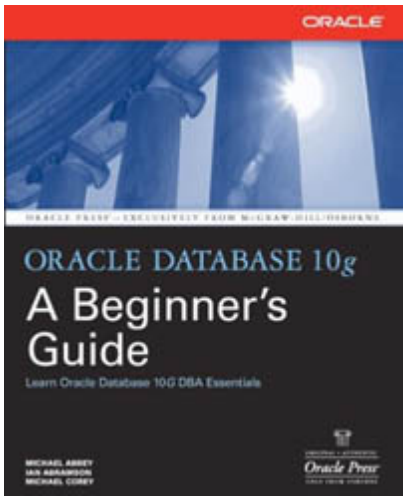
Cover



Oracle Press

Oracle Press

Oracle Database 10*g*: A Beginner's Guide

Ian Abramson
Michael S. Abbey
Michael Corey

McGraw-Hill/Osborne

New York Chicago San Francisco
Lisbon London Madrid Mexico City Milan
New Delhi San Juan Seoul Singapore Sydney Toronto

About the Authors

Ian Abramson is the CTO of Red Sky Data, a Toronto-based company that has been building a results-based track record for delivering high-quality Data Warehouse and Oracle solutions to its clients around the world.

Ian is coauthor of several Oracle titles including books on Data Warehousing, on Tuning, and, of course, in the Oracle Beginner's Guide series. He is very active in the Oracle user community and is the International Oracle Users Group's Director of Education and Programming. Ian is also well known for his lively seminars and technical training classes. While Ian is busy during the day building Oracle Data Warehouses and applications, by night he continues to engage in his dreams of being a professional hockey goalie. Ian may be contacted at ian.abramson@redskydata.com.

Michael S. Abbey is a frequent presenter at many Oracle user group events. He has been working with the software since the mid '80s, having had the pleasure of experiencing versions 3 through 10*g*. Michael has been very active in the International Oracle Users Group, a Chicago-based voice for the Oracle software user community. Michael's fort is anything related to installation, configuration, backup/recovery, and management of very large databases. He is recognized in many circles as an expert in many facets of the Oracle technology. He is the owner of two 1970s Fender Precision bass guitars, evidence of one of his other passions loud, grinding rock music.

Michael Corey, an entrepreneur, author, and recognized expert on relational databases and Data Warehousing, founded Ntirety Technologies in May 2001. As CEO, he leverages his extensive experience and business savvy to manage and expand the premier Remote Administration Services firm in New England. Corey is a frequent speaker at technology conferences throughout the world and has written numerous articles and books (published by McGraw-Hill/Osborne) on topics such as SQL Server 7 Data Warehousing, and Oracle8 and Oracle8*i* Data Warehousing; he's also contributed to several Beginner's Guides. Corey also founded and is actively involved in numerous Oracle associations and industry user groups.

Previous Next

About the Contributing Authors

Steve Jones is a founding partner of Red Sky Data (www.redskydata.com), a consulting firm specializing in Data Warehousing and business intelligence implementations. Steve has been working with Oracle databases for a number of years in large-scale Data Warehousing environments such as insurance and telecommunications. His involvement on projects has spanned all phases of the development lifecycle including analysis, design and architecture, development, and testing. Steve lives in Toronto and can be reached at steve.jones@redskydata.com.

After developing a solid foundation of Oracle skills in his native Ottawa during the '90s, **Mark Kerzner** moved with his wife, Arlene, and their four children to the warmer climate of Phoenix, Arizona, in 1998. Mark works for the Pharma division of NDC Health and has served in both lead software engineer and project manager roles for them since moving to Phoenix. Currently, he is the technical lead for the Data Warehouse Solutions group. In 2001, Mark earned a Project Management Professional (PMP) designation from the Project Management Institute. You can contact him by e-mail at kerz@cox.net.

Michael Mallia, an Oracle expert in the National Capital region around Ottawa, is a recognized expert in the utilization of XML and the Oracle RDBMS for "Near Real Time" data integration. Michael is the senior data architect and founding member of Xteoma, Inc. He can be reached at mmallia@xteoma.com.

Tim Quinlan is an Oracle Certified Database Administrator with over 10 years of Oracle experience; he has worked with databases since 1981. Tim has performed the roles of DBA, architect, designer, and implementer of enterprise-wide Data Warehouse and transactional databases. This work has been performed in many business sectors including government, financial, insurance, pharmaceutical, energy, and telecommunications. Tim has spoken

at many conferences, taught database courses, and written feature articles for leading database publications. His main (professional) interest is designing and implementing very large, high-performance, high-availability database systems.

George Trujillo is the president and CEO of Trubix, Inc., the largest third-party provider of Oracle education materials in the world. Trubix focuses on integrating leading-edge technologies such as Java, Web Services, Oracle, and XML. George Trujillo is internationally recognized and has been selected as a keynote and master presenter at numerous technical and business conferences. He has over 17 years of Oracle DBA and developer consulting experience.

Contents

Acknowledgments

Ian Abramson: I would like to thank my wife, Susan (who is the best decorator in the world), and my children, Baila and Jillian. We are strongest as one, we all draw strength from each other. Thanks also to my coauthors: you have helped to make this book a great project! YATFG to all! I would also like to thank the people who I work with and the people who I play with, so thanks to ReMax All-Stars hockey, Red Sky Data hockey, David Stanford, Paul Herron, Rob Snoyer and Ted Falcon, Jack Chadirjian, and, of course, my dad, Joe, who has taught me about what is truly important in life family. Thank you to all, I could not have done it without you, and I share this book with each of you!

Michael Abbey: I would like to recognize my immediate and extended families, who have always helped me find ways to advance my career and satisfy my voracious appetite for Oracle's technology.

Michael Corey: Thanks to my family, and to my friends Mike Abbey and Ian Abramson, for all of their understanding and support. Thanks also to my good friends Ed Marram and Les Charm for all their help and support. To bring this book to press required a lot of time and effort from a lot of great people at McGraw-Hill/Osborne thank you once again.

Steve Jones: I would like to thank my wife, Sandra, for her unwavering support and encouragement, as well as her patience and understanding. Thanks also to my loving kids, Devon, Spencer, and Matthew, for keeping me young and reminding me of the important things in life. Last but not least, I would like to thank coauthor Ian Abramson for his advice and support, and for giving me the opportunity to contribute to this book.

Mark Kerzner: Thank you to my wife, Arlene, and our four children, Marissa, Amanda, Shane and Dalia, whose excitement about this opportunity rivaled mine. They support every step I take, and for that I am deeply grateful. To my parents for their unconditional love and support. To my mentors, Ian Abramson and Michael Abbey, who launched my IT career and have always been there to encourage and

support me. To the many friends I have made over the years, especially the JPL friends who are my Arizona family. You all have contributed to whom I have become.

Michael Mallia: First and foremost, to my soulmate, Shauna. Without her, our house would not be a home. Secondly, to my four-year-old's Godfather and his number-one birthday party invitee, MichaelAbbey (MichaelAbbey is one word!), for the dedication and love he obviously has for my family. May we know each other until I change my belt size.

Tim Quinlan: Special thanks to Helen, Ryan, and Brendan for supporting and helping me with this work.

George Trujillo: I would like to say a special thanks to my wife, Karen, and kids, Cole, Madison, and Gage, for their love and patience during all the long nights and early mornings while writing course materials.

Introduction

Oracle Database 10*g* marks the latest release by a company that has experienced a meteoric rise to success over the past 25-plus years. They have been grossing many billions of dollars annually for many years, vending a suite of solutions powered by their flagship product the Oracle database. It has gone through many changes in names v6, Oracle7, Oracle8*i*, Oracle9*i*, and now Oracle 10*g*. Regardless of what it is called, the Oracle server has been catapulted to the forefront of our Internet-savvy society, playing a role as the primary data server on a web site in your neighborhood. This book is your introduction to the Oracle Database 10*g* technology. It is the start of your journey a quick start to a complex and popular technology.

Oracle Database 10*g* is the culmination of thousands upon thousands of person hours building an infrastructure to deliver data to a hungry, worldwide community, just as electricity is delivered to a three-prong outlet near you. Larry Ellison, CEO of Oracle Corporation, is a visionary steering Oracle's product set in directions unheard of before. You cannot read any public relations or technical material from Oracle Corporation without hearing that four letter word *grid*. With grid computing, the industry envisions a computational grid where machines all the way from the Intel-based server to the high-end servers from HP, IBM, and Sun are interlaced with one another is a massively scalable and sharable environment.

There have been many advances in the processing power of computer chips over the past few decades, and grid computing is seen as allowing applications to harness that power. Idle processor time is deliberately consumed by shared applications. The analogy to the electricity grid is an interesting one. When you plug your iron into a socket in your basement, you neither know nor care where the electricity is coming from it's just there and taken for granted. With Oracle Database 10*g* grid computing, transparent access is provided to a wide network of remote computers. Unbeknownst to application users, processing is shared between widely disparate sites, where the location of nodes responsible for data delivery is dynamic hence the likeness to the

CHAPTER 1
Database Fundamentals

CRITICAL SKILLS

1.1 Define a Database

1.2 Learn the Oracle Database 10*g* Architecture

1.3 Learn the Basic Oracle Database 10*g* Data Types

1.4 Work with Tables

1.5 Work with Stored Objects

1.6 Become Familiar with Other Important Items in the Oracle Database 10*g*

1.7 Work with Object and System Privileges

1.8 Introduce Yourself to the Grid

1.9 Tie It All Together

This chapter is your first one on your Oracle Database 10*g* journey. From here on out, we will walk you through the skills that you need to begin working with the Oracle Database 10*g*. We'll begin at the core of this product, with the fundamentals of a database. This chapter will also help you form an understanding of the contents of your database and prepare you to move into the complex areas of Oracle Database 10*g* technology.

CRITICAL SKILL 1.1
Define a Database

Oracle Database 10*g* the latest offering from a software giant in northern California. Perhaps you have heard a lot of hype about Oracle Database 10*g*, perhaps not. Regardless of your experience, 10*g* is a rich, full-featured software intended to revolutionize the way many companies do their database business. *Database* you say now there's a word you hear all the time! In a nutshell, a database is an electronic collection of information designed to meet a handful of needs:

1. Databases provide one-stop shopping for all your data storage requirements, be they in diverse areas such as human resources, finance, inventory, or sales and then some. The database contains any amount of data, from the small to the huge. Data volumes in excess of many hundreds of gigabytes are commonplace in this day and age, where a gigabyte is 1,073,741,824 bytes.

2. Databases must provide mechanisms to retrieve data quickly as applications interact with their contents. It is one thing to store tax information for the 300 million citizens of a country, but it's another kettle of fish to retrieve that data, as required, in a short time period.

3. Databases allow the sharing of corporate data such that personnel data is shared amongst one's payroll, benefits, and pension systems. A familiar adage in the database industry is "write once, read many." Databases are a manifestation of that saying one's name, address, and other tombstone personnel information are stored in one place and read by as many systems requiring these details.

There is a great deal of academic interest in the database industry, the theory of the relational database being founded in relational algebra. As data is entered into and stored in the Oracle Database 10*g*, the relationships it has to other data are defined as well. This allows the assembling of required data as applications run. These relationships can be described in plain English for a fictitious computer parts store as follows:

- Each geographical location within which the store does business is uniquely identified by a **quad_id.**

- Each manufacturer that supplies parts is uniquely identified by a ten-character **manufacturer_id.** When a new manufacturer is registered with the system, it is assigned a **quad_id** based on its location.

- Each item in the store's inventory is uniquely identified by a ten-character **part_id,** and must be associated with a valid **manufacturer_id.**

Based on these three points, practitioners commonly develop statements similar to the following to describe the relationships between locations, manufacturers, and parts:

- There is a one-to-many relationship between locations and manufacturers more than one manufacturer can reside in a specified location.

- There is a one-to-many relationship between manufacturers and computer parts the store purchases many different parts from each manufacturer.

These two relationships are established as data is captured in the store's database and other relationships can be deduced as a result for example, one can safely say "parts are manufactured in one or more locations based on the fact that there are many manufacturers supplying many different products." Oracle has always been a relational database product, commanding a significant percentage of market share compared to its major competition. Let's get started and look at the Oracle Database 10*g* architecture.

CRITICAL SKILL 1.2
Learn the Oracle Database 10*g* Architecture

As with many new software experiences, there is some jargon that we should get out of the way before starting this section.

- Oracle Database 10*g* is said to be *started* when the appropriate commands have been invoked to make it accessible on a day-to-day basis to applications.

- The act of stopping Oracle Database 10*g* is called *shutdown*. When Oracle Database 10*g* is shut down, nobody can access the data in its files.

- An *instance* is a set of processes that run in a computer's memory and provide access to the many files that come together to define themselves as Oracle Database 10*g*.

- A *background process* supports access to a started Oracle Database 10*g*, playing a vital role in Oracle's database implementation. Various background processes are spawned when starting the database and each performs a handful

of tasks until a database is shut down.

Let's now look at the assortment of files and background processes that support the Oracle Database 10*g*.

Previous Next

As with most lists, after reading the preceding bullet points, you may wonder what else DBAs do with their time. As you work with the Oracle Database 10*g*, you will experience other activities that will plug the loopholes that may exist in the previous list.

CRITICAL SKILL 1.3
Learn the Basic Oracle Database 10*g* Data Types

Very early in one's journey through the world of Oracle Database 10*g*, it becomes time to learn its common data types. Regardless of one's past experiences in information technology, data types are nothing new. Let's look at the most common type of data that can be stored in the Oracle Database 10*g*, keeping in mind that the list is much longer than the one we present here.

varchar2

By far the most common data type, varchar2 allows storage of just about any character that can be entered from a computer keyboard. In earlier software solutions, we commonly referred to this as *alphanumeric* data. The maximum length of varchar2 is 4000 bytes or characters. It is possible to store numeric data in this data type. This is a variable length character string, with no storing of trailing insignificant white space. Thus, if "Turkey" is passed to a column defined as varchar2, it will store the text as "Turkey". The following listing shows a few sample varchar2 data definitions.

```
SQL  select * from date_test;
```

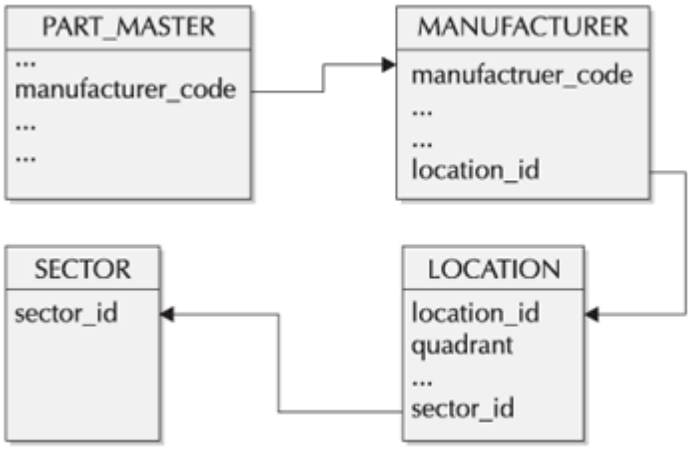


FIGURE 1-2. *Relationships to* ***part_master***

Suppose someone wanted to know where in the country a certain part was manufactured. By looking at Figure 1-2, that information is not readily available in **part_master.** However, **part_master** has a **manufacturer_code.** So, a person would traverse to **manufacturer** using **manufacturer_code** to get a **location_id.** Armed with that value, one then proceeds to **location** to get a **quadrant** column value. After this navigation is complete, a person would know where a specific part is built. Table 1-3 maps out this journey.

