. Often, several separate applications with similar data, such as customer information, billing, and shipping, exist in different database schemas and an MIS department must manage the interoperation. Corporate management of the information becomes a difficult task of integrating different relational objects and different applications, possibly from different vendors, into a more coherent end-user data model. By enhancing the relational database with object extensions, Oracle addresses the need to simplify data modeling and extend the database with new datatypes.

The new, object-relational features include the following:

- Object Types and Views
- Calling External Procedures From Within the Database
- Client-Side Support for Objects
- Evolution of Relational Environments
- Development Tools for Object Modeling

- Multimedia Data
- Java
- Extensibility

Object Types and Views

Object types provide a way to extend Oracle's relational datatype system. Relational databases support three datatypes: characters, numbers, and dates. Object types allow you to define new datatypes and use them as you would regular relational datatypes. For example, you can create a new type called Address. This object type can have data, called attributes, such as Street, City, and Postal Code. The object type can also have methods or stored procedures, such as Distance, for computing the distance between addresses. These methods can be written in either PL/SQL or C. An address can then be used anywhere a regular datatype could, whether in column definitions, in PL/SQL variables, or even as the definition for an object table.

Oracle's object types can use powerful object modeling techniques for complex objects. For example, you can represent collections of similar objects in array structures or nested tables. You can also store references to objects for fast traversal without joining tables.

Object types allow application developers to code application logic in the database or the middle-tier application server, as opposed to using client-side code. All applications can then share the logic of the new datatypes so developers do not need to rewrite the code. This feature provides the advantages of creating re-usable code components and transparent application partitioning so that the code can reside and execute on the tier that will yield the best performance: client, application server, or database server.

The Oracle8 Enterprise Edition follows the emerging SQL3 standard for object type definition and object modeling techniques. SQL3 defines syntax for creating and modifying object types, generating and storing object identifiers (OIDs), creating references or pointers to objects, and modeling collections of similar objects.

For more information, see *Oracle8i Concepts* and *Oracle8i Application Developer's Guide - Fundamentals*.

Calling External Procedures From Within the Database

Oracle8 and the Oracle8 Enterprise Edition provide a safe, fast way for the database to make a call to an external program. The call also can be made through open

protocols like HTTP or IIOP (a CORBA standard). External procedures allow you to use existing application code, or write highly-optimized code for specific purposes, such as a computationally complex algorithm like Fast Fourrier Transform (FFT). Also, you can use external procedures to interface with other applications or with specialized devices like embedded systems.

For more information, see *PL/SQL User's Guide and Reference* and *Oracle8i Application Developer's Guide - Fundamentals*.

Client-Side Support for Objects

The client-side object cache allows user applications to retrieve a complex hierarchy of objects into an application cache. The application can then traverse the objects without performing additional network retrievals. This provides a convenient and fast way to use objects in a client application and write code that is more like the native object-oriented code.

A new utility, the Object Type Translator, has been introduced as a quick way of generating header and implementation files for applications running against object schemas.

Evolution of Relational Environments

The Oracle8 Enterprise Edition is designed to allow users to easily evolve into using the new object-oriented functionality, as all existing applications are upwardly compatible. The new object-relational extensions are built on the same foundation as the relational functionality, which means that users do not have to discard or rewrite their existing relational applications before migrating to the Oracle8 Enterprise Edition. Unlike other object-relational databases, this design allows the older relational applications, which still read and write rows and columns, to coexist with new object-oriented applications, which read and write objects. The Oracle8 Enterprise Edition provides object views to retrieve relational data and represent the data to a client as if it were an object and vice-versa.

For example, an existing relational order-entry system might need a new front-end for the World Wide Web. The existing applications accessing the relational schema can remain in operation, and a new set of object views can be developed as an object representation for the client. New and old applications can be based on the same data, but each has its own representation.

For more information, see *Oracle8i Concepts*.

Development Tools for Object Modeling

The Object Database Designer offers object support in a design tool. It helps you design, create, and access object-based Oracle8 Enterprise Edition systems.

For more information, see *Object Database Designer.*

Multimedia Data

Large Objects (LOBs) handle unstructured data such as images, sounds, video, and text, and have much richer functionality than their predecessors, LONG and LONG RAW. Character LOBs (CLOB or NCLOB), Binary LOBs, and BFILES (externally stored LOBs), can be replicated and can be an attribute of an object. You can also have more than one LOB per table. LOBs also have a greater maximum size than LONGs and have different mechanisms for maintaining read-consistency and random access.

LOB data is indexed for fast access starting at a specified byte. For example, you can read / write at specific byte-offsets. You can also read / write LOBs through the Oracle8 buffer cache or access them directly from disk.

LOB functionality is available with both Oracle8 and the Oracle8 Enterprise Edition and does not require the Objects option to be licensed and installed.

For more information, see *Oracle8i Concepts.*

Java

Oracle has a long history of supporting cross-platform environments. Oracle8 and the Oracle8 Enterprise Edition build upon that with strong support for Java.

Oracle currently has two methods for accessing Oracle data from Java programs: an Oracle-provided JDBC driver integrated with Oracle's object types, and JSQL for embedding SQL statements into Java code. Oracle provides its own JDBC drivers for better performance. JSQL allows you to include SQL statements in a Java application. The JSQL precompiler then converts the SQL into JDBC calls, which allows you to use existing SQL code in new Java applications.

Oracle8i JDBC Developer's Guide and Reference

Extensibility

Extensibility allows you to define your own datatypes. You can extend the capabilities of your current database by creating new datatypes for your specific applications. These new datatypes can be used in the same manner as the ones you

have now; the same operations can be performed. Image, Spatial, Time Series, Visual, and ConText cartridges are now available as extensions to the Oracle database server.

Migration and Interoperability

A simple and fast migration utility rebuilds the data dictionary and converts the control files, log files, and data blocks. The migration utility converts any Oracle 7.1, 7.2, or 7.3 database into an Oracle8 or Oracle8 Enterprise Edition database. Oracle7 applications run unchanged against either of the Oracle8 products. Distributed commands from either of the Oracle8 products run against Oracle7, and vice-versa.

For more information, see *Oracle8i Migration*.

Other Enhancements

Index-Organized Tables

Index-organized tables store the data columns of a table within the leaf nodes of the Oracle8 B-tree index structure. This reduces overall storage requirements when most columns are indexed by storing the columns only once, rather than in both an index and a separate table. Index-organized tables also reduce access time by retrieving all columns from one location instead of two.

For more information, see *Oracle8i Concepts*.

Reverse Key Indexes

Reverse key indexes reduce the "hot spots" in indexes, especially ascending indexes. Unbalanced indexes can cause the index to become increasingly deep as the base table grows. Reverse key indexes reverse the bytes of leaf-block entries, therefore preventing "sliding indexes".

For more information, see *Oracle8i Concepts*.

Improved Constraint Processing

Constraint processing has been dramatically improved. You can now use non-unique indexes to enforce UNIQUE and PRIMARY KEY constraints. This eliminates redundant indexes and permits indexes to remain valid while constraints are disabled. Deferred-constraint checking shifts integrity-constraint checking from

the end of statement execution to when a commit is issued. This simplifies the coding of certain operations involving integrity constraints. Also, all constraints can be enabled concurrently and in parallel with other constraints, and permit concurrent DML while the enabling continues.

Two Character Sets in One Database

In both Oracle8 and the Oracle8 Enterprise Edition, a new NCHAR datatype allows a second character set in one database. This improves performance and storage predictability for some Asian language, multibyte character set databases. Also, fixed-length native Unicode 2.0 and Chinese GBK support is new.

Miscellaneous Improvements

A non-updatable view (i.e., a joined view) can be updatable through the use of a new type of trigger, the INSTEAD OF trigger. This trigger allows you to replace INSERT, UPDATE, and DELETE operations on views with your own logic so that even views based on complex joins can be modified. Also, you can place subqueries in the select list of other queries to return result sets from an arbitrary list of detail tables without performing join operations. By placing a cursor expression in a select list, you can fetch rows from detail tables in a 3GL program without specifying complex join conditions in the FROM clause.