As illustrated in Table 1-3, we can deduce that part 33499909 comes from the Pacific Northwest a deduction that is made by following the relationships between matching columns in the three tables in question.

CRITICAL SKILL 1.5
Work with Stored Objects

Oracle Database 10*g* offers the ability to store user-defined programming units in the data dictionary, called *stored objects*. These programming units are written in PL/SQL, the topic of Chapter 7. Without worrying about what goes inside these objects, let's do an overview of what they are all about.

Table	Part Number	Column Value	Related Column Value
part_master	33499909	manufacturer_code	3490
manufacturer	3490	location_id	5
location	5	quadrant	Pacific Northwest

TABLE 1-3. *Following Relationships Between Tables*

CRITICAL SKILL 1.6
Become Familiar with Other Important Items in the Oracle Database 10*g*

So far, we have had a brief look at tables, views, tablespaces, and a handful of stored objects views, triggers, procedures, packages, and functions. Let's round out this introduction to Oracle Database 10*g* architecture by covering a few other items commonly encountered from day one. This discussion is a hodge-podge of things that are necessary for a person's understanding of the Oracle Database 10*g* architecture and operations. We must spend a bit of time first looking at the database administrator, affectionately called the *DBA*, the gatekeeper of the database and the person responsible for its smooth operation. There is a more detailed look at the DBA in Chapter 3, with more information on how DBAs go about carrying out their administrative chores.

Indexes

Tables are made up of rows and columns, being the baseline of all objects in the Oracle Database 10*g*. As applications interact with the database, they often retrieve vast amounts of data. Suppose MyYP, a fictitious Internet company, provided Yellow Pages listings for North America, and the data was stored primarily in a table called YP_MASTER. Each row in the YP_MASTER table is uniquely identified by a combination of company name, municipality, and geographic location (state or province). As words are retrieved from the database to satisfy online queries, indexes would provide a quick access path to the qualifying data. These characteristics about indexes are relevant to the power they deliver in the Oracle Database 10*g*. For instance:

- They are built on one or more columns in a table using simple SQL statements.
- They are separate from the tables upon which they are built, and can be dropped without affecting the data in the table itself. On the contrary, when a table is dropped, any indexes it has disappear with the table.
- The function they perform can be likened to the index in a book. If one were looking for a specific topic in a textbook, the best place to start would be the index it provides a shortcut to the information being sought. If one imagined that YP_MASTER were a book rather than a table, finding Y M Plumbing in Pensacola, Florida would be faster using the index than reading the book from the start into the entries for the 25th letter of the alphabet. The names on the corner of the pages in a phone book are like an index.
- Indexes occupy space in the database and, even though there are ways to keep their space to a minimum, extra space is required and must be preallocated.

CRITICAL SKILL 1.7
Work with Object and System Privileges

It's next to impossible to work with data in the Oracle Database 10*g* without looking at object privileges. In this section, we are going to look at these privileges as well as a suite of system privileges closely related to managing the Oracle Database 10*g*. The four main object privileges are **select, insert, update,** and **delete,** discussed in the next four sections. Oracle Database 10*g* uses the term *grant* when referring to giving out both object and system privileges.

Select

This is the primary and most commonly used privilege, permitting other users to view your data. There are three parts to grant statements:

1. The keywords grant select on.

2. The name of the object upon which the privileges are being given out.

3. The recipient of the grant.

Once the **select** privilege has been given out, the recipients, using a private or public synonym as described earlier in the "Synonyms" section of this chapter, can reference your objects in their SQL statements.

Insert

This privilege allows users to create rows in tables belonging to other users. The creator of new rows in other users' objects is bound by the same rules used if they owned the objects themselves. They must adhere to the boundaries defined by the data types of the columns in the rows they create. For example, when rows are inserted into a table that has a column defined as type DATE, they must ensure that valid date type data is placed in the column so defined. As rows are created in an Oracle Database 10*g* table, the transaction must be committed to the database before the row becomes part of the information available to other users. With Oracle Database 10*g*, we use the term *commit* synonymously with *save* with other types of software.

Progress Check Answers

1. Installation, upgrades, tuning, and environment setup are four of many tasks performed by the DBA.

2. A private synonym can only be referenced in an SQL statement by the account who created and owns the synonym. A public synonym, created by a centralized user such as a DBA, is available to all users.

3. The default tablespace is the one within which users occupy space by default, unless another tablespace is mentioned as a table is created.

4. Quota on tablespaces is usually given out using bytes or megabytes as units of measurement.

5. The DBA goes to MetaLink to request assistance from Oracle's support organization.

6. Triggers cannot exist on their own without association with an Oracle Database 10*g* table.

Previous Next

Ask the Expert

Q: Name the four main object privileges used in the Oracle Database 10*g*.

A: The four most common privileges are **select, insert, update,** and **delete.**

Q: Placing an Oracle Database 10*g* in a state where it can be accessed by applications is referred to as what activity?

A: Putting an Oracle Database 10*g* in a normal operating mode for day-to-day access by a company's applications is referred to as *startup*.

Q: How many integer and decimal digits can a field defined in the data dictionary as *number(10,2)* accommodate?

A: The field would be able to store up to eight integer digits and two decimal digits.

Q: When Oracle Database 10*g* is passed the value "Beginner " for storage in a varchar2 column, how does it deal with trailing insignificant spaces?

A: The trailing spaces are trimmed before the information is stored in the database. Though not as common as varchar2, the char data type can be used to store trailing spaces.

Q: What would Oracle Database 10*g* store as a value in a number(6,2) field when passed the value 9.8882?

A: It would store 9.89 in a *number(6,2)* field when passed 9.882.

System privileges were introduced with early releases of Oracle7 (circa 1993) and have played a useful role in the division of labor in the database since their inception. Now it's time to get into the meat of the seventh letter of the alphabet, *g*, that throughout this chapter has followed the two-digit version number of this software release 10.

CRITICAL SKILL 1.8
Introduce Yourself to the Grid

As many have heard, the "*g*" in Oracle Database 10*g* stands for *grid*. Grid computing is a technology that allows for seamless and massively scalable access to a distributed network of diverse yet homogenous computer types. Oracle Database 10*g* is the glue

FIGURE 1-4. *OEM Startup*

CRITICAL SKILL 1.9
Tie It All Together

Now that was quite a journey! We have covered database fundamentals, with an Oracle Database 10*g* flavor. Relational database management systems have been around for a few decades, and the release of Oracle Database 10*g* is a landmark in the industry. There have been many academic discussions about the grid technology some claim Oracle Database 10*g* is a grid implementation, some don't. Regardless of which side of the fence you're on, Oracle Database 10*g* is a big step. Let's pull it all together and spend a bit of time on the big picture.

Oracle Database 10*g* is a collection of special files created using its database configuration assistant, then completing the work using OEM Grid Control. Access to these database files is facilitated by a set of shared memory processes referred to

CHAPTER 2
SQL: Structured Query Language

CRITICAL SKILLS

2.1 Learn the SQL Statement Components

2.2 Use Basic insert and select Statements

2.3 Use Simple where Clauses

2.4 Use Basic update and delete Statements

2.5 Order Data

2.6 Employ Functions: String, Numeric, Aggregate (No Grouping)

2.7 Use Dates and Data Functions (Formatting and Chronological)

2.8 Employ Joins (ANSI vs. Oracle): Inner, Outer, Self

2.9 Learn the group by and having Clauses

2.10 Learn Subqueries: Simple and Correlated Comparison with Joins

2.11 Use Set Operators: Union, Intersect, Minus

2.12 Use Views

2.13 Learn Sequences: Just Simple Stuff

2.14 Employ Constraints: Linkage to Entity Models, Types, Deferred, Enforced, Gathering Exceptions

2.15 Format Your Output with SQL*Plus

SQL is the fundamental access tool of the Oracle database; in fact, it is the fundamental access tool of all relational databases. SQL is used to build database objects and it is also used to query and manipulate both these objects and the data they may contain. You cannot insert a row of data into an Oracle database unless you have first issued some basic SQL statements to create the underlying tables. While Oracle provides SQL*Plus, a SQL tool that enables you to interact with the database, there are also many GUI tools that can be used, which then issue SQL statements on your behalf behind the scenes.

CRITICAL SKILL 2.1
Learn the SQL Statement Components

Before learning many of the SQL commands that you will use frequently, first let's take a look at the two different categories into which SQL statements are classified. They are DDL, or *data definition language*, and DML, or *data manipulation language*. The majority of this chapter will deal with the latter.

DDL

DDL is the set of SQL statements that define or delete database objects such as tables or views. For the purposes of this chapter, we will concentrate on dealing with tables. Examples of DDL are any SQL statements that begin with **create, alter, drop,** and **grant.** Table 2-1 is a sample list of some DDL statements. It does not completely represent the many varied statements that all have a unique purpose and value.

SQL Command	Purpose
create table	Creates a table
create index	Creates an index
alter table	Adds a column, redefines an existing column, changes storage allocation
drop table	Drops a table
grant	Grants privileges or roles to a user or another role
truncate	Removes all rows from a table
revoke	Removes privileges from a user or a role
analyze	Gathers performance statistics on database objects for use by the cost-based optimizer

TABLE 2-1. *Common Formats of Date Type Data*

■

All DML commands require reference to an object that will be manipulated. More often than not, the object being referenced is a table.

■

A conditional statement can be added to any **select, update,** or **delete** command. Absence of a conditional statement means that the command will be performed against every record in the object. A conditional statement is used when the DML command is intended to only act upon a group of records that meet a specific condition. The **where** clause will be discussed a little later in this chapter.

More optional DML statements will be described later in this chapter. For now, let's concentrate on understanding the fundamental structure of each DML statement starting with the **insert** and **select** statements.

CRITICAL SKILL 2.2
Use Basic insert and select Statements

Getting data into and out of a database are two of the most important features of a database. Oracle provides two basic features that help you do just that. To get data into the database, use the **insert** command; to get it back out, use the **select** command. You must master these commands, as they form the basic of most data access to your Oracle database. This section talks first about how to get data into your database, and then how to get data out.

insert

Using the state table created in the DDL example, the following is an illustration of using the **insert** statement in its simplest form:

CRITICAL SKILL 2.3
Use Simple where Clauses

Up to now, you have seen how the **select** command can be used to retrieve records from a table. However, our basic examples have all retrieved every record from the table. If you want to see only certain rows, you must add a **where** clause.

Since our previous examples returned every record in the table, we created a simple table with a few rows in it for illustration purposes. Had we chosen to illustrate the **select** command against the large sample tables provided by Oracle, we would have returned thousands of rows far too many for listing in this chapter. Now that we are introducing the **where** clause, we will be able to control the output. As a result, the remaining examples in this chapter will now use the customers, products, sales, and costs tables that are part of the Oracle sample database. Let's describe each of these tables.

between A and B	Greater than or equal to A and less than or equal to B.	select * from sales where amount_sold is between 100 and 500;
not between A and B	Not greater than or equal to A, and not less than or equal to B.	select * from sales where amount_sold is not between 100 and 500;
like '%tin%'	Contains given text (for example, 'tin').	select * from customer where cust_last_name is like '%tin%';

TABLE 2-2. *Common Comparison Operators*

CRITICAL SKILL 2.4
Use Basic update and delete Statements

While **select** will likely be the command you use the most; you'll use the **update** and **delete** commands regularly, too. As you will in Chapter 6, your programs will have a mixture of DML statements. In this section, we'll take a closer look at the **update** and **delete** commands.

update

It is often necessary to change data stored within a table. This is done using the **update** command. There are three parts to this command:

1. The word **update** followed by the table to which you want to apply the change. This part is mandatory.

2. The word **set** followed by one or more columns in which you want to change the values. This part is also mandatory.

3. A **where** clause followed by selection criteria. This is optional.

Let's imagine that one of our customers has requested an increase in their credit limit and our accounting department has approved it. An update statement will have to be executed to alter the credit limit. For illustration purposes, a customer record will be displayed before and after the update. The following example illustrates a simple update for one customer:

CRITICAL SKILL 2.5
Order Data

So far, all of our **select** queries have returned records in random order. Earlier, we selected records from the customer table where the customer was located in either Connecticut or Utah and had a credit limit of $15,000. The results came back in no apparent order. It is often desirable to order the result set on one or more of the selected columns. In this case, it probably would have been easier to interpret the results if they were sorted by state, and within that state were then sorted by customer ID. Let's take a look at the query syntax and resulting output:

CRITICAL SKILL 2.7
Use Dates and Data Functions (Formatting and Chronological)

Date is the next commonest type of data you'll find in an Oracle database after character and numeric data. The date data type consists of two principal elements: date and time. It's important to keep in mind that the date data type includes time when comparing two dates with each other for equality.

The default date format in many Oracle databases is DD-MON-YY, where DD represents the day, MON is the month and YY is the two-digit year. A date can be inserted into a table without specifying either the four-digit year or a value for the time element. Oracle will default the century to '20' for years '00 49' and '19' for years '50 99'. Without a specific time being specified during an insert, the time will default to midnight, which is represented as '00:00:00'.

Date Functions

As with the numeric and character data types, Oracle has provided many date functions to help with the manipulation of date data. If you were to routinely print customized letters to your best customers offering them a special deal that expires on the last day of the month, the last_day function could be used to automatically generate the expiration date for the offer. Table 2-6 shows the commonest date functions.

Function	Action	Example	Displays
Sysdate	Returns current system date. Time could also be retrieved using the to_char function, which is discussed in the next section.	select sysdate from dual;	17-MAR-04 on March 17, 2004
last_day(*date*)	Returns last day of the month for *date*.	select last_day('17-MAR-04') from dual;	31-MAR-04

CRITICAL SKILL 2.8
Employ Joins (ANSI vs. Oracle): Inner, Outer, Self

Up until now, all of the examples in this chapter have selected data from only one table. In actual fact, much of the data that we need is in two or more tables. The true power of a relational database (and the source of its name) comes from the ability to relate different tables and their data together. Understanding this concept is critical to harvesting the information held within the database. This is more commonly known as *joining* two or more tables.

With Oracle Database 10*g*, queries can be written using either Oracle's SQL syntax or ANSI syntax. While Oracle hasn't made ANSI syntax available until recently, it has been used in non-Oracle environments for some time. Many third-party tools accept ANSI SQL and, as you'll see shortly, the joins are quite different.

Inner Joins

An inner join, also known simply as join, occurs when records are selected from two tables and the values in one column from the first table are also found in a similar column in the second table. In effect, two or more tables are joined together based on common fields. These common fields are known as *keys*. There are two types of keys:

- A *primary key* is what makes a row of data unique within a table. In the CUSTOMERS table, CUST_ID is the primary key.

- A *foreign key* is the primary key of one table that is stored inside another table. The foreign key connects the two tables together. The SALES table also contains CUST_ID, which in the case of the SALES table, is a foreign key back to the CUSTOMERS table.

Oracle Inner Joins

The tables to be joined are listed in the **from** clause and then related together in the **where** clause. Whenever two or more tables are found in the **from** clause, a join happens. Additional conditions can still be specified in the **where** clause to limit which rows will be returned by the join. For example, when we queried the SALES table on its own, the only customer information available to us was the CUST_ID. However, if we join each record, we retrieve from the SALES table by the CUST_ID to the same column in the CUSTOMERS table, and all the customer information becomes available to us instantly.

Any of the combinations will produce exactly the same results. Let's hold off on the *left outer join* example until we revisit the outer join idea later in Project 2-4.

ANSI Full Outer Joins A *full outer join* is possible when using the ANSI syntax without having to write too much code. With a *full outer join*, you will be able to return both the *right outer join* and *left outer join* results from the same query.

The *full outer join* queries can be written as **full outer join** or **full join** and once again, the **on, using,** or **natural** joins are all possible. Let's revisit the Outer Joins Project and try the ANSI syntax out.

Project 2-2 Joining Data Using ANSI SQL Joins

Using the temp1 and temp2 tables we created and populated, let's try out the ANSI *right, left*, and *full outer joins*.

Step by Step

We've just learned that you can write the ANSI *outer joins* with or without the **outer** keyword in each of the ANSI *right, left*, and *full outer joins*. We also learned that the ANSI **on, using,** and **natural** join syntax is available as well. The following step-by-step instructions use a combination of these for illustration purposes. Feel free to try alternate syntax, but we encourage you to adopt a consistent style to allow your code to be self-documenting and traceable by other developers.

1. Use the ANSI right outer join:

Project 2-3 Grouping Data in Your select Statements

One final example will demonstrate the grouping of multiple columns and more than one function being performed for each group. As we build on this example, we will introduce column aliases, a **round** function combined with an **avg** function and the use of a **substr** function, which will serve to select only a specified number of characters for the product subcategories and names results.

Step by Step

Let's start with the preceding **group by** example and build on it as we introduce some formatting and intermediate concepts. Look at the output each time and see how we are transforming it along the way. A final output listing has been provided at the end for you to compare against.

1. Start SQL*Plus and re-execute the preceding **group by** example:

CRITICAL SKILL 2.10
Learn Subqueries: Simple and Correlated Comparison with Joins

Within SQL, functionality exists to create subqueries, which are essentially queries within queries. This power capability makes it possible to produce results based on another result or set of results. Let's explore this concept a little further.

Simple Subquery

Without the functionality of subqueries, it would take a couple SQL queries to retrieve product information for the product with the maximum list price. The first query would have to find the value of **max(prod_list_price)**. A subsequent query would have to use the value resolved for **max(prod_list_price)** to find the product details. Let's take a look at how we can resolve this with a subquery embedded in the **where** clause of the main query:

CRITICAL SKILL 2.11
Use Set Operators: Union, Intersect, Minus

One of the nice things about a relational database is that SQL queries act upon sets of data versus a single row of data. Oracle provides us with a series of set functions that can be used to join sets together, for example. The set functions will be discussed in the next few sections using two single column tables: table x and table y. Before proceeding to the discussion on the set functions, let's first take a look at the contents of these tables.

Table x:

NOTE
Please be aware that the ***intersect*** *set operator can introduce major performance problems. If you are venturing down this path, weigh the alternatives first*.

minus

The **minus** set function returns all the rows in the first table minus the rows in the first table that are also in the second table. The order of the tables is important. Pay close attention to the order of the tables and the different results in these two query examples:

CRITICAL SKILL 2.12
Use Views

Views are database objects that are based on one or more tables. They allow the user to create a pseudo-table that has no data. The view consists solely of an SQL query that retrieves specific columns and rows. The data that is retrieved by a view is presented like a table.

Views can provide a level of security, making only certain rows and columns from one or more tables available to the end user. We could hide the underlying tables, CUSTOMERS and SALES from all the users in our organization and only make available the data for states they are entitled to see. In the following example, we are creating a view to only show specific details about Utah-based customers sales:

rows and columns from this view. The second example selects only the name and total columns for customers whose sales are greater than 20,000. Keep in mind, this is still only for Utah customers.

Sequences are objects in the database that can be used to provide sequentially generated integers. Without these valuable objects available to users, generating values sequentially would only be possible through the use of programs.

Sequences are generally created and named by a DBA. Among the attributes that can be defined when creating a sequence are a minimum value, a maximum value, a number to increment by and a number to start with. They are then made available to the systems applications and users that would need to generate them.

For the following example, we have established a **cust_id_seq** sequence, which increments by one each time it's called. When we created the sequence, we specified that 104501 should be the number to start with. For demonstration purposes, we'll use the DUAL table to select the next two sequence numbers. More often than not, an application will retrieve and assign the sequence numbers as records are inserted into the associated table.

Ask the Expert

Q: Is a single space an acceptable entry into a NOT NULL constrained column?

A: Yes. Oracle will allow you to enter a space as the sole character in a NOT NULL constrained column. Be careful though. The single space will look like a NULL value when a **select** statement retrieves and displays this row. The space is very different than a NULL.

Deferred

When constraints are created, they can be created either as **deferrable** or **not deferrable**. A constraint that is not deferred is checked immediately upon execution of each statement and if the constraint is violated, it is immediately rolled back. A constraint that is deferred will not be checked until a **commit** statement is issued. This is useful when inserting rows or updating values that reference other values that do not exist but are part of the overall batch of statements. By deferring the constraint checking until the **commit** is issued, we can complete the entire batch of entries before determining if there are any constraint violations.

CRITICAL SKILL 2.15
Format Your Output with SQL*Plus

Throughout this chapter, we've seen the results of many SQL queries. In some, we added functions like **substr** to reduce the size of the columns and keep the results confined within one line. In SQL*Plus, there are many parameters that can be set to control how the output is displayed. A list of all of the available settings is easily obtained by issuing the **show all** command within SQL*Plus. Alternatively, if you know the parameter and want to see its current value, the command **show parameter_name** will give you the answer. Before we close out this chapter, let's visit a number of the more useful SQL*Plus parameters.

Page and Line Size

The **set linesize** command tells Oracle how wide the line output is before wrapping the results to the next line. To set the line size to 100, enter the command **set linesize 100**. There is no semicolon required to end **set** commands.

Ask the Expert

Q: Once I set parameters, do I ever have to set them again?

A: Yes. Parameters are good only for the current setting. The parameters always reset to their default settings when you start up a new SQL*Plus session. However, the parameter defaults can be overwritten at the start of each SQL*Plus session by entering and saving them in the login.sql file.

The **set pagesize** command determines the length of the page. The default page size is 14 lines. If you don't want to repeat the result headings every 14 lines, use this command. If you want your page to be 50 lines long, issue the command **set pagesize 50**.

Page Titles

The **ttitle** command includes a number of options. The default settings return the date and page number on every page followed by the title text centered on the next line. Multiple headings can also be produced by separating the text with the vertical bar character. The command **ttitle 'Customer List Utah'** centers the text "Customer List" on the first line followed by "Utah" on the second line.

Page Footers

The **btitle** command will center text at the bottom of the page. The command **btitle 'sample.sql'** places the text "sample.sql" at the bottom center of the output listing. The command **btitle left 'sample.sql'** results in the footer text "sample.sql" being placed at the left edge of the footer.

Formatting Columns

Quite often, you'll need to format the actual column data. The **column** command is used to accomplish this. Suppose we are going to select the last name from the CUSTOMERS table along with a number of other columns. We know that, by default, the last name data will take up more space than it needs. The command **column cust_last_name format a12 wrap heading 'Last Name'** tells SQL*Plus that there should be only 12 characters of the last name displayed and that the column title 'Last Name' should be displayed on two separate lines.

Project 2-5 Formatting Your SQL Output

Let's put these SQL*Plus concepts together and format the output of a SQL query. The following step-by-step instructions will lead you through a few of these basic formatting commands.

CHAPTER 3
The Database Administrator

CRITICAL SKILLS

3.1 Learn the Job of the DBA

3.2 Understand the Oracle Database 10*g* DBA Skill Set

3.3 Perform Day-to-Day Operations

3.4 Understand the Oracle Database 10*g* Infrastructure

3.5 Operate Modes of an Oracle Database 10*g*

3.6 Get Started with Oracle Enterprise Manager

3.7 Manage Database Objects

3.8 Manage Space

3.9 Manage Users

3.10 Manage Privileges for Database Users

So, you've decided to be a *Database Administrator (DBA)*. Great choice! On top of that, you've chosen Oracle as the *Database Management System (DBMS)* that you want to work with. Even better! All you need to do now is figure out how to learn what you need to know to do the job. Reading this book is a great start. However, the job of a DBA cannot be learned entirely in a few short months. It is a work in progress that can take several years to become really good at. Don't get us wrong you can learn the basics that will make you a productive DBA in a few short months, but there is a great deal to learn, and we don't become really good at this job until we've actually run the utility, executed the SQL, or performed the task. In other words, don't just read this book try the examples and don't be afraid to make mistakes.

CRITICAL SKILL 3.1
Learn the Job of the DBA

The role of a DBA is more of a career than a job. Those of us who have been doing this for many years are always learning new things and just trying to keep up! That's the exciting thing about being a DBA: the job keeps changing. Databases are growing at a phenomenal pace, the number of users is increasing, availability requirements are striving for that magical 24/7 mark, and security has become a much greater concern. As you will see in this book, databases now include more than just data. They are also about the Internet and grid computing and XML and Java. So, how long will it take you to learn how to be a DBA? For as long as you're practicing this career.

There are some concrete steps that you can take to jump-start your learning process. Undertaking an Oracle Certification will provide you with a structured program that offers you clear steps to help learn the details of the job. Instructor-led courses as well as CD- and Internet-based classes can help you through the process. Also, read as much as you can and then get your hands on a test database and practice what you've learned.

Applications come and go, but data stays around. All of the information that makes your company valuable is (or should be) stored in a database. Customer, vendor, employee, and financial data, as well as every other corporate data is stored in a database, and your company would have great difficulty surviving if any of that data was lost. Learn your job well. People are depending on you.

CRITICAL SKILL 3.2
Understand the Oracle Database 10*g* DBA Skill Set

There is good news for DBAs: Oracle has tools to help you do your job and manage your databases. These tools have existed for many versions of Oracle and have improved with each release to the point where the Oracle Database 10*g* offerings are

extensive. In many cases, you will have the option of doing your job using a *Graphical User Interface (GUI)*, and you will also have the option of using a command-line interface. We recommend learning both. You will need to use the command-line interface in many cases to schedule work through scripts. The GUI can be used for performing day-to-day operations and can also be used as a great learning tool the first time you perform an operation. In many cases, you will be able to generate the low-level commands from the GUI and can copy them to a file to be used later on.

As we've mentioned, there is a great deal that you will need to know in order to be able to provide well-rounded coverage of your Oracle environment. We can categorize the specialized areas of database management so that you will be aware of the whole picture and can break your work into well-defined groupings.

CRITICAL SKILL 3.3
Perform Day-to-Day Operations

In order to properly perform the role of Database Administrator, you will need to develop and implement solutions that cover all areas of this discipline. The amazing part of this job is that you may be asked to do many, or perhaps all, aspects of your job on any given day. Your daily tasks will vary from doing high-level architecture and design to performing low-level tasks. Let's take a look at the things that you will be getting involved in.

Architecture and Design

DBAs should be involved with the architecture and design of new applications, databases, and even technical infrastructure changes. Decisions made here will have a large impact on database performance and scalability and database knowledge will help choose a better technical implementation. Data design tools such as Oracle Designer can assist the DBA.

Capacity Planning

Short and long range planning needs to be performed on your databases and applications. This will focus on performance and sizing characteristics of your systems that will help to determine upcoming storage, CPU, memory, and network needs. This is an area that is often neglected and can lead to big problems if it is not done properly.

Backup and Recovery

A backup and recovery plan is, of course, critical in order to protect your corporate data. You need to ensure that data can be recovered quickly to the nearest point in time as possible. There is also a performance aspect to this since backups must be

TIP
Do not reorg unless you absolutely need to.

Change Management

Being able to upgrade or change the database is a discipline that includes many areas. Upgrades to the database schema, procedural logic in the database, and database software must all be performed in a controlled manner. Change control procedures and tools such as Oracle's Change Manager and third-party offerings will assist you.

Schedule Jobs

Oracle Database 10*g* comes with a new scheduler that allows you to schedule a job for a specific date and time, and to categorize jobs into job classes that can be prioritized. So, resources can be controlled by job class. Of course, other native scheduling systems such as "at" in Windows and crontab in UNIX can be used as well as other third-party offerings.

Network Management

Oracle Networking is a fundamental component of the database that you will need to become comfortable with. Database connectivity options like Tnsnames, the Oracle Internet Directory (OID), and the Oracle Listener require planning to ensure that performance and security requirements are met in a way that is simple to manage. You will see more of this in the next chapter.

Troubleshooting

Though troubleshooting may not be what you'd consider a classic area of Database Management, it is one area that you will encounter daily. You will need tools to help you with this. Oracle MetaLink technical support, available to customers who purchase the service, is invaluable. Oracle alert logs and dump files will also help you greatly. Experience will be your biggest ally here and the sooner you dive into database support, the faster you will progress.

You've seen the areas of database management that need to be handled, now it's time to look at the Oracle schema and storage infrastructure.

CRITICAL SKILL 3.4
Understand the Oracle Database 10*g* Infrastructure

Oracle's memory and process infrastructure have already been discussed in Chapter 1. In this section, we will take a look at the Oracle schema and storage infrastructure since these are a large part of what you will be required to manage.