4

Oracle8*i* Feature and Option Availability

This chapter provides the option and feature factoring matrixes for Oracle8*i*. They identify the options and features for this release and identify the product configurations for which they are available (feature factoring).

The following topics are included in this chapter:

- Options
- Features
- The V$OPTION Table

Unlike prior releases, all features and options applicable to your configuration are included on your product CD-ROM and are shipped enabled. *Options must be licensed for use.*

Options

The following table lists the options available with this release. Options which are new, or whose factoring status has changed, are noted in *bold italic* print.

Many options, not marked as new, have undergone significant enhancement. Descriptions of these enhancements, as well as the new options, are contained in Chapter 2.

Table 4–1 Oracle8i Option Factoring **(Page 1 of 3)**

Option	Oracle8*i*	Oracle8*i* EE	Oracle8*i* Work-station	Notes
Oracle Advanced Security	*N*	*Y*	*Y*	*Advanced Security Option provides a single source of integration with network encryption, single sign-on services, smartcard, token and biometric user authentication.*
Oracle Change Management Pack	*N*	*Y*	*Y*	*Oracle Change Management Pack eliminates errors and loss of data when upgrading databases to support new applications. The pack analyzes the impact and complex dependencies associated with application change and automatically performs database upgrades.*
Oracle Diagnostics Pack	*N*	*Y*	*Y*	*This is an advanced set of tools used to monitor the state of databases and systems and pinpoint, analyze, and repair any weaknesses.*
Oracle interMedia	*Y*	*Y*	*Y*	*interMedia enables Oracle8i to manage text, documents, images, audio, and video in an integrated fashion with other enterprise information.*
Oracle JServer	*Y*	*N*	*N*	*Includes Java stored procedures, methods, and triggers; EJB, CORBA, IIOP, and HTTP support.*
Oracle JServer Enterprise Edition	*N*	*Y*	*Y*	*Includes Java stored procedures, methods, and triggers; EJB, CORBA, IIOP, and HTTP support*
Oracle JServer Enterprise Edition Accelerator	*N*	*Y*	*Y*	*The Native Compiler allows Java programs to be compiled into C code, resulting in better performance and scalability. Not available in Oracle8i Release 8.1.5*

Table 4–1 Oracle8i Option Factoring **(Page 2 of 3)**

Option	Oracle8*i*	Oracle8*i* EE	Oracle8*i* Work-station	Notes
Oracle Parallel Server	N	Y	N	Oracle Parallel Server allows database workload to be distributed across a cluster, resulting in high scalability. High availability is assured because if one node fails, the remaining nodes still can access the database.
Oracle Partitioning	N	Y	Y	Partitioning allows large tables to be broken into smaller pieces that improve manageability, availability, and scalability. No CREATE PARTITION statements are possible without this option.
Oracle Programmer	Y	Y	Y	It is a family of products consisting of three SQL-style interfaces: precompilers, SQL*Module, and SQLJ; three call level interfaces: Oracle Call Interface (OCI), ODBC, and JDBC; Oracle Objects for OLE; and Object Type Translator and JPub.
Oracle Spatial	N	Y	Y	Oracle8*i* Spatial is an integrated set of functions and procedures that enables spatial data to be stored, accessed, and analyzed quickly and efficiently in an Oracle database.
Oracle Standard Management Pack	*Y*	*N*	*Y*	*A complete set of database tuning, diagnostic, change management, and other manageability technologies for Oracle8i. This is an extra cost option.*
Oracle Time Series	*Y*	*Y*	*Y*	*Manages time-based data and provides functionality to do time-based analysis on the data.*

Table 4–1 Oracle8i Option Factoring (Page 3 of 3)

Option	Oracle8*i*	Oracle8*i* EE	Oracle8*i* Work-station	Notes
Oracle Tuning Pack	*N*	*Y*	*Y*	*Oracle Tuning Pack provides database administrators with expert performance management for the Oracle environment, including SQL tuning and storage optimization.*
Oracle Visual Information Retrieval	N	Y	Y	Visual Information Retrieval (VIR) functionality from Virage for advanced image searching and management.
Oracle WebDB	*Y*	*Y*	*Y*	*The Oracle WebDB is a complete and cost-effective solution for building, deploying, and proactively monitoring Web database applications and content-driven Web sites.*

Features

The following table lists the features available with this release. Features which are new, or whose factoring status has changed, are noted in *bold italic* print.

Many features, not marked as new, have undergone significant enhancement. Descriptions of these enhancements, as well as the new features, are contained in Chapter 2.

Table 4–2 Oracle8I Feature Factoring (Page 1 of 10)

Area	Feature	Oracle8*i*	Oracle8*i* EE	Oracle8*i* Work-station	Notes
Data Warehouse/ VLDB	*Automated parallel query degree*	*N*	*Y*	*Y*	*Automated parallel query optimization based on system utilization.*
Data Warehouse/ VLDB	*CUBE and ROLLUP*	*Y*	*Y*	*Y*	*OLAP operators CUBE and ROLLUP to produce sub-totals and crosstab reports easily and efficiently using a single SQL statement.*

Table 4–2 Oracle8I Feature Factoring **(Page 2 of 10)**

Area	Feature	Oracle8*i*	Oracle8*i* EE	Oracle8*i* Work-station	Notes
Data Warehouse/ VLDB	*Descending indexes*	*Y*	*Y*	*Y*	*Provides better performance when indexed data needs to sorted in descending order.*
Data Warehouse/ VLDB	*Direct Path Load API*	*Y*	*Y*	*Y*	*Direct Load API allows complete access to all load functionality via OCI API for creation of high performance load programs.*
Data Warehouse/ VLDB	*Export transportable tablespaces*	*N*	*Y*	*Y*	*Functionality to quickly move or create a copy of tablespace data. A transportable tablespace can be "plugged in" to all editions of the Oracle8i database, but only the Enterprise Edition can create a transportable tablespace.*
Data Warehouse/ VLDB	*Function-based indexes*	*N*	*Y*	*Y*	*Allows indexes to be created on expressions or functions.*
Data Warehouse/ VLDB	*Import transportable tablespaces*	*Y*	*Y*	*Y*	*Functionality to quickly "plug in" a transportable tablespace. Oracle8i Enterprise Edition must be used to create (export) a transportable tablespace.*
Data Warehouse/ VLDB	*Long operations monitor*	*Y*	*Y*	*Y*	*Progress of long running database and application operations can be monitored and information can be displayed, such as percent complete.*
Data Warehouse/ VLDB	*Materialized views*	*N*	*Y*	*Y*	*Includes summary management functionality, join indexes, and summary advisor. Powerful functionality that automatically redirects queries to stored summary tables, improving query performance dramatically.*
Data Warehouse/ VLDB	*Optimizer statistics management*	*Y*	*Y*	*Y*	*Schema object statistics, used by the cost based optimizer, can be copied and moved to another database.*

Table 4–2 Oracle8I Feature Factoring **(Page 3 of 10)**

Area	Feature	Oracle8*i*	Oracle8*i* EE	Oracle8*i* Work-station	Notes
Data Warehouse/ VLDB	*Parallel load*	*Y*	*Y*	*Y*	*Fast load of data accomplished by use of multiple, parallel processes.*
Data Warehouse/ VLDB	*Sample scan*	*N*	*Y*	*Y*	*The SAMPLE keyword is used to in SQL statements to easily select a percentage of random data within a table; this is useful in data mining applications.*
Data Warehouse / VLDB	Bitmap indexes	N	Y	Y	Index type commonly used in a data warehouse for columns with small number of distinct values, such as 'Y' or 'N', resulting in dramatic performance gains in a data warehouse application.
Data Warehouse / VLDB	Parallel analyze	N	Y	Y	The analyze command, used to gather statistics on tables, can be run using parallel processing. This feature requires the Partitioning Option.
Data Warehouse / VLDB	Parallel bitmap star query optimization	N	Y	Y	This algorithm utilizes single-table bitmap indexes and an advanced star query join method, resulting in excellent performance while efficiently utilizing space.
Data Warehouse / VLDB	Parallel DML (insert / update / delete)	N	Y	Y	DML operations are transparently divided across multiple processes. This feature requires the Partitioning Option.
Data Warehouse / VLDB	Parallel index build	N	Y	Y	Indexes can be created using parallel processes, significantly improving performance of the operation.
Data Warehouse / VLDB	Parallel index scans	N	Y	Y	Queries that use of an index can scan the index in parallel with multiple processes. This requires the Partitioning Option