Storage Structures

As shown in Figure 3-1, the physical schema objects are stored as segments in the database. Each segment can only be stored in a single tablespace and a tablespace can be made up of one or more datafiles. If a tablespace is running out of space, you can expand the datafiles it is made up of, or you can also add a new datafile to the tablespace. A datafile can only store data for a single tablespace.

A single tablespace can store data for multiple segments and in fact for several segment types. Segments from multiple schemas can also exist in the same tablespace. So, for example, table_a from schema1 and index_b from schema2 can both be implemented in the same tablespace. Oh and by the way, a tablespace can only store data for a single database.

The logical structures such as views and source code are stored in the Oracle catalog but are part of a schema. So, this means that the Oracle-supplied SH schema can contain all of the objects that it needs to run the entire application under its own schema name. This provides strong security and management benefits.

Progress Check 

1. Name five areas that you will need to address as a DBA.

2. What is a schema and what does it contain?

3. Can a tablespace store more than one segment type?

4. Under which circumstances would you bother to start up the database in nomount mode?

CRITICAL SKILL 3.5
Operate Modes of an Oracle Database 10*g*

Oracle is a software package like many others that you may have used. However, when you run most programs, they run one and only one way. So when I open my accounting software, I run it the same way all the time. However, you have options

Progress Check Answers

1. As a DBA, you will need to address architecture, capacity planning, backup and recovery, and security and performance, among others.

2. A schema is a logical structure that contains objects like segments, views, procedures, functions, packages, triggers, user-defined objects, collection types, sequences, synonyms, and database links.

3. A single tablespace can store data for multiple segments and different segment types. Segments from multiple schemas can also exist in the same tablespace. So, for example, table_a from schema1 and index_b from schema2 can be implemented as two segments in the same tablespace.

4. When started in nomount mode, the parameter file is read and memory structures and processes are started for the instance. The database is not yet associated with the instance. You will use this in cases where you need to re-create the controlfile.

Previous Next

FIGURE 3-3. *Enterprise Manager instance configuration view*

CRITICAL SKILL 3.6
Get Started with Oracle Enterprise Manager

Oracle Enterprise Manager (OEM) is a great tool to assist the beginner DBA through to the experienced one. You should, however, also learn the low-level commands that will allow you to do your job through an interface like SQL*Plus. OEM can help you with this by showing you the SQL that it has generated when you select the Show Sql button that exists on many windows. Given how many options OEM has to help you do your job as a DBA, we will take a quick look at them here.

First off, OEM can be used to manage all of the databases in your network. As you can see in Figure 3-3, once you expand the network, you can select Databases

Database change management can be performed through the Change Management Pack. Under Tools, choose Change Management Pack or Standard Management Features and these will lead you to the Change Manager utility.

Database applications will provide support for the spatial index advisor, SQL*Plus Worksheet, and the Oracle Text Manager.

Tuning facilities such as performance manager, outline management, and tablespace maps are provided through Standard Management Features and Tuning Features, as shown in Figure 3-6.

As you can see from this overview of OEM console capabilities, many of the tools that we need to perform our day-to-day tasks can be found in this one console. Now that we have confidence that there's a toolset to support us, let's take a quick look at what you need to think about when managing database objects.

CRITICAL SKILL 3.7
Manage Database Objects

A large part of your job as a DBA will be to manage the objects that exist in a database. Let's look at the objects that you need to concern yourself with and discuss the main management issues that you will have in each of these areas.

Controlfiles

It is critical to the database that you have at least one valid control file for your database. These are small files and can be multiplexed by the Oracle instance. Ensuring that you have at least three copies of the controlfiles (remember, they are small), as well as text and binary backups whenever a data file, log file, or tablespace is changed and on a regularly scheduled basis (at least daily) will go a long way towards ensuring that your control files are in good shape. Controlfiles will be discussed in more detail in Chapter 5.

Redo Logs

Redo logs are necessary to ensure database integrity and should be duplexed in Oracle. Oracle mirroring helps even if your redo logs are mirrored by your storage subsystem since Oracle will use the alternate redo log if one should become corrupt. You will need to ensure that you have enough redo logs and that they are sized properly to support database performance. How large should your redo logs be? They should be large enough that a log switch does not usually occur more than once every 15 minutes due to the checkpointing that occurs during a log switch and the overhead that is incurred during this operation. How many redo logs should you have? You should have enough redo logs that the system will not wrap around to a log that has not yet completed a checkpoint or completed archiving (for systems in archivelog mode). Redo logs can be added, deleted, and switched through OEM.

Ask the Expert

Q: Why is it important for DBAs to get involved with the architecture and design of a new system?

A: Decisions made on the technical infrastructure as well as data and application designs here will have a large impact on database performance and scalability. Database knowledge will help choose a better technical implementation. Once chosen, these can be difficult to change.

Q: Which method do you normally use to shut down a database?

A: Although the **shutdown normal** operation is a recommended approach, it is often impractical since you need users to disconnect themselves. The approach that I prefer is to perform a checkpoint using the command **alter system checkpoint** which will write data out to data files and speed up the restart. I then perform a **shutdown abort**, immediately followed by a **startup restrict**, and **shutdown immediate**. This is a fast, guaranteed shutdown that leaves the database in a consistent state once all of the steps have been completed.

Q: What is the best way to become a good Oracle DBA quickly and then to keep improving?

A: There are many things that you will need to do and many skills that you'll need to develop to do this job. First, learning the basic DBA skills, which you can get from books such as this as well as from courses, will give you a head start. Practicing what you see is probably the quickest and most practical way to learn. Getting involved in supporting some databases in development and production will force you to learn very quickly. Then working on development systems for different types of applications will help to round out your skills. Keep reading and learning and never assume that you know it all and you will do very well.

CRITICAL SKILL 3.8
Manage Space

The challenge of managing data in your Oracle Database 10*g* is one that provides you with options. In this section, we will look at the methods that have been used in the many versions of the database to manage your information. Today's version of the

Progress Check

1. What's better: "shutdown transactional" or "shutdown immediate"?

2. Do you only need to worry about logical schema objects that do not take up a large amount of space?

3. What happens if your archive log directory becomes full?

4. Why would you want to use a command-line interface rather than a GUI to perform your tasks as a DBA?

CRITICAL SKILL 3.9
Manage Users

Before you can do anything in Oracle, you need to have a user ID created to enable you to log in to Oracle. As a DBA, you will begin with the SYS or SYSTEM accounts since these accounts both have the DBA role and exist in all Oracle databases. They are often used to perform database administration tasks. The SYS account is also granted the **sysdba** privilege and is the schema that the Oracle catalog is stored in. You should only use the SYS account when you need to perform a task as SYS or need the **sysdba** privilege. If your database was created using the Database Configuration Assistant (dbca), then you will also automatically get the SYSMAN and DBSNMP accounts. SYSMAN is used to administer Oracle Enterprise Manager (OEM) and DBSNMP is used by the agent that OEM employs to monitor Oracle databases. Several other accounts will also be set up for the "example" schemas, such as the Sales History ('SH') user that we will utilize throughout this book. The OUTLN schema will be created to allow you to use plan stability through the stored outline feature. Depending on the options you choose when creating your database, other accounts may be set up for you. For example, if you install the OLAP option, the OLAPSYS account will be created.

Progress Check Answers

1. Both leave your database in a consistent state. It depends on how long your transactions will take to complete or roll back. If all things are equal and you think that it will take as long to commit the transactions that are already running, then you should use "shutdown transactional" since commits will be allowed to complete and no data will be lost.

2. Logical schema objects need to be watched to ensure they are in a valid state.

3. If database attempts to write an archive log after the directory has become full, the database activity will be suspended until sufficient space is made available for the new file.

4. You may want to place the command in a script that is scheduled or run as a repetitive task.

CRITICAL SKILL 3.10
Manage Privileges for Database Users

Creating a user in Oracle has accomplished the first part of user setup and that is authentication. We have a user ID and password and have authorized this user to use an Oracle database. Once the user logs in, however, they will not be able to do very much because they will not have privileges that allow them to access any objects. This leads us to the second step of setting up a user: authorization. In order to authorize a user to perform their tasks, we need to grant access.

Grant Authority

You now need to give permission to the user to do things in Oracle. Actions like accessing a table or executing a procedure or running a utility require you to "grant" the authority to that user. When you perform a grant, you can specify four things:

- The user that is being granted the authority.
- The object that is being granted. Examples of these are a table, procedure, or role.
- The type of access being granted, such as select, insert, update, or delete on a table, or execute on a procedure, function, or package.
- Whether this user has authority to then grant the same authority to other users. By default, they do not, but this can be added by using the With Grant option.

Here are two examples that grant a user "NEWUSER" access to a table and then to a package.

In Conclusion

As you have seen in this chapter, there is a great deal that a DBA needs to be aware of to properly manage a database. The good news is that you will have tools such as OEM to help you. Do your best to keep your environment as simple as you possibly can! You will be glad that you did as your overall database environment continues to grow.

Project 3-1 Creating Essential Objects

This project will walk you through the creation of the essential storage and schema objects after a database has been created, which in this project will be called ora10g. You will create a new tablespace called **NEW_TS** and will than add a user **NEW_USER** who will be given the authority to this tablespace. You will then create a role called **NEW_ROLE** and grant privileges to it. Afterward, you'll grant this role to the new user. A table and index will be created on this tablespace by the new user. Lastly, you will resize the undo tablespace to make it larger. You will see how to do this in OEM and the generated SQL will also be shown to you so you can do this in SQL*Plus.

Step by Step

1. You have been asked to create a new used named **NEW_USER** who will need to create objects in a new tablespace called **NEW_TS** that should be sized at 5MB. Your first step will be to create the tablespace. In OEM, log in as user SYSTEM, go to database ora10g, choose storage, then choose tablespace and select an existing tablespace to model. Under Objects in the toolbar, select the Create Like option to model your new tablespace after the existing one. Enter the new tablespace name, datafile name, and all properties including the size. Make this a locally managed tablespace 5MB in size with uniform extents 96KB in size. If you choose the Show Sql button, you will see the generated SQL. It should look something like the following SQL. You can either apply the change in OEM or you can copy and paste the generated SQL and run it in SQL*Plus.

CHAPTER 4
Networking

CRITICAL SKILLS

4.1 Use Oracle Net Services

4.2 Learn the Difference Between Dedicated and Shared Server Architectures

4.3 Define Connections

4.4 Use the Oracle Net Listener

4.5 Learn Naming Methods

4.6 Use Oracle Configuration Files

4.7 Use Administration Tools

4.8 Use Profiles

4.9 Network in a Multitiered Environment

This chapter introduces Oracle Net Services, which allow database applications running on remote systems to access an Oracle database. It creates and maintains the network connection, and also exchanges data between the application and the database.

Oracle networking plays a critical role in performance and availability. Each new version of Oracle is designed to support more data and users than the previous release. This increased amount of database activity and network traffic needs to be addressed from an availability and performance perspective and should be managed by the DBA. A DBA also has to be able to determine if a performance issue is due to networking, and if so, then they must be able to resolve any network performance issues from a database configuration perspective.

Throughout this chapter we will refer to DBAs, which in this context means anyone that is performing networking administration operations to make the database connectivity work. These days, more developers are managing their own development databases and performing operations traditionally reserved for DBAs.

NOTE
Oracle Net Services is a large topic. The emphasis in this chapter is to introduce DBAs to Oracle Net Services terminology and concepts, feature/functionality, and key components and tools. Once a beginning DBA reads this section, they should be able to understand the Oracle networking references and be capable of performing simple operations using the Oracle GUI tools and wizards for Oracle Net Services.

CRITICAL SKILL 4.1
Use Oracle Net Services

Oracle Net Services is the software component that allows enterprise connectivity across heterogeneous environments. Oracle Net is the part of Oracle Net Services that manages data communication between a remote application and the Oracle database, and runs on top of a network protocol like TCP/IP. The software used by Oracle Net software resides on the remote system and the Oracle database platform.

A listener process must be running on the database server to receive the network request. (A listener is a program that listens on a port for incoming network requests and then hands the request to another program for processing.) The listener then determines the appropriate type of process to handle the request.

The network protocol sends a request to the Oracle Protocol layer, which sends the information to the Oracle Net Foundation layer, which then communicates with the database server. The Oracle network communication stack, shown in Figure 4-1, is similar on both the client- and server-side.

Define a Location

Locations need to be defined so a remote application can find the correct Oracle database server on the network. A service name, such as **customer.us.trubix.com**, is used to define the unique location of each database server. In the preceding example, **customer** is the database name and **us.trubix.com** is the domain name. On the plus side, if the physical location of the database is changed, the service name can stay the same.

A database can support multiple services. The service name, defined with the initialization parameter **SERVICE_NAMES**, makes the physical location of the database transparent and will default to the global database name (the name of your database), which uses the format database_name.database_domain, as in customer.us.trubix.com.

The database domain name is the domain where the database is located, and is made up of the initialization parameters **DB_NAME** and **DB_DOMAIN**. The combination of the **DB_NAME** and **DB_DOMAIN** (customer.us.trubix.com) name distinguishes one database from another, as shown in the following examples:

a look at the data in these tables before and after you implement shared servers, to see how they change your database and how it functions.

CRITICAL SKILL 4.3
Define Connections

This section will discuss the core components required to handle Oracle connections.

A Connect Descriptor

A connect descriptor is used to define the service name and the location of the database. The address component of a connect descriptor defines the protocol, host name, and port number. Though port numbers can be between 1 to 65535, those from 1 to 1024 are usually reserved for special processes. The connect data component of the description describes the service to which you want to connect. If you do not include the **instance_name** in your descriptor, it will default to the Oracle SID if not defined.

A sample connect descriptor for **customer.us.trubix.com** looks like the following:

CRITICAL SKILL 4.4
Use the Oracle Net Listener

The Oracle Net Listener (listener) listens on a network port (listening endpoints) for incoming database requests. A listening endpoint defines the protocol addresses the listener is defined to listen on. Listening endpoints include HTTP, FTP, WebDAV, and Oracle XML. Look at the *ORACLE XML DB Developer's Guide* for more detail on registering FTP, HTTP, and WebDAV listening points.

The process is fairly simple. The listener receives a request and hands the request to a service handler, which is a server process that runs on the same platform as the Oracle database server. The service handler can be a dedicated server or a dispatcher, the latter of which works with shared servers.

The PMON background process registers the service information to the listener. During registration, PMON gives the listener information on the database services and instance information. PMON then tries to register with the listener once the listener has been started. Dynamic registration is supported with the **alter system register** command. If PMON has not registered with the listener, a TNS listener error will occur. View the *Oracle Database 10*g *Error Messages* reference manual for more details.

The listener will receive the database request and spawn a dedicated server process if the environment is configured for the dedicated server architecture. The dispatcher will hand the request over to a dispatcher if running a shared server architecture. A client application can bypass the listener if it is running on the same platform as the database server. Once the listener hands off the request it will resume listening for additional network requests.

A default listener (named listener) is configured at installation with the Oracle Net Configuration Assistant making it easy to start up the default listener when a system is first built. An additional ICP protocol address is defined for external routes during installation.

Progress Check Answers

1. The protocol *WebDAV* supports collaborative authoring over the Internet.

2. False. The SDP protocol is used with high-speed networks.

3. True. A virtual circuit is a section of shared memory that contains information for client communication.

4. False. Ports 1 to 1024 are used for special processes. They are not reserved for SSL.

5. The *dedicated* server architecture does not support FTP, HTTP, or WebDAV clients.

6. True. Yes, this is one of the advantages of using the Oracle Connection Manager.

Multiple Listeners

Multiple listeners can be defined for a service, and offer a number of advantages for more complex environments. These advantages include the following:

- Failover
- Transparent application failover
- Load balancing

The following is a sample connect descriptor for a listener:

naming methods need to be configured. This method cannot be used if more advanced features are required.

The External Naming Method

The external naming method uses net service names that are defined in a non-Oracle environment. This naming method works well for administrators that want to use their native naming service, and allows them to use native tools and utilities with which they have experience. The disadvantage of this approach is that Oracle Net tools cannot be used for these native naming methods. Supported non-Oracle services include the Network Information Service (NIS) or Cell Directory Services (CDS). CDS is part of a Distributed Computing Environment (DCE) environment. DCE is an integrated distributed environment designed to resolve interoperability issues with heterogeneous environments. DCE is maintained by the Open Systems Foundation (OSF).

Which Naming Method to Use

The local naming method (tnsnames.ora) has traditionally been the most popular method. However, there are a number of administration and security issues in stored local configuration with a tnsnames.ora file. The directory (centralized) naming method is more scalable and has less administration than the local naming method. For large systems, the directory method is becoming more popular. Oracle Names is an Oracle proprietary centralized naming method and is no longer supported in Oracle Database 10*g*. Oracle Name environments should migrate to the directory naming method. The easy naming method and external naming method are not used as often.

CRITICAL SKILL 4.6
Use Oracle Configuration Files

Remote applications will look for Oracle Net configuration files to determine how to access the Oracle database server. Configuration files can be found in the ORACLE_HOME/network/admin directory location. Table 4-5 defines the primary configuration files.

Syntax for Configuration Files

DBAs can use the management tools to modify Oracle Net Services configurations. However, since the configuration files have a simple syntax, it is easy to modify the configuration files directly. The following is an example of the listener.ora file.

Should a DBA want to modify the files directly, the following syntax rules must be followed:

- Comments must begin with a pound sign ( ). Anything following the pound sign is treated as a comment.
- Keywords are not case sensitive and cannot contain spaces.
- Spaces are optional around equal (=) signs.
- Values can only contain spaces if they are surrounded by quotes. The values may be case sensitive depending on the operating system and protocol.
- A connect descriptor can be no more than 4KB in length.
- All characters must be part of the network set.

CRITICAL SKILL 4.7
Use Administration Tools

Oracle Net Services contains a number of user interfaces and tools that simplify the management of the Oracle network, including the following:

- Oracle Enterprise Manager (OEM)
- The OEM console
- Oracle Net Manager
- Oracle Net Configuration Assistant
- Oracle Connection Manager
- Oracle Internet Directory Configuration Assistant
- Command-line utilities

- Oracle Advanced Security

The Oracle Enterprise Manager

Along with database administration, OEM allows configuration of Oracle Net Services. OEM can be used to perform the following administration features:

- The configuration of listeners
- The configuration of naming definitions such as connect descriptors

Previous Next

Project 4-1 Testing a Connection

The following project will walk you through the steps of testing a connection to an Oracle database server.

Step by Step

The first step is to test the network connectivity between the remote system and the Oracle database server. The **ping** command will verify network access. If ping is successful, the remote system can resolve the name of the host server name. The host server name should be defined in the hosts file for the operating system.

The hosts file in UNIX is in the /etc directory; the hosts files in Windows is in the\winnt directory. The following is an example hosts file entry.

CRITICAL SKILL 4.8
Use Profiles

A profile contains a set of parameters that define Oracle Net options on the remote or database server. Profiles are stored in the sqlnet.ora file and can be used to

- Route connections to specific processes
- Control access through protocol-specific parameters
- Prioritize naming methods
- Control logging and tracing features
- Configure for external naming
- Define how the client domain should append to unqualified names

During installation, the priority order for the naming methods will be defined. If the first naming method cannot resolve the connect identifier, the next naming method will be checked. The results will then be stored in the sqlnet.ora file, as shown in the following example:

The Oracle Net Manager is used to define database access control. To define database access control, perform the following steps using Oracle Net Manager:

1. After starting Net Manager, select Local Profile.

2. Choose General.

3. Select Access Rights.

4. Choose Check TCP/IP Client Access Rights.

5. In the Clients Excluded From Access and the Clients Allowed To Access fields access control can now be defined.

CRITICAL SKILL 4.9
Network in a Multitiered Environment

Although the Oracle Database 10*g* has new features that simplify database administration, the environment the database server runs in is becoming more complex. The following areas continue to increase the complexity of Oracle networking environments:

- Oracle Database 10*g*-supporting HTTP, FTP, and WebDAV protocols are changing how data is used and accessed.
- The OEM Central Console is changing how Oracle DBAs perform administration across multiple databases.
- Multitiered architectures are placing increasing demands on network performance and security.

Traditionally, most Oracle networks have been set with the local naming method. In the future, more Oracle networking environments will work with multitiered architectures, the Oracle OEM Central Console, encryption, and the directory naming method. Companies are going to need people with skills to manage these complex environments. This chapter introduced you to the main components of Oracle Net Services. To begin working with Oracle Net Services, you may want to look at the following areas in the following order in terms of developing your skills:

- Strengthen your understanding of the Oracle Net Services architecture.
-

Obtain a solid understanding of configuring dedicated and shared server environments.

■

Become comfortable working with listeners and the local naming method.

Previous Next

CHAPTER 5
Backup and Recovery

CRITICAL SKILLS

5.1 Oracle Backup and Recovery Fundamentals

5.2 Learn about Oracle User-Managed Backup and Recovery

5.3 Write a Database Backup

5.4 Back Up Archived Redo Logs

5.5 Get Started with Oracle Data Pump

5.6 Use Oracle Data Pump Export

5.7 Work with Oracle Data Pump Import

5.8 Use Traditional Export and Import

5.9 Get Started with Recovery Manager

This chapter discusses many concepts that are very important to Oracle DBAs and users. Backing up your data is crucial, and this chapter discusses how to do so as well as how to recover when things go wrong. As we have said before, the best way to learn is to do, and backup and recovery are tasks that every DBA must learn and, more important, practice. Just remember that if you do plan to perform the exercises and examples in this chapter, do it on a database that is not being used, or create one just for this purpose...just in case.

CRITICAL SKILL 5.1
Oracle Backup and Recovery Fundamentals

As you should already know, data is a valuable asset. To ensure that you can protect your investment, it is important to insure your valuable property. To support this important data need, Oracle Database 10*g* provides numerous features to enable you to protect your investment. The ability to back up your data in case of a failure is an invaluable capability. Now you have the chance to back up your data without interruption to your business processes. Just as important as it is to back up your data is the ability to quickly recovery from a failure. Whether you lose data due to hardware, software, or human failures, the time to recover costs businesses opportunity and money. This chapter introduces you to how Oracle supports the need to back up and recover data.

Where Do I Start?

Oracle's implementation of backup and recovery is an extensive one that provides you with the advantage of having many options that you can use. This is a good thing, but can leave you wondering, "Where should I start and which options are best for me?" This chapter will take you on a quick tour of backup and recovery and should leave you with a good understanding of how this is implemented in Oracle. As you review the backup and recovery utilities presented in this chapter, keep in mind that we strongly recommend using *Recovery Manager (RMAN)* for performing backup and recovery and will explain why later. But for now, just know that we're off to a good start since we've answered our first question already!

One of the most important elements of an advanced database management system (DBMS) is the capability to perform backup and recovery in a manner that guarantees data will not be lost. Oracle provides you with many options that can be used, from basic backup and recovery through advanced facilities to keep the database up in a high-availability environment. As a DBA, when you need to deal with a situation where the database is corrupted and needs to be recovered, there is nothing more comforting than knowing that you have valid backups for recovery and that you know how to use them!

CRITICAL SKILL 5.2
Learn about Oracle User-Managed Backup and Recovery

Oracle supports backing up data in a number of ways. This section discusses how and why to use user-managed backups. These backups are by nature handled more mechanically than other methods and are just as effective. Also presented here is the information needed to recover your first database. Please remember that you should try this on test databases before trying your first backup and ultimate recovery on a business database.

Types of User-Managed Backups

User-managed database backups can be performed as either cold or hot physical backups. A *cold backup* means that all users are disconnected from the database and it is shut down in order to perform the backup. A *hot backup* is performed while the database is up and end users can remain connected to the database. In fact, they can be changing the very data that is being backed up! Let's examine these in more detail.

Cold Backups

Cold backups are the simplest type of backup operation you can perform. Cold backups are performed with the database completely shut down in a consistent manner. Once that is done, all database files should be backed up to disk or tape. Once the file copies are complete, the database can be started and users can resume their activity. The database does not need to be in archivelog mode in order to perform a cold backup but without archive logging, the database can only be recovered to the point in time that the cold backup was done. Cold backups are a simple option and limited in the way they must be run, but once you have a cold backup, it can be easier to work with and can provide a fair degree of functionality.

In order to perform a cold backup, the database must be shut down in a consistent manner. In other words, the database should be shut down by issuing one of the following commands:

- **shutdown normal**
- **shutdown immediate**
- **shutdown transactional**

Do not perform a cold backup of the database immediately after a **shutdown abort**. If you must shut down the database in this manner, follow it up with a **startup restrict** and **shutdown [immediate, transactional, normal]**. In this way, you can be confident that you have a database in which all of the transactions have

Ask the Expert

Q: What would happen if I restored the redo logs before I performed a point-in-time recovery? It seems like this is the best approach to use.

A: There is an end-of-redo marker on the online redo logs that will stop the recovery immediately. Oracle will think the forward recovery is complete and archive logs will not be applied.

Q: How can I find out the state of a file when a backup control file was taken? I need to know which files are read-write, read-only, or offline, but how can I do that?

A: Whenever you back up a control file, run an sql script that queries **dba_data_files** and writes the status of all of your data files to a file that is kept with the backup control file. We will show you an example of this in the following section.

need to be performed when the database is opened and this will create new redo logs as well as a new version, or *incarnation*, of the database. You also need to know the status of data files when the backup control file was created. If a data file had a status of read-write when the backup control file was created, but should be opened as a read-only file, then it should be taken offline before recovery begins. Backup control files are a useful and sometimes required feature, but can complicate your database recovery.

TIP
Always back up the database after performing an incomplete recovery and opening the database with the Resetlogs option.

CRITICAL SKILL 5.3
Write a Database Backup

When you decide to use a user-managed backup strategy rather than RMAN, you will need to develop scripts to perform hot or cold backups. One of the most important things you should do to simplify your maintenance requirements is to develop scripts that are generated from the Oracle catalog. If done properly, you won't need to change your backup scripts every time you add a new file or change the location of a file.

- Using the Parallel parameter to define the maximum number of threads and degrees of parallelism for export and import jobs

- The ability to now monitor jobs by detaching and reattaching to running jobs

- The ability to estimate the amount of space an export file will occupy by using the **estimate_only** clause

The Data Pump Export and Import for data and metadata are performed using the Data Pump *application programming interface (API)* and uses procedures in the DBMS_DATAPUMP PL/SQL Package. The metadata API is implemented through the DBMS_METADATA package. This package retrieves metadata in XML format that can be used in many ways. For example, XML can be transformed into DDL or *XSLT (Extensible Stylesheet Language Transformation)* and the XML itself can be used to create an object. There is a new Remap attribute that allows the attributes of an object to be changed. For example, schema names can be changed using this feature. The Data Pump API supports all objects needed to perform a full export.

There are three ways to perform Data Pump Export and Import utilities. There is the command-line interface where export and import parameters are listed on a command line or in a script. A variation of the command-line interface is to add a parameter file using the **parfile** parameter, which points to a different file where all of the import or export parameters are listed. An interactive-command interface can also be used by entering CTRL-C during an import or export run which will then allow you to enter commands when prompted.

Let's now explore some details about actual running Data Pump exports and imports.

CRITICAL SKILL 5.6
Use Oracle Data Pump Export

There are five mutually exclusive modes for performing the Oracle Data Pump Export:

- Full Export is where the entire database is exported using the **full** parameter. This can be used to completely rebuild the database if needed.

- Schema Export is the default mode and allows you to export one or more schemas in the database. The **schemas** parameter is used to run this. Please note that objects in other schemas related or dependent on objects in this schema are not exported unless the other schema is also mentioned in the schema list.

- Table Export allows for the export of tables or partitions and their dependent objects using the **tables** parameter.

- Tablespace Export can be used to unload all of the tables that have been created in the given tablespace set using the

tablespaces parameter.

Previous Next

Here is an example of data filtering where the tables in the SH schema are exported except for the PROMOTIONS table and CUSTOMERS table which has one row exported.

CRITICAL SKILL 5.8
Use Traditional Export and Import

The original (non-Data Pump) Export and Import utilities that were used in pre-Oracle10*g* versions can be found in Oracle Database 10*g*. However, we strongly recommend you use the new Data Pump utilities since they support all Oracle Database 10*g* features and will increase performance. Here, we'll review the original Export and Import utilities since you'll be using them with earlier versions of Oracle.

How to Run the Original Utilities

Before running the original export and import, the catexp.sql catalog script needs to be run to prepare Oracle for these utilities, and it is invoked from the catalog.sql script. These scripts can be found in the ORACLE_HOME/rdbms/admin directory.

Once the catalog has been set up for export and import, you are ready to run the utilities. As with Data Pump, these utilities can be run as a command-line interface, by using parameter files or interactive commands.

To run an original export, issue the **exp** executable in a manner similar to using Data Pump. Use the following syntax to run a command-line export or an export using a parameter file or an interactive export, respectively:

This can be transformed to a *user* export by replacing **full=y** with the **owner** parameter:

also restored. We have changed the archive log directory destination to write the archive log restores to. Notice that there is no mention of file names in this script. The recovery catalog has kept track of all of the files for us and if files were stored on tape, the Media Management Layer software may have also assisted with this.