Table 4–2 Oracle8I Feature Factoring **(Page 4 of 10)**

Area	Feature	Oracle8*i*	Oracle8*i* EE	Oracle8*i* Work-station	Notes
Data Warehouse / VLDB	Parallel query	N	Y	Y	Oracle8*i* transparently distributes query execution across multiple processes, resulting in excellent performance.
Data Warehouse / VLDB	Star query optimization	Y	Y	Y	This star-query optimization algorithm utilizes B-tree indexes (not bitmap indexes)
Database features	Advanced queuing	N	Y	Y	Rules-based, publish and subscribe queuing system that can be used to develop large-scale, message-oriented distributed applications.
Database features	*Database event triggers*	*Y*	*Y*	*Y*	*Database triggers, such as database startup or shutdown triggers, DDL statement triggers, and logon/logoff triggers are fired when the event occurs.*
Database features	*DBMS_REPAIR package*	*Y*	*Y*	*Y*	*Provides early detection and correction of software and hardware corruption.*
Database features	*Drop column*	*Y*	*Y*	*Y*	*Functionality to remove an unwanted column from a table.*
Database features	*Fine-grained access control*	*N*	*Y*	*Y*	*Includes functionality to create security policies that implement row-level security.*
Database features	*Index coalesce*	*N*	*Y*	*Y*	*Free space in index leaf blocks can be defragmented while table is online.*
Database features	Index-organized tables	Y	Y	Y	Table where data is stored in a B*-tree index structure for better performance and reduced storage.
Database features	*Indexes on NLS collating sequences*	*N*	*Y*	*Y*	*Efficient linguistic sorting using NLS sorting through use of function-based indexes (available in Enterprise Edition and Personal Oracle8i).*

Table 4–2 Oracle8I Feature Factoring **(Page 5 of 10)**

Area	Feature	Oracle8*i*	Oracle8*i* EE	Oracle8*i* Work-station	Notes
Database features	Instead-of triggers	N	Y	Y	Triggers that execute instead of the DML transaction execution.
Database features	LOB (large object) support	Y	Y	Y	Datatypes and functionality for storing and manipulating large objects, such as images or other unstructured data.
Database features	*Locally-managed tablespaces*	*Y*	*Y*	*Y*	*Efficient storage management functionality that improves reliability and reduces fragmentation.*
Database features	*LogMiner*	*Y*	*Y*	*Y*	*LogMiner provides the functionality to analyze transaction log information online to enable auditing of transactions and the ability to "back out" individual transactions.*
Database features	National language support (NLS)	Y	Y	Y	Native language support
Database features	*Objects and extensibility*	*Y*	*Y*	*Y*	*Object relational functionality, including object types and methods. Also, database extensibility via APIs to database services and functionality.*
Database features	*Online index build*	*N*	*Y*	*Y*	*Indexes can be built and re-built without locking the table during the operation.*
Database features	Password management	Y	Y	Y	Provides password expiration, complexity, and security policy management.
Database features	PL/SQL stored procedures, triggers	Y	Y	Y	Provides mechanism to deploy logic directly in the database.

Table 4–2 Oracle8I Feature Factoring **(Page 6 of 10)**

Area	Feature	Oracle8*i*	Oracle8*i* EE	Oracle8*i* Work-station	Notes
Database features	*Plan Stability*	*N*	*Y*	*Y*	*This allows execution plans for SQL to be stored so that the plan remains consistent throughout schema changes, database reorganizations, and data volume changes.*
Database features	Reverse key indexes	Y	Y	Y	Indexed columns can be indexed on reversed column values for better index performance.
Database features	*Temporary tables*	*Y*	*Y*	*Y*	*Temporary tables to allow users to manipulate data for the duration of a transaction or session.*
Distributed	Advanced Replication	N	Y	Y	Advanced replication includes basic replication functionality plus multi-master replication, Replication Manager, and parallel propagation to maximize throughput.
Distributed	*Basic replication*	*Y*	*Y*	*Y*	*Oracle8i server fully supports bi-directional replication with automated conflict detection and resolution. Supported configurations include a single updatable master site with multiple updatable or read-only snapshot sites.*
Distributed	Distributed queries	Y	Y	Y	Queries can include tables from multiple databases.
Distributed	Distributed transactions	Y	Y	Y	Transactions can include tables from multiple databases. Includes transparent two phase commit functionality and XA support.
Distributed	Heterogeneous services	Y	Y	Y	Queries and transactions can involve Oracle and non-Oracle databases transparently. This functionality is used by Oracle Transparent Gateways.

Table 4–2 Oracle8I Feature Factoring **(Page 7 of 10)**

Area	Feature	Oracle8*i*	Oracle8*i* EE	Oracle8*i* Work-station	Notes
Networking	*N-tier authentication/authorization*	*N*	*Y*	*Y*	*N-tier authentication functionality that preserves client identity through all tiers.*
Networking	Network Access Control	N	Y	Y	Provides filtering capabilities between clients and servers and acts as relay for thin JDBC connections that have no access privileges to the database host
Networking	Connection Pooling	Y	Y	Y	Connection pooling enables the server to time-out idle sessions and use that connection to service an active session, resulting in more efficient network usage.
Networking	Multi-protocol connectivity	N	Y	Y	Bridges Net8 communities of users that use different network protocols such as SPX/IPX, TCP/IP or LU6.2
Networking	Multiplexing	N	Y	Y	Multiplexing enables multiple network sessions to coexist over a single physical transport, reducing the number of physical connections a server must maintain to support a population of clients.
Networking	Net8	Y	Y	Y	Net8 enables heterogeneous, distributed computing across machines regardless of vendor, operating system, or hardware architecture.
Networking	Oracle Connection Manager	N	Y	Y	Multiplexing, Multi-protocol connectivity, Network Access Control.
Networking	Oracle Names	Y	Y	Y	Provides fully integrated, multi-protocol network naming service that allows administrators to centrally define service addresses, inter-database links, aliases and client configuration profiles.