CHAPTER 6
PL/SQL

CRITICAL SKILLS

6.1 Define PL/SQL and Why We Use It

6.2 Describe the Basic PL/SQL Program Structure

6.3 Define PL/SQL Data Types

6.4 Write PL/SQL Programs in SQL*Plus

6.5 Handle Error Conditions in PL/SQL

6.6 Include Conditions in Your Programs

6.7 Create Stored Procedures How and Why

6.8 Create and Use Functions

6.9 Call PL/SQL Programs

The basic way we access data with Oracle is via SQL. It provides us with the ability to manage both the database and the information. However, you generally will find that SQL cannot do everything that the programmer needs to do. SQL has an inherent lack of procedural control of the output. (It has no array handling, looping constructs, and other programming language features.) PL/SQL can be regarded as an extension to SQL for fine control of database data processing. To address this need, Oracle developed PL/SQL Oracle's proprietary programming language.

To this end, Oracle provides us with a built-in programming language called Procedural Language for Structured Query Language (PL/SQL). PL/SQL, Oracle's contribution to the programming world, is a programming environment that resides directly in the database. We will discuss its architecture later in this chapter. First though, some background about this powerful programming environment.

PL/SQL first appeared in Oracle Version 6 in 1985. It was primarily used within Oracle's user interface product SQL*Forms to allow for the inclusion of complex logic within the forms; it replaced an odd step-method for logical control. It also provided a reasonably simple block-structured programming language that resembles ADA and C. We can use PL/SQL to read our data, perform logical tasks, populate our database, create stored objects, and even to display web pages. PL/SQL has certainly developed into a mature product, and Oracle has shown a very strong dedication to the language, as illustrated by its use of PL/SQL in many of its products (such as Oracle Applications). Oracle also uses the web extensions of PL/SQL quite extensively in many other applications and products.

In this chapter, we will discuss the basic concepts and constructs of PL/SQL so you'll understand how to create your own PL/SQL programs. There is a lot to cover, but, as important as it is to learn SQL, you need to know PL/SQL as well, because if you're looking to become a DBA or an Oracle developer, you must have knowledge of PL/SQL in your database toolkit.

CRITICAL SKILL 6.1
Define PL/SQL and Why We Use It

The Oracle Database 10g is more than just a database. It is also an engine for many programming languages. Not only does it serve as a Java engine with the built-in Java Virtual Machine (JVM), it's a PL/SQL engine as well. This means that the code you write may be stored in the database and then run as required.

The PL/SQL engine is bundled together with the database, and is an integral part of Oracle's database, providing you with a powerful language to empower your logic and data. Let's look at how PL/SQL fits into the Oracle database. Figure 6-1 shows you how PL/SQL works from within, and from outside, the database.

contained within it. The following PL/SQL programs can be called from these Oracle development environments:

- SQL*Plus
- Oracle Grid Control/Oracle Enterprise Manager
- Oracle Precompilers (such as Pro*C, Pro*COBOL, and so on)
- Oracle Call Interface (OCI)
- Server Manager
- Oracle Application Server 10g
- Java Virtual Machine (JVM)

As you can see, PL/SQL is well established within Oracle's line of products.

The reasons for using PL/SQL are primarily its tight integration with the database server and its ease of use. You will find that there are few tasks that PL/SQL cannot handle.

TIP
Use PL/SQL to program tasks that are complex or for program elements that may be used many times over and over again.

CRITICAL SKILL 6.2
Describe the Basic PL/SQL Program Structure

The structure we use in PL/SQL is the foundation for all of the language. Once you've mastered it, you will then be able to move forward; however, if you do not take the time to get this first step right, your journey will be difficult. Thankfully, it's quite simple.

The structure is quite basic. You will have areas for your program parameters (these are used to pass values from outside a program to the program itself), your internal variables, the main program code and logic, and various ways to deal with problem situations. Let's look at the basic form of a PL/SQL block:

CRITICAL SKILL 6.3
Define PL/SQL Data Types

The use of local variables within a PL/SQL program is an important knowledge point for everyone using the language. It is a basic component of each program and as such it is invaluable to gain the knowledge of what is available and how best to use it. We can now look at how we use and define variables and working storage within our PL/SQL programs.

The PL/SQL Character Set

As with all programming languages, there are characters that you use to write your programs. Each language has its own rules and restrictions when it comes to the valid characters. In the following sections, we will show you the following:

- Valid characters when programming in PL/SQL
- Arithmetic operators
- Relational operators

Supported Characters

When programming in PL/SQL, you may use only characters as defined here:

- Characters can be typed in either upper- or lowercase. PL/SQL is case insensitive.
- All digits between 0 and 9.
- The following symbols: () + * / = ! ; : . ' @ % , " ' ^ _ { } ? []

Some of these characters are for program commands; others serve as relational or arithmetic operators. Together they form a program.

Arithmetic Operators

Table 6-1 shows the common arithmetic operators used in PL/SQL. They are listed in the order of precedence in which they are executed (that is, by priority). When the functions appear in the same line, this means they are executed with the same level of precedence, so the position of the expression determines which goes first.

Table 6-2 shows the common relational operators used in PL/SQL. These are the logical variables that are used to compare data.

The use of variables in a PL/SQL program is usually something that is required to truly leverage the power of the language. It is here that we define how our data is to be held while we work it through our program. These variables can be the same types as we have already learned about in the SQL language. However, in addition to these standard data types, we have some special ones that have been created specifically for the PL/SQL language.

Previous Next

Progress Check

1. Name four programs or facilities where you can use PL/SQL.

2. Name three sections that may be contained in a PL/SQL block.

3. What is the only required section in a PL/SQL block?

4. What data type would you use to store each of the following?

A. 12344.50

B. True

C. April 11, 1963

D. PINK FLOYD

CRITICAL SKILL 6.4
Write PL/SQL Programs in SQL*Plus

When we write PL/SQL programs, we have a couple of options on how to run a program. A program may be run directly in SQL*Plus (or some other SQL environment), or it can be stored in the database and then run from a SQL environment or a program. When you store a program in the database, we call this a stored program or stored object. We'll cover this later in the chapter. For now let's discuss how to write a program using SQL*Plus.

Progress Check Answers

1. Any four from among the following would be acceptable answers: Oracle Forms, Reports, Warehouse Builder, Oracle Applications, Oracle Portal, SQL*Plus, Oracle Grid Control, Oracle Pre-compilers, and Oracle Application Server.

2. The three sections that may be contained in a PL/SQL block are the Declaration, Execution, and Exception sections.

3. The Execution section is the only required section in a PL/SQL block.

4. The data types used to store each of the variables would be

A. Number or number(8,2). The storage of a number should always be done in a number data type. You can specify the precision or simply define it as a number with no precision, when you do not know the exact nature of your data.

B. boolean. The boolean data type is used to store true and false information.

C. Date. The date data type stores date and time information.

D. varchar2(10). Character values should be stored in the varchar2 data type. This is more effective for storing the data, yet it has a limit of 4000 bytes. If you need more than 4000 bytes, you should then use the LONG data type, which allows you to store up to 2GB of data.

Previous Next

Now let's move on to illustrate how to construct a PL/SQL program and get some output of our results.

Project 6-1 Creating a PL/SQL Program

This will be the first PL/SQL program that you will create. The concept is straightforward. We will declare some variables, place some values into them, and then output the data to the screen with SQL*Plus.

Step by Step

1. Log into SQL*Plus.

2. At the SQL prompt, enter the serveroutput command: **set serveroutput on;**

3. Enter the following PL/SQL program:

CRITICAL SKILL 6.5
Handle Error Conditions in PL/SQL

As we have seen in the previous section, bad things happen to good programs. However, you also have to deal with bad or problematic data as well. To deal with problems during the processing of data, PL/SQL provides us with the robust ability to handle these types of errors. We call this type of program code *exception handling*.

To raise an error from within a PL/SQL program, use the built-in function named **raise_application_error**. The function requires two arguments. One is for the error number. This number must be between 20000 and 20999. The second argument is the error that you want the user to see.

As with all exception handling, this program code is placed into the **EXCEPTION** section of your PL/SQL program. Thus, our program structure will now be

In our example, we have now used the pseudo-columns, sqlcode and sqlerrm. You should always consider using these variables in your program code to ensure all your PL/SQL program completes in a manageable manner and provides the necessary feedback to diagnose potential problems and errors.

Progress Check 

1. What facility do you use to get output from within a PL/SQL program?

2. What is wrong with the following cursor declaration?

So, as you might have guessed, we have six employees in our company. It may be small, but it's very good. However, the important thing to know is that you may use variables in your loops. The other line you may have noticed was the SELECT statement contained in the PL/SQL block.

Project 6-2 Using Conditions and Loops in PL/SQL

In this project, we will create a PL/SQL program that will read the data in the products table and then print out the products that have a price above $50.

Step by Step

1. Log into SQL*Plus.

2. At the SQL prompt, enter the serveroutput command: **set serveroutput on;**.

3. Enter the following PL/SQL program:

CRITICAL SKILL 6.8
Create and Use Functions

We may also create stored objects that can be used within a **select** command. Oracle provides us with functions. There are functions to trim spaces from a field, or replace one character with another. All of these provide us with an ability to extend the capabilities of Oracle itself.

Functions are very much like stored procedures. The main difference is that functions may be used within a select statement in the column list or may also be used in the where clause.

When creating a function, you perform a **create or replace function** command. You can have variables input to the function, as well as return a value to the calling statement. A function must return a value. The data type of the returned value must be defined when creating the function. This is how a function differs from a procedure. The function will then perform its task during regular processing, allowing you to utilize the results along with your regular data, thus extending the value of your data and your database.

Project 6-3 Creating and Using a Function

The following project will walk you through the process of defining a function. Then we use the function in two select statements. The first will use the function in the returned columns and the next will use it as a data constraint. The function that we will create will perform the simple task of adding a 15-percent tax to the list price to give the price with taxes included.

Step by Step

1. Log into SQL*Plus.

2. At the SQL prompt, type in the following command:

CRITICAL SKILL 6.9
Call PL/SQL Programs

Up to this point in the chapter, we have done the following:

- Defined a PL/SQL block
- Defined a PL/SQL program
- Created a stored program
- Debugged our code

This is fine if we simply want to write every program and have it run as a simple standalone program. However, as with many programming languages it is important to write a number of separate programs that perform specific tasks, rather than a single program that performs all of your tasks. So, when writing PL/SQL programs, you should think the same way. Have programs that perform table maintenance, that perform complex logic, or simply read data from tables or files. This leads us to how to call programs from other programs, a process similar to calling subroutines within C programs. Procedures may be called from programs ranging from Oracle Forms to perl scripts, but for this section we will simply show you how to call procedures from each other.

To call a procedure from another, we can use our previous procedure, named *print_products*. Let's create another procedure that calls this procedure.

Page 241

CHAPTER 7
Java

CRITICAL SKILLS

7.1 What Does Java Mean to an Oracle DBA?

7.2 Overview of Java

7.3 Configure Java for Oracle

7.4 Java in Oracle

7.5 JDBC Drivers

7.6 Use JDBC

7.7 Use SQLJ

7.8 Java-Stored Procedures

7.9 Create Java Objects in Oracle

7.10 Understand Oracle Java Products

This chapter introduces the key features of Java that an Oracle DBA needs to understand when managing Oracle Database 10*g*. Topics include Java features and terminology, configuring Java in the database, connecting to Oracle using Java, JDBC, and SQLJ, Java-stored procedures, and the direction of Java in Oracle. This chapter may give some traditional Oracle DBAs a headache but the information in this chapter describes a popular application environment that Oracle Database 10*g* will run in, and which some DBAs need to be able to support.

Java Server Fundamentals

Mention Java to an Oracle DBA and you will often see a look of curiosity, disdain, disinterest, or fear. It has been a subject that most production DBAs have ignored for years. Although a core language since Oracle 8.1.5, most production DBAs have not placed a lot of emphasis on learning Java. A primary reason is that Java has not impacted most Oracle DBAs' job responsibilities in older releases of Oracle. However, changes in the industry, markets and Oracle technology directions no longer provide DBAs with that luxury. Java technology is integrated tightly with the Oracle Database 10*g*, Oracle Application Server 10*g*, Oracle JDeveloper 10*g*, Oracle Application Server Web Services, Oracle XML DB, and the Oracle Developer Suite 10*g*, including Oracle Forms and Reports.

NOTE
Many of the topics discussed in this chapter could, each on their own, take an entire book to cover completely. Since this is an introductory book, specifics for some topics have been omitted. Real-world experiences and additional reading will build on this material.

CRITICAL SKILL 7.1
What Does Java Mean to an Oracle DBA?

Java is one of the core languages (SQL, PL/SQL, Java, XML, and HTML) that are tightly integrated in the Oracle Database 10*g* suite of products. The Extensible Markup Language (XML) is a hierarchical data structure that describes data by embedding the metadata with the data structure. Java has become the primary language for new applications that are written for Oracle databases. Why does an Oracle DBA care about Java? Java in an Oracle environment impacts architectural decisions, performance, leveraging new technologies, security, and application development. Java-stored procedures can run in the Oracle database server. So, if Java code can run

- Oracle Database 10*g* supports technologies such as Java, XML, HTTP, and Web Services directly in the database server.

- Successful and marketable DBAs in Oracle Database 10*g* environments need to be able to do more than just performance tuning, backup/recovery, and storage management.

These technologies provide open standard solutions for integration and communication across heterogeneous, distributed environments. What's more important is that the industry is supporting these new standards and leveraging them. These standards, however, are still evolving and changing. Nevertheless, the strong advantages of these technologies cannot be ignored and large organizations are seriously looking at them.

Traditionally, Web Services has been deployed in the middle-tier. The middle-tier is the level between the client-tier (presentation-tier) and the information-tier (data-tier or database server-tier) that contains the business logic. The middle-tier usually contains an application or web server. In Oracle Database 10*g*, the database server can be a Web Services provider. Supporting Web Services in the database offers Enterprise Information Integration. With the proliferation of web-based applications, the integration and connectivity is critical. Web Services can provide the glue that leverages different technologies and environments. In disconnected environments (Internet), accessing stored procedures (Java), data, and metadata (XML) through Web Services offers additional options and flexibility.

So what does Java mean to an Oracle DBA? Java is a core language that is integrated in the Oracle Database 10*g* server and primary Oracle products such as the application server, Oracle development products, and applications. Second, Java is a key technology that is integral to technologies that are playing larger roles in database applications such as Web Services, application servers, XML, and the development of standards-based environments.

CRITICAL SKILL 7.2
Overview of Java

This section will introduce DBAs to the Java programming language from a DBA perspective. After reading this section, DBAs will also understand enough terms and acronyms to speak with confidence at any barbeque party where Java developers are in attendance. The emphasis in this section will be on terminology and concepts important for a new Oracle Database 10*g* DBA. Here are some of the reasons why Java has become a mainstream language:

- **Object-Oriented** Java is an object-oriented programming (OOP) language that has become quite mainstream. An object-oriented language uses

CRITICAL SKILL 7.3
Configure Java for Oracle

The Oracle JVM must be configured and installed to run Java in Oracle Database 10*g*. The core Java classes, including the JDBC (Java Database Connectivity APIs) and SQLJ (embedded SQL for Java) classes are all natively compiled for performance. Once the Oracle JVM is installed, Java-stored procedures can be executed in the Oracle database, but they must be configured separately on the middle and client tiers. During the database installation, a large number of Java classes are loaded into Oracle Database 10*g*.

CRITICAL SKILL 7.4
Java in Oracle

The Java runtime environment contains the JVM, the Java runtime class libraries, JDBC, SQLJ, and a Java application launcher. Oracle Database 10*g* is a deployment environment, not a development environment. Java code should be created in a Java development tool such as Oracle JDeveloper 10*g*. The appropriate Java files should then be loaded into the Oracle database.

Once the Oracle JVM option is configured, Java-stored procedures can be executed using JDBC and SQLJ. The Oracle JVM also uses a PL/SQL package DBMS_JAVA. Be aware that Java names can exceed the SQL identifier length. The database allows a Java name to be up to 4000 characters in length. If a Java name (long name) exceeds the maximum SQL identifier name length, then Oracle will also create an alias (short name). The following query will display long and short names. The long names are abbreviated for display purposes.

Format the query, and display long and short names:

CRITICAL SKILL 7.5
JDBC Drivers

Java DataBase Connectivity (JDBC) is a set of APIs that provide SQL capability for Java. JDBC is an industry-standard set of APIs for database access. There is one industry standard of APIs for JDBC. Each database vendor provides additional vendor-specific JDBC drivers that support database specific functionality. Oracle supports three different types of JDBC drivers.

JDBC Thin Driver

A JDBC thin driver is a driver written completely in Java. It is referred to as a thin driver because it does not require additional vendor-specific networking code. A JDBC thin driver is dynamically loaded at runtime. A thin driver, which uses TCP/IP, works well with standalone applications and firewalls within an intranet. Oracle networking software is not required on the client with a thin driver.

JDBC Thick Driver

A JDBC thick (OCI) driver requires additional vendor networking software. Oracle Net Services must be installed on the client or middle tier to use the JDBC thick driver. Oracle Net Services software uses the JDBC Oracle Call Interface (OCI) to access the Oracle database. Do not use the thick driver with applets. Java code using the JDBC thick driver can leverage the feature/functionality of Oracle Net Services such as connection pooling. The thick driver requires additional resources but performs better than the thin driver.

JDBC Server-Side Driver

A Java program that runs in the Oracle database server uses a server-side (KPRB) driver. This allows these server-side Java programs to access Oracle data directly without having to go through the network. With these programs running in the Oracle kernel space, this can offer a significant performance gain for data-intensive algorithms.

Use the Proper JDBC Driver

The type of JDBC driver to use is defined during database connection. Java applications that are running on a client can use a thin or thick driver. Thick drivers are used on the middle tier for application servers. Server-side drivers can only be used inside of the Oracle database. When a Java program tries to connect to the Oracle server, the JDBC driver needs to be specified. The following examples use a thin and OCI driver, respectively:

Java-stored procedures run inside of the Oracle kernel memory. To run a Java-stored procedure, a session must already be established. The connection inside a Java-stored procedure specifies to use the existing session similar to how a PL/SQL program runs inside of a current session. The following example tells the Java program to use the current session. A connection cannot be closed inside of a Java-stored procedure.

4. Checking for end of data.

5. Closing the cursor.

When writing Java database programs, a similar set of steps need to be performed:

1. Registering a driver.

2. Connecting to a database.

3. Executing a statement.

4. Fetching data. (The check for end of data is done during the fetch.)

5. Closing the resources.

Project 7-1 Accessing the Database with Java

JDBC is required for accessing databases from within Java. In this project, you will walk through the necessary JDBC steps for performing a simple query.

Step by Step

1. The first step in connecting to a database requires registering a driver. Oracle offers two different ways of registering a driver:

statement variable and use your previous connection. Then we will send the results of our SQL query into a result set variable, ready for use.

translator (precompiler) is used to convert a .sqlj file containing embedded SQL statements into a .java file containing JDBC statements. The translated .java file is then compiled. SQLJ statements can contain queries, DML, transaction control, and DDL statements. The SQLJ translator in the database will automatically convert .sqlj files into compiled Java code.

Sample SQLJ Code

SQLJ makes it easier for traditional Oracle developers to write Java database code. Oracle developers can embed the SQL statements without having to work with all of the JDBC interfaces. SQLJ performs compile time checking of SQL statements while JDBC performs runtime checking of SQL statements. The following code snippets are some examples of using SQLJ.

Privileges

To load Java classes, the **create procedure** and **create table** minimum privileges are required. Java classes are schema objects just like PL/SQL procedures or tables. Java files are stored in Java ARchive (JAR) files. A JAR file is a specialized type of Zip file. The following related Java files can be found in a JAR file: .java, .class, .properties, .sqlj, or .ser files.

Java classes can be loaded or created individually, or the loadjava tool can be used. The loadjava utility is similar to the SQL*Loader tool. loadjava loads Java files while SQL*Loader loads data. By default, Java-stored procedures run under invoker's rights.

Resolver Specifications

Programs often need to access additional programs. Typically, a search path is defined where the programs should look for additional files. Operating systems organize files in directories, while Oracle organizes database objects in schema. A Java class, on the other hand, is loaded into a schema. If a Java class needs additional classes, a search path needs to be defined. To do this, a resolver specification (spec) can be used, which defines where additional classes can be found. In other words, a resolver spec is a search path in Oracle of schemas to find Java classes.

The default resolver will automatically look in the current schema and then in PUBLIC, which is where the core Java class libraries are located. Resolvers can also be defined to specify additional schemas.

Project 7-2 Creating a Java-Stored Procedure

This project will take a DBA through the basic steps of creating a Java-stored procedure. Java programs can have a main () method. This is an entry point for a Java application. Java-stored procedures will be initiated from another application so they do not need a main () method.

Step by Step

1. Create a simple class named **MyFirstProgram.java:**

Project Summary

This project walked you through the steps a DBA will take to create and test a Java-stored procedure.

CRITICAL SKILL 7.9
Create Java Objects in Oracle

loadjava is one way of creating Java classes in the database. It is a preferred method since it can leverage Java development environments and then load the generated Java class and JAR files. The DDL statements described in the following sections can also be used to create Java classes or related files.

create java class

The **create java class** command can be used to load a Java class file from the operating system. An Oracle directory object must be created so the class file can be found on the operating system. The following statement can be used to load an individual Java class into Oracle:

CHAPTER 8
XML

CRITICAL SKILLS

8.1 Understand XML

8.2 Oracle XML DB: Use XML in the Database

8.3 SQLX: Create XML from Data Stored in Oracle

8.4 Store XML in Oracle XML DB

8.5 Use Simple Queries

8.6 Create a Relational View from XML

8.7 Learn Programmatic Access Using XSLT

Before we begin to see how Extensible Markup Language (XML) and Oracle Database 10*g* work together, we need to look at why XML is important for us and why Oracle has included significant support for XML in the Oracle XML DB. This chapter will discuss what XML is and why it is important to our computer futures. Additionally, we will show you how Oracle can become an integral part of all of your solutions that need to use XML.

CRITICAL SKILL 8.1
Understand XML

Most of you have visited countless web sites, have navigated to the menu bar at the top of a browser screen, and then selected View Source. At this point, guess what you are looking at html or *hypertext markup language*. As html became more and more popular, the most brilliant minds decided to extend the power of the language. They were specifically interested in designing a way to enhance HTML's functionality with a set of user-defined tags. A *tag* is simply a keyword bound by greater-than signs, less-than signs, and a specially placed forward slash. The following listing shows a few sample tags in html; the tags themselves are bolded.

NOTE
As you may have noticed, we have glossed over the discussion of XML it is a very big topic demanding its own phone book sized volume. We will attend to Oracle XML DB specifics in the rest of this chapter, and leave you to discover the big picture of XML on your own.

CRITICAL SKILL 8.2
Oracle XML DB: Use XML in the Database

XML is quickly being adopted as the data transport mechanism on the information highway it is a W3C-endorsed standard markup language for documents. *W3C* stands for World Wide Web Consortium, an organization of approximately 400 members. They represent both public and private sectors, whose goal is to develop interoperable technologies (specifications, guidelines, software, and tools) for the World Wide Web. The following is an example XML document, leveraging the power of custom tags:

CRITICAL SKILL 8.3
SQLX: Create XML from Data Stored in Oracle

For years, organizations have been pouring their data into relational databases. If you're trading data with another organization, however, it's likely that you'll need to pull data out of your relational database and format that data as XML before transmitting it.

Let's first point out that generating XML from a relational source is a not a trivial task. It involves understanding how to map relational data into a hierarchal structure. On the other hand, creating the XML once you understand the mapping you want has been made easy.

The SQL/XML Standard

Oracle Database 10*g* implements a number of standards-based functions enabling you to query relational data and return XML documents. These functions collectively fall under the heading of SQL/XML, sometimes referred to as SQLX. SQL/XML is part of the ANSI/ISO SQL standard.

The international standard for SQL/XML defines the following elements:

- **XML** A data type to hold XML data
- **XMLAgg** A function to group, or aggregate, XML data in **group by** queries
- **XMLAttributes** A function used to place attributes in XML elements returned by SQL queries
- **XMLConcat** A function to concatenate two or more XML values
- **XMLElement** A function to transform a relational value into an XML element, in the form: *elementName value / elementName*
- **XMLForest** A function to generate a list, called a "forest," of XML elements from a list of relational values
- **XMLNamespaces** A function to declare namespaces in an XML element
- **XMLSerialize** A function to serialize an XML value as a character string

Oracle Database 10*g* has implemented the following XML data type (as XMLType), XMLAgg, XMLConcat,

XMLElement, and XMLForest. Support for the other functions is planned in future releases. In addition to the functions and the data type, the SQL/XML standard defines rules for transforming column names into XML element names and for transforming SQL data types into XML data types. These rules are applied automatically by XMLElement and the other SQL/XML functions. Let's look at four of many SQLX functions that are commonly used to produce XML from the relational database.

Previous Next

The following is the expected result (partial) of our SQL query:

3. When can the **xmlattributes()** function be called?

4. What corresponding SQL clause must accompany an **xmlagg()** function?

CRITICAL SKILL 8.4
Store XML in Oracle XML DB

Databases are relational and XML is hierarchical, so prior to the introduction of Oracle XML DB there had been no simple, elegant way to integrate the two. Traditionally, developers have had two choices: either use a parser to deconstruct the document data into relational data and store it as such in the database, or store the entire document as a text file, preserving its text-based structure. Oracle XML DB starts it all off by first solving the problem of representing an XML document in its native format the XMLType.

The Native XMLType

The XMLType was created to be able to preserve the XML paradigm while getting the benefits of relational performance and scalability. It is a native server data type that allows the database to understand that a column or table contains XML in the same way that the DATE data type allows the database to understand that a column contains a date. The twist is that the XMLType also provides methods that allow common operations such as schema validation and XSL transformations to be performed on XML content.

The XMLType data type can be used just like any other data type, such as when creating a column in a relational table, when declaring PL/SQL variables, and when defining and calling PL/SQL procedures and functions. Since XMLType is an object type, it is also possible to create a table of XMLType. The following listing shows how to create a simple table with an XMLType column:

Project 8-3 Using Simple Queries

In this project, we will be querying for a specific XML document and then updating it.

Step by Step

1. Log into SQL*Plus.

2. At the SQL prompt, enter the long command, **set long 10000**, and then press ENTER.

3. At the SQL prompt, enter the long command, **set pagesize 80**, and then press ENTER.

4. Enter the following SQL query:

Project Summary

This project illustrated the use of **existsnode(), extract()**, and **extractvalue()** when identifying an XML document stored in the Oracle Database 10*g*.

In the future, Oracle is looking at adding several new capabilities to this list, including additional database and XML functionalities, such as XQuery, which will be a language specifically designed to query XML data from a document perspective rather than the rows-and-tables perspective of SQL. We have had fun finding and returning XML documents. Let's proceed and look at how we can take XML documents and represent them relationally.

Progress Check

1. Is the Oracle XML DB repository separate from the Oracle Database 10*g*?

2. Why must we "register" the URL of XML Schemas into the Oracle XML DB repository?

3. How do we load an XML document into the Oracle XML DB repository?

4. What is the difference between the **existsnode()** function and the **extract()** function?

CRITICAL SKILL 8.6
Create a Relational View from XML

Oracle XML DB makes it possible to expose XML content, stored in the database through conventional relational views. This means that tools, applications, and programmers who have no understanding of XML, but understand the Oracle Database 10*g*, can now work with XML content. To accomplish this, the view

Progress Check Answers

1. No, the Oracle XML DB is integrated with Oracle Database 10*g*.

2. The XML Schema defines the legal structure for an XML document. To confirm that the XML stored within the XMLType is "valid," the URL for the XML Schema it is associated with must be registered.

3. XML documents can be loaded into the Oracle XML DB via programs using SQL, PL/SQL, and Java or by going through the protocols FTP, HTTP, and WebDAV.

4. The **existsnode()** function identifies the existence of a node returning a Boolean value, while the **extract()** function extracts the node or set of nodes identified, returning a single text node.

Previous Next

CRITICAL SKILL 8.7
Learn Programmatic Access Using XSLT

Data encapsulated in XML can be used in a number of ways. One common means of manipulating it is through the use of Extensible Stylesheet Language Transformations (XSLT), which enable developers to define operations that must be performed on an XML document to produce a specific result. This ability to transform information on the fly makes it possible to use a single source for multiple outputs such as HTML, whether those outputs lead to different databases or to different browsers. XML documents have structure but no format.

Oracle XML DB uses the template rules and other formatting elements that appear within the XSLT style sheets. It includes an XSLT-based transformation engine to automatically transform XML-tagged documents into multiple display formats, store the transformed renditions generated by the XSLT style sheets, and deliver content in the appropriate formats to various devices.

The following example shows how **transform()** can apply XSLT to an XSL style sheet, PurchaseOrder.xsl, to transform the PurchaseOrder.xml document:

CHAPTER 9
Large Database Features

CRITICAL SKILLS

In this chapter, we will be covering topics and features available in Oracle Database 10*g* with which you will need to be familiar when working with large databases. These features are among the more advanced that you will encounter, but they're necessary, as databases are growing larger and larger. When you start working with Oracle, you will find yourself facing the trials and tribulations associated with large databases sooner rather than later. The quicker you understand the features and know where and when to use them, the more effective you will be.