Table 4–2 Oracle8I Feature Factoring **(Page 8 of 10)**

Area	Feature	Oracle8*i*	Oracle8*i* EE	Oracle8*i* Work-station	Notes
Program Interfaces	*AppWizard for Visual Studio (NT only)*	*Y*	*Y*	*Y*	*AppWizard to ease development of Oracle based applications using Visual Studio.*
Program Interfaces	*Autonomous transactions*	*Y*	*Y*	*Y*	*Blocks of PL/SQL can commit independently of each other to ease complex transaction programming.*
Program Interfaces	*COM cartridge (NT only)*	*Y*	*Y*	*Y*	*COM cartridge allows PL/SQL developers to programmatically manipulate COM objects through the OLE Automation interface. Entry points are exposed for access to the methods defined by the OLE Automation interface for easy application integration.*
Program Interfaces	JDBC drivers	Y	Y	Y	JDBC access to Oracle8: Java call-level interface to SQL that is standards compliant and supports Oracle-specific features.
Program Interfaces	*Microsoft Transaction Server Integration (NT only)*	*Y*	*Y*	*Y*	*Full, native integration with Microsoft Transaction Server (MTS) to allow developers to deploy COM-based applications using MTS against Oracle.*
Program Interfaces	Objects for OLE	Y	Y	Y	Oracle Objects for OLE is a COM-based database connectivity tool that provides seamless and optimized access to Oracle databases.
Program Interfaces	ODBC driver	Y	Y	Y	ODBC driver for access to the Oracle database.
Program Interfaces	Oracle Call Interface (OCI)	Y	Y	Y	Lowest level API interface to the Oracle database.
Program Interfaces	Pro*C	Y	Y	Y	Allows C programs to easily use SQL to access data. Requires Oracle Programmer

Table 4–2 Oracle8I Feature Factoring **(Page 9 of 10)**

Area	Feature	Oracle8*i*	Oracle8*i* EE	Oracle8*i* Work-station	Notes
Program Interfaces	*SQLJ*	*Y*	*Y*	*Y*	*Allows embedded SQL statements in Java for concise and easy access to Oracle data. Requires Oracle Programmer.*
System Management	*Automated and readable standby database*	*N*	*Y*	*Y*	*Multiple remote standby databases can easily be configured and automatically managed by the server, including copying and applying the transaction log files at the standby sites. Standby databases can be opened for query-only access.*
System Management	*Database resource management*	*N*	*Y*	*Y*	*Resources, such as CPU and degree of parallelism, can be allocated to groups of users such that a prioritization of tasks can be managed easily.*
System Management	*Duplexed backup sets*	*N*	*Y*	*Y*	*Backup sets can be written to multiple devices in parallel.*
System Management	*Fast-start fault recovery*	*N*	*Y*	*Y*	*Fast-start fault recovery provides fast and predictable recovery from system failures.*
System Management	Incremental backup and recovery	N	Y	Y	This functionality allows a backup process to be set up such that only changed blocks are written to the backup file.
System Management	Legato Storage Manager	Y	Y	Y	Integrated functionality from Legato to allow backups directly to tapes.
System Management	Online backup and recovery	Y	Y	Y	Backups can be run while the database is online. Recovery operations can also be run while the database is running.
System management	Oracle DBA Management Pack	Y	Y	Y	This is an OEM management pack that is available free with all editions of Oracle8*i*. Includes Schema Manager and other tools. Contains no new functionality.

Table 4–2 Oracle8I Feature Factoring **(Page 10 of 10)**

Area	Feature	Oracle8*i*	Oracle8*i* EE	Oracle8*i* Work-station	Notes
System Management	Oracle Enterprise Manager	Y	Y	N	Single, integrated console that allows easy management and monitoring of the Oracle and non-Oracle database environment.
System Management	Oracle Fail Safe for Oracle8 on NT	Y	Y	N	Oracle Fail Safe makes it easy to deploy highly available single-instance Oracle8 database solutions on Microsoft Windows NT clusters.
System Management	Parallel backup and recovery	N	Y	Y	Both backup and recovery can be run using parallel processing.
System Management	Point-in-time tablespace recovery	N	Y	Y	Allows a tablespace to be recovered up to a specified point-in-time after a failure or inadvertent transaction execution.
System Management	*Readable standby database*	*Y*	*Y*	*Y*	*Databases, including a standby database, can be opened as "read-only", where no DML operations are allowed.*
System Management	Recovery Manager	Y	Y	Y	Easy to use, wizards-based functionality to setup and manage the entire backup and recovery process.
System Management	Server managed backup and recovery	Y	Y	Y	The database server manages the backup process, such as scheduling of backups, as well as the recovery process, such as applying the correct backup file when recovery is needed.
System Management	Transparent Application Failover	N	Y	Y	Oracle8*i* Transparent Application Failover (TAF) can completely mask many failures from end-users, by preserving the state of their application and resuming any work that had been in progress at the time of failure.

The V$OPTION Table

To check the values for different options on your database, query the V$OPTION table:

```
SQL> SELECT * FROM V$OPTION;
```

The response will, of course, vary depending upon the specific installation. A sample result might be:

PARAMETER	VALUE
Partitioning	TRUE
Objects	TRUE
Parallel Server	FALSE
Advanced replication	TRUE
Bit-mapped indexes	TRUE
Connection multiplexing	TRUE
Connection pooling	TRUE
Database queuing	TRUE
Incremental backup and recovery	TRUE
Instead-of triggers	TRUE
Parallel backup and recovery	TRUE
Parallel execution	TRUE
Parallel load	TRUE
Point-in-time tablespace recovery	TRUE
Fine-grained access control	TRUE
N-Tier authentication / authorization	TRUE
Function-based indexes	TRUE
Plan Stability	TRUE
Online index segment creation	TRUE
Index segment coalesce	TRUE
Managed Standby	TRUE

PARAMETER	VALUE
Materialized view rewrite	TRUE
Materialized view warehouse refresh	TRUE
Database resource manager	TRUE
Spatial	TRUE
Visual Information Retrieval	TRUE
Export transportable tablespaces	TRUE
Transparent Application Failover	TRUE
Fast Start recovery	TRUE

5

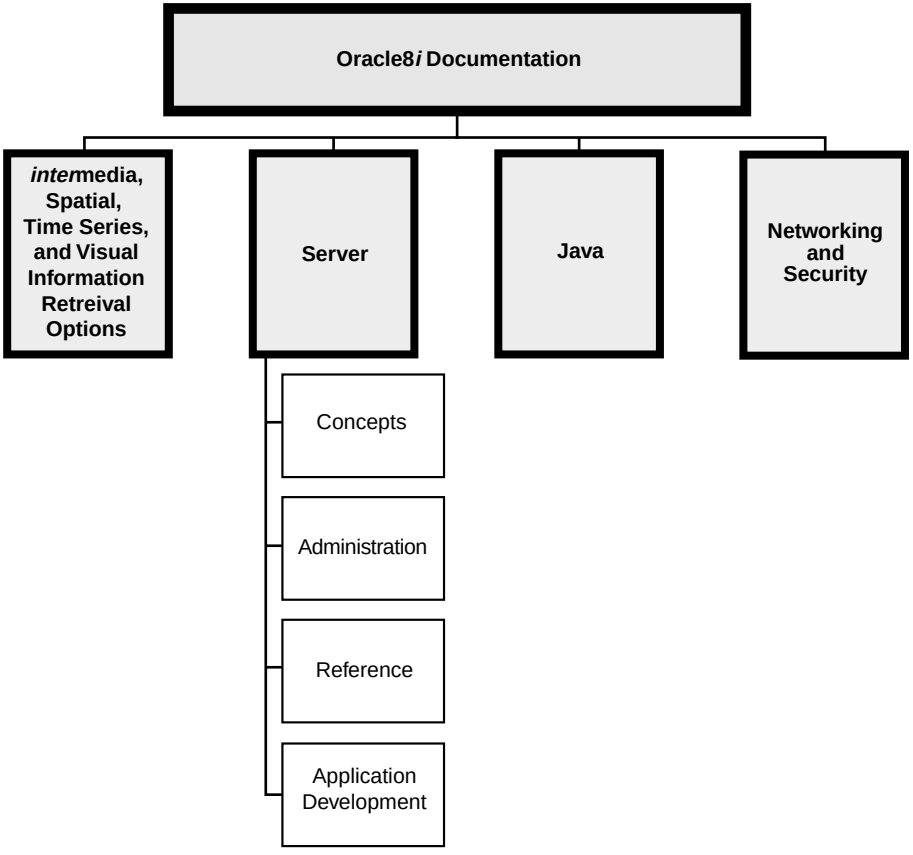
Oracle8*i* Documentation

This chapter lists the generic (non operating-system specific) books, and their part numbers, that document Oracle8*i*, release 8.1.5. Oracle8*i* documentation is available in both online and printed book format, although not all books are available in printed format.

This chapter includes the following sections:

Online Documentation Structure

All books are available in electronic format in the *Oracle8i, Release8.1, Online Generic Documentation* CD-ROM. The following figure shows how the books are organized on this CD.



*inter*Media, Spatial, Time-Series, and Visual Information Retrieval Options

Oracle8i interMedia Text Reference

Oracle8i ConText to interMedia Text Migration

Oracle8i interMedia Audio, Image, and Video User's Guide and Reference

Oracle8i interMedia Audio, Image, and Video Java Client User's Guide and Reference

Oracle8i interMedia Locator User's Guide and Reference

Oracle8i Time Series User's Guide

Oracle8i Visual Information Retrieval User's Guide and Reference

Oracle8i Visual Information Retrieval Java Client User's Guide and Reference

Oracle8i Spatial User's Guide and Reference

Server

Concepts:

Getting to Know Oracle8i

Oracle8i Concepts

Oracle8i Parallel Server Concepts and Administration

Administration:

Oracle8i Administrator's Guide

Oracle8i Backup and Recovery Guide

Oracle8i Distributed Database Systems

Oracle8i Migration

Oracle8i Replication

Oracle8i Tuning

Oracle8i Utilities

Legato Storage Manager Administrator's Guide

Oracle8i Parallel Server Setup and Configuration Guide

Oracle Intelligent Agent User's Guide

Reference:

Oracle8i Reference

Oracle8i SQL Reference

*SQL*Plus User's Guide and Reference*

*SQL*Plus Quick Reference*

Oracle8i National Language Support Guide

Oracle8i Error Messages

Oracle8i Replication API Reference

Oracle SNMP Support Reference Guide

Application Development:

Oracle8i Application Developer's Guide - Fundamentals

Oracle8i Application Developer's Guide - Large Objects (LOBs)

Oracle8i Application Developer's Guide - Advanced Queuing

Oracle8i Supplied Packages Reference

Oracle Call Interface Programmer's Guide

PL/SQL User's Guide and Reference

*Pro*C/C++ Precompiler Programmer's Guide*

*Pro*COBOL Precompiler Programmer's Guide*

Oracle8i Data Cartridge Developer's Guide

*SQL*Module for Ada Programmer's Guide* (Deprecated)

Programmer's Guide to the Oracle Precompilers (Deprecated)

*Pro*Fortran Supplement to the Oracle Precompilers Guide* (Deprecated)

Java

Oracle8i Java Developer's Guide

Oracle8i Java Stored Procedures Developer's Guide

Oracle8i SQLJ Developer's Guide and Reference

Oracle8i JDBC Developer's Guide and Reference

Oracle8i Enterprise JavaBeans and CORBA Developer's Guide

Networking and Security

Net8 Administrator's Guide

Oracle Advanced Security Administrator's Guide

Printed Documentation and Part Numbers

The following tables list the titles and part numbers which comprise the generic documentation sets. An *Oracle8i Generic Documentation Master Index* is also available, but its part number is not listed below.

Oracle8*i* Database Server

The following represents the Oracle8*i* server printed documentation set. It may be ordered by part number A67844-01.

Document Title	Part Number
Getting to Know Oracle8*i*	A68020-01
Legato Storage Manager Administrator's Guide	A68160-01
Oracle Advanced Security Administrator's Guide	A67766-01
Oracle Call Interface Programmer's Guide (2 volume set)	A67846-01
Oracle Call Interface Programmer's Guide, Volume 1	A67848-01
Oracle Call Interface Programmer's Guide, Volume 2	A67847-01
Oracle Net8 Administrator's Guide	A67440-01
Oracle8*i* Administrator's Guide	A67772-01
Oracle8*i* Application Developer's Guide (3 volume set)	A68006-01
Oracle8*i* Application Developer's Guide - Advanced Queuing	A68005-01
Oracle8*i* Application Developer's Guide - Fundamentals	A68003-01
Oracle8*i* Application Developer's Guide - Large Objects (LOBs)	A68004-01
Oracle8*i* Backup and Recovery	A67773-01
Oracle8*i* Concepts (2 volume set)	A67781-01
Oracle8*i* Concepts, Volume 1	A67782-01
Oracle8*i* Concepts, Volume 2	A67783-01
Oracle8*i* Data Cartridge Developer's Guide	A68002-01
Oracle8*i* Distributed Database Systems	A67784-01
Oracle8*i* Error Messages	A67785-01
Oracle8*i* Error Messages, Volume 1	A67787-01

Document Title	Part Number
Oracle8*i* Error Messages, Volume 2	A67786-01
Oracle8*i* Error Messages, Volume 3	A67788-01
Oracle8*i* Java Documentation Set	A68825-01
Oracle8*i* Enterprise JavaBeans and CORBA Developer's Guide	A64683-01
Oracle8*i* Java Developer's Guide	A64682-01
Oracle8*i* Java Stored Procedures Developer's Guide	A64686-01
Oracle8*i* JDBC Developer's Guide and Reference	A64685-01
Oracle8*i* SQLJ Developer's Guide and Reference	A64684-01
Oracle8*i* Migration	A67774-01
Oracle8*i* National Language Support Guide	A67789-01
Oracle8*i* Parallel Server Concepts and Administration	A67778-01
Oracle8i Parallel Server Setup and Configuration Guide	A67439-01
Oracle8*i* Reference	A67790-01
Oracle8*i* Replication	A67791-01
Oracle8*i* Replication API Reference	A67793-01
Oracle8*i* SQL Reference (2 volume set)	A67779-01
Oracle8*i* SQL Reference, Volume 1	A67794-01
Oracle8*i* SQL Reference, Volume 2	A67795-01
Oracle8*i* Supplied Packages Reference (2 volume set)	A68001-01
Oracle8*i* Supplied Packages Reference, Volume 1	A67840-01
Oracle8*i* Supplied Packages Reference, Volume 2	A67841-01
Oracle8*i* Tuning	A67775-01
Oracle8*i* Utilities	A67792-01
PL/SQL User's Guide and Reference	A67842-01
Pro*C/C++ Precompiler Programmer's Guide (2 volume set)	A68022-01
Pro*C/C++ Precompiler Programmer's Guide, Volume 1	A68021-01
Pro*C/C++ Precompiler Programmer's Guide, Volume 2	A68024-01
Pro*COBOL Precompiler Programmer's Guide	A68023-01

Document Title	Part Number
SQL*Plus Quick Reference	A66735-01
SQL*Plus User's Guide and Reference	A66736-01

The following books are either deprecated or only available online.

Document Title	Part Number
Oracle SNMP Support Reference Guide	A67822-01
Oracle Intelligent Agent User's Guide	A67825-01
SQL*Module for Ada Programmer's Guide (Deprecated)	A58231-01
Programmer's Guide to the Oracle Precompilers (Deprecated)	A42523-01
Pro*Fortran Supplement to the Oracle Precompilers Guide (Deprecated)	A42525-01

Oracle*8i interM*edia, Spatial, Time-Series, and Visual Information Retrieval Options

The following is the Oracle8*i interM*edia, Spatial, Time-Series, and Visual Information Retrieval Options Documentation Set. These are also orderable as two separate sets as shown; set part numbers will be assigned.

Document Title	Part Number
Oracle8*i interM*edia Documentation Set	
Oracle8*i* ConText to *interM*edia Text Migration	A67845-01
Oracle8*i interM*edia Audio, Image, and Video Java Client User's Guide and Reference	A67296-01
Oracle8*i interM*edia Audio, Image, and Video User's Guide and Reference	A67299-01
Oracle8*i interM*edia Locator User's Guide and Reference	A67298-01
Oracle8*i interM*edia Text Reference	A67843-01
Oracle8*i* Spatial, Time Series, and Visual Information Retrieval Set	
Oracle8*i* Spatial User's Guide and Reference	A67295-01
Oracle8*i* Time Series User's Guide	A67294-01
Oracle8*i* Visual Information Retrieval User's Guide and Reference	A67293-01

Document Title	Part Number
Oracle8*i* Visual Information Retrieval Java Client User's Guide and Reference	A67300-01

6

Deprecated and Desupported Features

This and prior releases of the Oracle database server have made several features or certain functionality no longer advisable for use. Some of this functionality has been, or will be, desupported.

This chapter discussed the following deprecated or desupported features.

- Strings of Zero Length Are Not Equivalent To a NULL
- The SELECT Privilege
- Date Format Strings Are Stricter
- SERIALIZABLE=TRUE Is No Longer Supported
- Non-Deferred Linking
- Single-Task Linking
- CONNECT INTERNAL
- Partition Views
- V6 Compatibility Behavior
- Use of "THE(subquery)" Expression
- Server Manager Desupport
- The SGADEF File
- LONG Column Support
- The Oracle Security Server and Crytographic Toolkit
- Dynamic Views Used for Monitoring Parallel Execution Performance

Strings of Zero Length Are Not Equivalent To a NULL

A string of zero length (''') is not equivalent to a NULL.

According to the ANSI SQL 1992 Transitional standard, a zero-length or empty string is not the same as NULL. The Oracle database server *may comply fully with this aspect of the standard in the future*. Therefore, it is recommended that applications ensure that empty strings values and NULL are not treated equivalently.

The SELECT Privilege

The SELECT privilege *may be required* on tables that users update. Always grant the SELECT privilege to a user or role if you grant the UPDATE or DELETE privileges on the table.

In conformance with SQL92, Oracle releases starting with release 7.1, allow the security administrator to require that users have SELECT privilege on a table when executing an UPDATE or DELETE statement that references table column values in a WHERE or SET clause.

For more information, refer to the discussion of the SQL92_SECURITY initialization parameter in *Oracle8i Reference*.

Date Format Strings Are Stricter

In Oracle7, a space or punctuation character in the format string caused the corresponding character in the date string to be discarded. This caused incorrect dates to be entered into the database since alphanumeric characters were thrown out. *Starting in Oracle8, an error occurs* if an alphanumeric character is found in the date string when a punctuation character or space is found in the format string.

Example:

TO_CHAR(TO_DATE('0297','MM/YY'), 'MM/YY')

Oracle7 result: 02/07

Oracle8 result: ORA-1861

SERIALIZABLE=TRUE Is No Longer Supported

The initialization parameter SERIALIZABLE=TRUE is *no longer supported in Oracle8 and beyond*. The default behavior henceforth is as if SERIALIZABLE was set to

FALSE. Use the SET TRANSACTION ISOLATION LEVEL SERIALIZABLE command to achieve similar transaction isolation behavior.

Non-Deferred Linking

Application developers are cautioned that Oracle *plans to desupport non-deferred mode linking beginning with the release of Oracle9* (it will continue to be supported with all the releases of Oracle8). Recognizing these plans, application developers should no longer use non-deferred mode linking in developing new applications. Currently Oracle supports two linking modes:

1. Non-deferred linking

   The Oracle6 OCI (client) only supported non-deferred linking which meant that for each SQL statement, a parse, a bind and a define call were each executed separately with individual round trips between the client and the server. This significantly increased network traffic between the client and the server and reduced both the performance and scalability of OCI applications.

2. Deferred linking

   Starting with the Oracle7 OCI, both non-deferred linking and deferred linking are supported. Deferred mode linking essentially defers the bind and define steps until the statement executes, so it automatically bundles and defers the bind and define calls to execution time. Further, when the application is linked with deferred mode and a special parsing call is used (the OPARSE call with the DEFFLG set to a non-zero value), even the parse call can be deferred to execution time. Note that deferred mode linking does not depend on the specific OCI calls that the application uses, only on the link option that is selected.

Deferred mode linking therefore significantly reduces the number of round trips between the client and the server and as a result improves the performance and scalability of OCI applications. The default behavior of Oracle7 OCI connected to the Oracle7 database server is deferred mode linking. However, Oracle7 OCI also supports non-deferred linking by setting specific link time options.

Further, starting with Oracle8 OCI there are two types of calls. First, all the Oracle7 OCI calls are supported with Oracle8 OCI, i.e., they will work with an Oracle8 OCI client by relinking the Oracle8 OCI libraries. Second, there are additional Oracle8-specific OCI calls. The default mode with the first type of calls continues to be deferred mode linking; however, non-deferred mode linking is supported for these calls through all releases of Oracle8 by setting link time options. However,

Oracle8-specific calls use a different paradigm and as a result non-deferred mode linking is not necessary.

The various combinations of client side libraries and server with which non-deferred linking is currently supported are summarized below:

Table 6–1 Non-deferred Linking: Server and Client-side Libraries

Server	Client-side Libraries				
	OCI release 6	OCI release 7	OCI release8 (release 7 calls)	OCI release 8 (release 8 calls)	OCI release 9
Oracle9	Not supported	Default: deferred Non-deferred supported	Default: deferred Non-deferred supported	Not supported	Not supported
Oracle8	Not supported	Default: deferred Non-deferred supported	Default: deferred Non-deferred supported	Not supported	Not supported
Oracle7	Non-deferred mode only	Default: deferred Non-deferred supported	Default: deferred Non-deferred supported	Not supported	Not supported
Oracle6	Non-deferred mode only	Default: deferred Non-deferred supported	Default: deferred Non-deferred supported	Not supported	Not supported

Oracle will continue to support deferred-mode linking with all the releases of Oracle8 (all 8.* releases). This has implications as discussed in the following.

Applications Using Oracle6 OCI Libraries

Since the Oracle6 OCI library is not supported against the Oracle8 database, applications using the Oracle6 library cannot be run against an Oracle8 database.

Applications Using Oracle7 OCI Libraries

Applications using Oracle7 OCI libraries can run in two configurations against an Oracle8 database:

1. They can be run with Oracle7 OCI libraries against an Oracle8 database in non-deferred mode provided link time options are set appropriately.
2. They can also be relinked with the Oracle8 OCI libraries and run in non-deferred mode provided link time options are set appropriately.

Oracle will support the first configuration through all the releases of Oracle8. However, the second configuration will not be supported in Oracle9. Therefore, applications that require non-deferred linking will not be able to upgrade to Oracle9 client-side libraries.

Applications Using Oracle8 OCI Libraries

Applications using Oracle8 specific OCI calls, such as those used to access Oracle8's object types, do not need to use non-deferred mode linking since the Oracle8 OCI uses a different paradigm. Applications using only Oracle7 OCI calls will be able to use non-deferred mode linking but only through Oracle Release 8.1 (Oracle8*i*).

Single-Task Linking

Single-task linking is a feature used by a limited number of Oracle's customers primarily on the OpenVMS platform. Application developers are cautioned that Oracle will continue to support single-task linking with all the releases of Oracle8 (all 8.* releases) but *will desupport it beginning with the first release after Oracle8i*.

With single-task linking, Oracle supports two configurations to link Oracle products and user-written applications against the Oracle database:

1. Single-task linking: In this case, applications are directly linked against the Oracle shareable image making single-task connection to Oracle.
2. Two-task linking: In this case, applications linked in a standalone configuration can only connect to Oracle using SQL*Net's two task drivers such as SQL*Net DECnet or SQL*Net VMS Mailbox on the OpenVMS platform. This is the typical configuration used in the large majority of client-server applications. With two task linking, applications and tools connect with the Oracle7 database through a programmatic interface that creates a shadow process for each user process. This shadow process runs a copy of the Oracle shareable image on behalf of the user process using SQL*Net protocols to communicate between the user and shadow processes. Therefore, with this interface, user routines that invoke the Oracle7 database server functions run as one process or task, and the Oracle7 routines that execute these functions run as the second task.

Application developers who would like to use single-task linking to run their applications will not be able to do so against the first server release after Oracle8.

CONNECT INTERNAL

CONNECT INTERNAL is currently supported for backwards compatibility only. CONNECT INTERNAL *will be completely desupported in Oracle8i, release 8.2*. If you have not done so already, you should plan to migrate your applications to use other connection syntax. See *Oracle8i Administrator's Guide* for further details.

Partition Views

In Oracle8*i*, *partitioned tables are strongly recommended in preference to partition views*. If you must use partition views, please see "Rules and Guidelines for Use" in the Release 7.3.3 Tuning Guide on page 11-2.

V6 Compatibility Behavior

With Oracle7, Oracle offered a Version 6 [V6] compatibility flag that allowed application developers developing Oracle7 applications to emulate Oracle6 behavior. With the release of Oracle8*i* (but starting in Oracle8, release 8.0), users are again cautioned that the *Version 6 compatibility flag has been desupported in all of the Oracle8 products* including PL/SQL8, all the Oracle Precompilers, the Oracle8 Oracle Call Interface, SQL*Module, and SQL*Plus. The desupport of the V6 compatibility flag is consistent with Oracle's policy of supporting backwards compatibility and behavior from one version release upgrade to another, i.e., from Oracle6 to Oracle7 but not for more than one version release upgrade.

Users who do not absolutely need to maintain V6 behavior are encouraged to upgrade their Oracle7 clients to Oracle8*i*. Users who absolutely need to continue to emulate V6 behavior for certain applications need to maintain one $ORACLE_HOME with an Oracle7 client for those applications. They can create a separate $ORACLE_HOME with an Oracle8*i* client for those applications for which they do not need V6 behavior.

Specifically, the V6 compatibility flag emulated the following aspects of OracleV6 behavior with Oracle7:

- String literals are fixed length in Oracle7 but are treated as variable length with the V6 flag.
- PL/SQL local CHAR variables are fixed length in Oracle7 but are treated as variable length with the V6 flag.
- Return value of SQL functions (e.g., USER) are fixed length characters in Oracle7 but are treated as variable length characters with the V6 flag.

- Select/Fetch of a NULL with no indicator raises an ORA-1405 error with Oracle7 but returns no error with the V6 flag.
- SQL group function is called at FETCH time with Oracle7 but is called at query execution time with the V6 flag.
- Describe of a fixed length string returns Type=96 with Oracle7 but returns Type=1 with the V6 flag.

All of the above are desupported with the desupport of the V6 Compatibility Flag in all releases above Oracle8, release 8.0.

Use of "THE(subquery)" Expression

The current syntax for the *table_expression_clause* of a SELECT, INSERT, UPDATE, or DELETE statement, includes a *table_collection_expression*. This expression is recognized by the TABLE( ) keyword (i.e.: TABLE(*collection_expression)*), and it informs Oracle that the collection value expression should be treated as a table. Further, *collection_expression* may be a subquery that selects a nested table column from a table or view.

Prior to Oracle8, release 8.0, *table_collection_expression* was expressed as "THE (subquery)". *That usage is now deprecated.*

For more information, see *Oracle8i SQL Reference*.

Server Manager Desupport

The Server Manager *will be desupported in a future release* of Oracle. You should begin to migrate your Server Manager SQL scripts to SQL*Plus scripts as soon as possible.

The SGADEF File

The contents of this file have been deleted, but the file remains because its existence is required by Oracle. *Do not delete this file*. It will be obsoleted in a future release.

LONG Column Support

LONG column support will be discontinued *in a future Oracle release*. You are advised to migrate LONG data into LOB columns.

The Oracle Security Server and Crytographic Toolkit

The Oracle Security Server and Cryptographic Toolkit products do not exist in Oracle8*i*, but their functionality has been integrated into the standards based products: Oracle Advanced Security option and Oracle Internet Directory. *Users must migrate* their applications to use these Oracle8*i* components.

Dynamic Views Used for Monitoring Parallel Execution Performance

The following dynamic views, used for monitoring parallel execution performance, *will be removed, renamed, or replaced in the next major Oracle release (Release 9.0).*

View	Description	Action
V$PQ_SESSTAT	Provides parallel execution statistics for each session	Removed
V$PQ_SYSSTAT	Provides parallel execution statistics for the system	Replaced by new view: V$PX_PROCESS_SYSSTAT
V$PQ_SLAVE	This view tallies the current and total CPU time and number of messages sent and received per parallel server process	Replaced by new view: V$PX_PROCESS
V$PQ_TQSTAT	Provides a detailed report of message traffic at the table queue level	Renamed to V$PX_TQSTAT

Glossary

application context

A secure data cache for storing information used by fine-grained access control to make access control decisions. Used in conjunction with fine-grained access control.

autonomous transactions

Transactions within an autonomous PL/SQL block, which is a block that has a transaction scope independent of the transaction scope of the calling PL/SQL block. Autonomous transactions are not nested transactions, since they do not share resources with the calling transaction and committed changes are immediately visible to other transactions regardless of whether the calling PL/SQL block commits or rolls back.

cache fusion

A new "diskless" ping architecture, used in the Oracle Parallel Server, that provides copies of blocks directly from the holding instance's memory cache to the requesting instance's memory cache.

composite partitioning

Partitions data using the range method and, within each partition, subpartitions it using the hash method (see hash partitioning). It provides the manageability and availability benefits of range partitioning with the data distribution advantages of hash partitioning.

CUBE

One of the Oracle8*i* operators that can be used in a GROUP BY clause for dimensional analysis. Similar to the ROLLUP operator (see ROLLUP), but also

includes cross-tabulations--i.e., it uses *every* possible combination of the columns or expressions in the GROUP BY clause to produce super-aggregate rows.

data sampling

A process used by the ANALYZE command or the DBMS_STATS package to *estimate* statistics for a database object (table, index, or cluster). It involves analyzing a specified (or default) percent (or number) of random rows or blocks to estimate statistics.

Database Configuration Assistant

A Java-based utility that enables the creation, modification, or deletion of an Oracle database. It can be invoked as a standalone Java application from the Oracle Universal Installer or as an applet from the Java-based Oracle Enterprise Manager.

dimension

Used in structuring materialized views (see materialized view), a dimension defines hierarchical (parent / child) relationships between pairs of columns or column sets. A hierarchical relationship is a functional dependency from one level of a hierarchy to a more abstract level in the hierarchy. Each value at the child level is associated with one and only one parent-level value.

DISABLE VALIDATE constraint state

A database state that saves space by requiring no index on a unique or primary key, and guarantees the validity of existing data in the table. This state also enables efficient loading of data from a non-partitioned table into a partitioned table.

domain index

A user-defined index for a complex datatype. The application software is responsible for defining the index structure, maintaining the index content during load and update operations, and searching the index during query processing. The index structure itself can be stored either in the Oracle database as tables or externally as files.

enterprise roles

Stored in the Oracle Internet Directory, they enable centralized authorization of users by defining a users privileges in the enterprise. If a user changes jobs, his enterprise role assignment can be changed, altering his privileges in multiple databases throughout the enterprise. Also, an administrator can add capabilities to enterprise roles (granted to multiple users) without having to update the authorizations of each user independently.

event triggers

Refers to triggers that can now be specified on database and schema events. These are:

- DML events (DELETE, INSERT, UPDATE)
- DDL Events (CREATE, ALTER, DROP)
- Database events (SERVERERROR, LOGON, LOGOFF, STARTUP, SHUTDOWN)

extensibility

Capability of being extended. For example, the Oracle8*i* extensibility framework allows partners to easily provide their own implementations of RDBMS services and register them with the server.

Fast-Start Fault Recovery

A cluster of features that provide high availability by enabling the database to recover quickly from instance failures. Fast-Start Parallel Rollback rolls back transactions in parallel. Fast-Start On-Demand Rollback recovers individual blocks in large transactions that require user access. Fast-Start Checkpointing allows users to set an upper limit for the number of blocks requiring recovery. Replaces incremental checkpoint feature of Oracle8.

fine-grained access control

A feature that supports user-defined security policy for row-level access control. Rather than embedding security rules in views, the server dynamically appends a WHERE condition to a query that allows security rules to be acquired when a table or view is referenced in a DML statement. This enforces security 100%, across all applications, since it is in the database that the security is enforced.

function-based index

The value of a function or expression is precomputed and stored in the index. Performance is dramatically improved when a computationally intensive expression is used to build the index. When an application uses such an expression in its where clause, much faster access is provided to the data. This greatly enhances the use of parallel I/O.

hash partitioning

Partitions data according to a hash function. Hash partitioning provides a very simple way to break data into evenly sized containers to be spread across multiple

I/O devices, or even multiple machines in a shared-nothing cluster. This greatly enhances the use of parallel I/O.

*inter*Media

Oracle8*i* *inter*Media. This is Oracle's offering for managing multimedia data in an integrated fashion with other enterprise data. It is specifically aimed to support the data requirements of the Internet. See multimedia data.

Internet Computing

Using the infrastructure of the Internet to bring business applications to the enterprise. These applications are deployed on scalable, professionally managed servers, and are accessible across the organization in a low cost computing environment. Internet computing makes use of such components as browsers, Web servers, and internet standards such as SSL and LDAP.

Internet File System (*i*FS)

A Java application that runs within Oracle8*i* and provides universal access to any data in the database. The *i*FS appears as if it were just another volume on the network. Relational data can appear as files; so too can hybrid documents that combine relational and non-relational data. Whether the user accesses the contents of *i*FS through Windows Explorer, a Web browser, an FTP client, or an e-mail client, the files appear the same.

Java VM

A complete, JDK 1.1-compliant Java execution environment. The Java VM runs in the same process space and address space as the database kernel, sharing its memory heaps and directly accessing its relational data. This design optimizes memory use, increases throughput, and delivers an open, highly available, secure, and manageable Java server.

JPublisher

A utility that translates user-defined types to Java wrapper classes. It is similar to the Object Type Translator (OTT) utility used in C/C++ environments.

JServer

Oracle's Java Virtual Machine (Java VM). It includes Java stored procedures, methods, and triggers. It also provides EJB, CORBA, and HTTP support.

JServer Accelerator

A native code compiler that speeds up the execution of Java code by eliminating interpreter overhead. It translates standard Java binaries into C programs that are processed by a platform-dependent C compiler into native libraries that the Oracle Java VM can load dynamically.

linguistic-sorted index

An NLS-supported, function-based index that uses a linguistic sort order for optimized performance. For example, you can create an NLS index on a table that returns an extended Spanish collation order.

locally managed tablespaces

A mechanism that manages space within a tablespace by using bitmaps to track all extent information in the tablespace itself. This optimizes space usage and consumption in the tablespace, minimizing fragmentation so that data does not need to be re-organized. Bitmaps require no dictionary access to allocate or update extents for the tablespace.

logical ROWID

Primary-key-based logical identifiers that expand ROWID functionality to index-organized tables (IOTS), whose rows do not have permanent physical addresses. Oracle uses these logical ROWIDs for the construction of secondary indexes on index-organized tables.

LogMiner

A utility that allows you to analyze online or archived redo logs according to user-specified criteria. For example, you can track changes to the database made by a specified user.

materialized view

A stored summary or snapshot (remote materialized view used for replication) containing precomputed results. A CREATE MATERIALIZED VIEW statement containing a subquery, typically a join or a data aggregation, is used to create the materialized view. This provides much faster access to summary data. Local materialized views can be used for query rewrite.

multimedia data

Includes text, audio data, video data, images, and graphic objects. Oracle8*i* *inter*Media enables Oracle8*i* to manage this data in an integrated fashion with other enterprise data.

Oracle Advanced Security

A renaming and expansion of the Oracle Advanced Networking Option.

Oracle Internet Directory (O*i*D)

An LDAPv3-compliant, hierarchical data repository for storing enterprise user information, including enterprise roles and Oracle Wallets. It combines a native implementation of the Internet Engineering Task Force's (IETF) LDAP v3 standard with an Oracle8*i* back-end datastore.

Oracle Universal Installer

A Java-based installation utility that has the same interface across all platforms. Oracle Universal Installer interprets the staging area prepared by Oracle Software Packager and performs the appropriate installation actions.

Oracle wallet

An encrypted data structure that contains a user's credentials: a private key, a user certificate, and a list of trustpoints (certificate authorities that the user trusts). Users can store wallets in a directory and thus access their credentials from anywhere in the organization.

plan stability

A method for stabilizing execution plans used by the optimizer that is independent of changes to the system configuration, parameters, statistics, or even the optimizer itself. You can use initialization parameters or SQL statements to create stored outlines containing a set of attributes used by the optimizer. Useful for optimizing applications for multiple environments.

query rewrite

With the Oracle8*i* summary management feature, this is an automatic rewrite of a query by the Oracle database server to use the summary data, rather than retrieving data from detail tables by doing expensive joins and aggregate operations. This rewrite facility is totally transparent to the application, which is not aware of the existence of the materialized view.

read-only standby database

A standby database that is opened in read-only mode, making it available for queries. You can move a standby database from recovery mode to read-only mode and back again.

ROLLUP

One of the Oracle8*i* operators that can be used in a GROUP BY clause for dimensional analysis. A commonly performed operation in a data warehousing environment, and in producing materialized views, is to show totals at a coarse granularity and then to show subtotals at successively finer levels. ROLLUP produces these results.

rule-based publish/subscribe

An Advanced Queuing model in which the sending applications anonymously publish messages and the receiving applications independently and autonomously subscribe to these messages (using SQL-based rules). This feature allows messages to be filtered by content.

secondary index

An index on an index-organized table that enables performing operations based on a column that isn't part of the primary key. Oracle constructs secondary indexes on index-organized tables using logical row identifiers (logical ROWIDs) that are based on the table's primary key.

security policy

Defines who is allowed access and to which data. Various means are available within Oracle8*i* for enforcing security policies including, but not limited to, privileges, roles, the virtual private database feature, and network authentication and authorization.

snapshot refresh group templates

A feature that allows a single refresh group to contain up to 400 snapshots (or remote materialized views) and also reduces the number of round trips required to refresh snapshots in a group.

SQLJ

Oracle's precompiler for Java source code. It enables a user to embed SQL statements in the Java source. The precompiler takes the source code and produces as output new Java source code, where all of the SQL statements are translated into Java class definitions that implement these statements.

summary advisor

A DBMS_OLAP package to help users design and evaluate materialized views for query rewrites.

temporary LOBs

LOBs (Large object datatypes) that do not exist permanently in the database, but exist mainly for the purpose of performing transformations on LOB data. Many applications have a need for temporary LOBs, which act like local variables.

temporary tables

A table with session-specific or transaction-specific data that is empty when a session or transaction begins and discarded when it ends. Temporary tables are useful for saving intermediate results that can be joined back into another table.

transportable tablespaces

A feature that allows a user to move a subset of an Oracle database into another Oracle database. It is also possible to clone a tablespace in one database and plug it into another, thereby copying the tablespace between databases.

user-defined operator

A SQL operator whose definition and implementation is provided by the user. Part of the extensibility architecture, the Oracle database server allows these user-defined operators to be used in SQL statements in the same manner as any of the predefined operators provided by Oracle.

vertical partitioning

Selecting only the relevant *columns* for replication to a remote site. In addition to minimizing the amount of data sent to a site, it provides protection from changes at the master site. When adding a column to a master table (or dropping a column not in use by a dependent snapshot), the snapshot can continue to perform fast refreshes, without being affected by the DDL change.

virtual private database

A feature consisting of the fine-grained access control and application context.

WebDB

An HTML-based development tool for building HTML Web pages with content based on data stored in the Oracle database. It also provides an ease of use GUI interface to Oracle data, so a user is not required to know SQL to access their data.

Index

A

B

C

D

E

F

G

H

I

J

L

M

N

O

P

R

S

T

U

V

W

X

Z