CRITICAL SKILL 9.1
What Is a Large Database?

Let's start by describing what we mean by a large database. "Large" is a relative term that changes over time. What was large five or ten years ago is small by today's standards, and what is large today will be peanuts a few years from now. Each release of Oracle has included new features and enhancements to address the need to store more and more data. For example, Oracle8*i* was released in 1999 and could handle databases with *terabytes* (1024 gigabytes) of data. In 2001, Oracle9*i* was released and could deal with up to 500 *petabytes* (1024 terabytes). Oracle Database 10*g* now offers support for *exabyte* (1024 petabytes) databases. You won't come across too many databases with exabytes of data right now, but in the future at least we know Oracle will support them.

The most obvious examples of large database implementations are data warehouses and decision support systems. These environments usually have tables with millions or billions of rows, or wide tables with large numbers of columns and many rows. There are also many OLTP systems that are very large and can benefit from the features we are about to cover. Since we've got many topics to get through, let's jump right in and start with data partitioning.

NOTE
Many of the topics discussed in this chapter could, each on their own, take an entire book to cover completely. Since this is an introductory book, specifics for some topics have been omitted. Real-world experiences and additional reading will build on this material.

CRITICAL SKILL 9.2
Why and How to Use Data Partitioning

As our user communities require more and more detailed information in order to remain competitive, it has fallen to us as database designers and administrators to help ensure that the information is managed efficiently and can be retrieved for analysis effectively. In this section, we will discuss partitioning data, and why it is so important when working with large databases. Afterward, we'll follow the steps required to make it all work.

Why Use Data Partitioning

Let's start by defining what we mean by *data partitioning*. In its simplest form, it is a way of breaking up or subsetting data into smaller units that can be managed and accessed separately. It has been around for a long time both as a design technique and as a technology. Let's look at some of the issues that gave rise to the need for partitioning and the solutions to these issues.

Tables containing very large numbers of rows have always posed problems and challenges for DBAs, application developers, and end-users alike. For the DBA, the problems centered on the maintenance and manageability of the underlying data files that contain the data for these tables. For the application developers and end users, the issues were query performance and data availability.

To mitigate these issues, the standard database design technique was to create physically separate tables, identical in structure (for example, columns), but with each containing a subset of the total data (we will refer to this design technique as *nonpartitioned*). These tables could be referred to directly or through a series of views. This technique solved some of the problems, but still meant maintenance for the DBA to create new tables and/or views as new subsets of data were acquired. In addition, if access to the entire dataset was required, a view was needed to join all subsets together.

Figure 9-1 illustrates this design technique. In this sample, separate tables with identical structures have been created to hold monthly sales information for 2005. Views have also been defined to group the monthly information into quarters using a **union** query. The quarterly views themselves are then grouped together into a view that represents the entire year. The same structures would be created for each year of data. In order to obtain data for a particular month or quarter, an end user would have to know which table or view to use.

Similar to the technique illustrated in Figure 9-1, the partitioning technology offered by Oracle Database 10*g* is a method of breaking up large amounts of data into smaller, more manageable chunks. But, unlike the nonpartitioned technique, it is transparent to

Some other points on global partitioned indexes:

- They require more maintenance than local indexes, especially when you drop data partitions.
- They can be unique.
- They cannot be bitmap indexes.
- They are best suited for OLTP systems for direct access to specific records.

Prefixed and Nonprefixed Partition Indexes In your travels through the world of partitioning, you will hear the terms *prefixed* and *nonprefixed* partition indexes. These terms apply to both local and global indexes. An index is prefixed when the leftmost column of the index key is the same as the leftmost column of the index partition key. If the columns are not the same, the index is nonprefixed. That's all well and good, but what affect does it have?

It is a matter of performance nonprefixed indexes cost more, from a query perspective, than prefixed indexes. When a query is submitted against a partitioned table and the predicate(s) of the query include the index keys of a prefixed index, then pruning of the index partition can occur. If the same index was nonprefixed instead, then all index partitions may need to be scanned. (Scanning of all index partitions will depend on the predicate in the query and the type of index, global or local if the data partition key is included as a predicate and the index is local, then the index partitions to be scanned will be based on pruned data partitions.)

Project 9-1 Creating a Range-Partitioned Table and a Local Partitioned Index

Data and index partitioning are an important part in maintaining large databases. We have discussed the reasons for partitioning and shown the steps to implement it. In this project, you will create a range-partitioned table and a related local partitioned index.

Step by Step

1. Create two tablespaces called **inv_ts_2007q1** and **inv_2007q2** using the following SQL statements. These will be used to store data partitions.

Progress Check

1. List at least three DML commands that can be applied to partitions as well as tables.

2. What does partition pruning mean?

3. How many table attributes can be used to define the partition key in list partitioning?

4. Which type of partitioning is most commonly used with a date-based partition key?

5. Which partitioning types cannot be combined together for composite partitioning?

6. How many partition keys can be defined for a partitioned table?

7. Which type of partitioned index has a one-to-one relationship between the data and index partitions?

8. What is meant by a prefixed partitioned index?

CRITICAL SKILL 9.3
Compress Your Data

As you load more and more data into your database, performance and storage maintenance can quickly become concerns. Usually at the start of an implementation of a database, data volumes are estimated and projected a year or two ahead. However, often times these estimates turn out to be on the low side and you find yourself

Progress Check Answers

1. The following DML commands can be applied to partitions as well as tables: **delete, insert, select, truncate**, and **update**.

2. Partition pruning is the process of eliminating data not belonging to the subset defined by the criteria of a query.

3. Only one table attribute can be used to define the partition key in list partitioning.

4. Range partitioning is most commonly used with a date-based partition key.

5. List and hash partitioning cannot be combined for composite partitioning.

6. Only one partition key may be defined.

7. Local partitioned indexes have a one-to-one relationship between the data and index partitions.

8. A partitioned index is prefixed when the leftmost column of the index key is the same as the leftmost column of the index partition key.

Previous Next

CRITICAL SKILL 9.4
Use Parallel Processing to Improve Performance

Improving performance, and by this we usually mean query performance, is always a hot item with database administrators and users. One of the best and easiest ways to boost performance is to take advantage of the parallel processing option offered by Oracle Database 10*g* (Enterprise Edition only).

Using normal (that is, serial) processing, the data involved in a single request (for example, user query) is handled by one database process. Using *parallel processing*, the request is broken down into multiple units to be worked on by multiple database processes. Each process looks at only a portion of the total data for the request. Serial and parallel processing are illustrated in Figures 9-5 and 9-6, respectively.

Parallel processing can help improve performance in situations where large amounts of data need to be examined or processed, such as scanning large tables, joining large tables, creating large indexes and scanning partitioned indexes. In order to realize the benefits of parallel processing, your database environment should not already be running at, or near, capacity. Parallel processing requires more processing, memory, and I/O resources than serial processing. Before implementing parallel processing, you may need to add hardware resources. Let's forge ahead by looking at the Oracle Database 10*g* components involved in parallel processing.

Parallel Processing Database Components

Oracle Database 10*g*'s parallel processing components are the *parallel execution coordinator* and the *parallel execution servers*. The parallel execution coordinator is responsible for breaking down the request into as many processes as specified by the request. Each process is passed to a parallel execution server for execution during which only a portion of the total data is worked on. The coordinator then assembles the results from each server and presents the complete results to the requester.

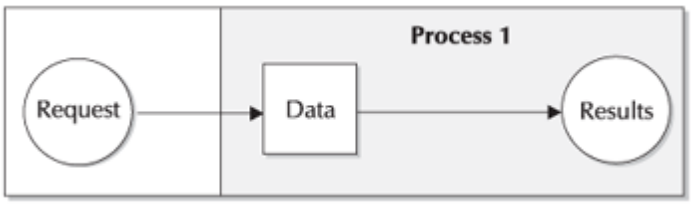


FIGURE 9-5. *Serial processing*

- Parallel processing will be disabled for DML commands (for example, **insert, update, delete**, and **merge**) on tables with triggers or referential integrity constraints.

- If a table has a bitmap index, DML commands are always executed using serial processing if the table is nonpartitioned. If the table is partitioned, parallel processing will occur, but Oracle will limit the degree of parallelism to the number of partitions affected by the command.

Parallel processing can have a significant positive impact on performance. Impacts on performance are even greater when you combine range or hash-based partitioning with parallel processing. With this configuration, each parallel process can act on a particular partition. For example, if you had a table partitioned by month, the parallel execution coordinator could divide the work up according to those partitions. This way, partitioning and parallelism work together to provide results even faster.

CRITICAL SKILL 9.5
Use Materialized Views

So far, we have discussed several features and techniques at our disposal to improve performance in large databases. In this section, we will discuss another feature of Oracle Database 10*g* that we can include in our arsenal: materialized views.

Originally called snapshots, *materialized views* were introduced in Oracle8 and are only available in the Enterprise Edition. Like a regular view, the data in a materialized view are the results of a query. However, the results of a regular view are transitory they are lost once the query is complete and if needed again, the query must be reexecuted. In contrast, the results from a materialized view are kept and physically stored in a database object that resembles a table. This feature means that the underlying query only needs to be executed once and then the results are available to all who need them.

From a database perspective, materialized views are treated like tables:

- You can perform most DML and query commands such as insert, delete, update and select.

- They can be partitioned.

- They can be compressed.

Progress Check 

1. True or False: Tables with many foreign keys are good candidates for compression.

2. Name the two processing components involved in Oracle Database 10*g*'s parallel processing.

3. What is the function of the SQLAccess Advisor?

4. True or False: In order to access the data in a materialized view, a user or application must query the materialized view directly?

5. List the ways in which parallel processing can be invoked.

6. In what situation can index key compression *not* be used on a unique index?

CRITICAL SKILL 9.6
Real Application Clusters: A Primer

When working with large databases, issues such as database availability, performance and scalability are very important. In today's 24/7 environments, it is not usually acceptable for a database to be unavailable for any length of time even for planned maintenance or for coping with unexpected failures. Here's where Oracle Database 10*g*'s Real Application Clusters (RAC) comes in.

Originally introduced in Oracle9*i* and only available with the Enterprise Edition, Real Application Clusters is a feature that allows database hardware and instances to be grouped together to act as one database using a shared-disk architecture. Following is a high-level discussion on RAC's architecture.

Progress Check Answers

1. True.

2. The Parallel Execution Coordinator and the Parallel Execution Servers.

3. The SQLAccess Advisor recommends potential materialized views based on historical or theoretical scenarios.

4. False. While the end user or application can query the materialized view directly, usually the target of a query is the detail data and Oracle's query rewrite capabilities will automatically return the results from the materialized view instead of the detail table (assuming the materialized view meets the query criteria).

5. Parallel processing can be invoked based on the parallelism specified for a table at the time of its creation, or by providing the parallel hint in a select query.

6. If the unique index has only one attribute, key compression cannot be used.

Previous Next

- The Global Cache Service controls data exchange between nodes using the *Cache Fusion* technology. Cache Fusion synchronizes the memory cache in each node using high-speed communications. This allows any node to access any data in the database.

- Shared storage consists of data and index files, as well as control files.

This architecture makes RAC systems highly available. For example, if Node 2 in Figure 9-9 fails or requires maintenance, the remaining nodes will keep the database available.

This activity is transparent to the user or application and as long as at least one node is active, all data is available. RAC architecture also allows near-linear scalability and offers increased performance benefits. New nodes can easily be added to the cluster when needed to boost performance.

Administering and maintaining data files on both RAC and single-node systems have always required a good deal of effort, especially when data partitioning is involved. Oracle's solution to reducing this effort is Automatic Storage Management, which we will discuss next.

CRITICAL SKILL 9.7
Automatic Storage Management: Another Primer

In previous versions of Oracle and with most other databases, management of data files for large databases consumes a good portion of the DBA's time and effort. The number of data files in large databases can easily be in the hundreds or even thousands. The DBA must coordinate and provide names for these files and then optimize the storage location of files on the disks. The new *Automatic Storage Management*(*ASM*) feature in Oracle Database 10*g* Enterprise Edition addresses these issues.

ASM simplifies the management of disks and data files by creating logical groupings of disks into *disk groups*. The DBA need only refer to the groups, not the underlying data files. Data files are automatically named and distributed evenly (striped) throughout the disks in the group for optimal throughput. As disks are added or removed from the disk group, ASM redistributes the files among the available disks, automatically, while the database is still running. ASM can also mirror data for redundancy.

ASM Architecture

When ASM is implemented, each node in the database (clustered or nonclustered) has an ASM instance and a database instance, with a communication link between

Ask the Expert

Q: After ASM disk groups are defined, how are they associated with a table?

A: ASM disk groups are referred to during tablespace creation, as in the following example:

```
1 create tablespace ts1
2   datafile  +diskgrp1 /alias1;
```

This listing creates tablespace **ts1** in disk group **diskgrp1**. Note that this assumes that both **diskgrp1** and **alias1** have previously been defined.

A table can now be created in tablespace **ts1** and it will use ASM data files.

While ASM can be implemented in a single-node environment, its real power and benefits are realized when used in RAC environments. This powerful combination is the heart of Oracle Database 10*g*'s grid computing database architecture.

CRITICAL SKILL 9.8
Grid Computing: The *"g"* in Oracle Database 10*g*

In this chapter, we have discussed many issues and demands surrounding large databases performance, maintenance efforts, and so on. We have also discussed the solutions offered by Oracle Database 10*g*. Now we will have a high-level look at Oracle Database 10*g*'s grid-computing capabilities.

NOTE
Oracle Database 10g's grid computing applies to both database and application layers. We will just be scratching the surface of grid computing and focusing on the database components.

The theory behind grid computing is that all parts of the grid (databases, applications, servers, disks, and so forth) work together in a highly integrated fashion, with each component being able to react appropriately to changes in other components in the grid. This results in efficient use of resources, faster response times, high availability, and so on.

Progress Check

1. In an RAC system, what component connects the nodes to the shared storage?

2. When a disk is added to an ASM disk group, what happens to the existing data in the group?

3. True or False: A database instance in an ASM system accesses the disk groups?

4. What component of a self-managing database contains workload and performance statistics used for self-management activities?

5. What are the database-related components that are part of grid computing?

6. What is the function of the Cluster Manager in RAC systems?

CRITICAL SKILL 9.9
Use SQL Aggregate and Analysis Functions

Once your database has been loaded with data, your users or applications will, of course, want to use that data to run queries, perform analysis, produce reports, extract data, and so forth. Oracle Database 10*g* provides many sophisticated aggregation and analysis functions that can help ease the pain sometimes associated with analyzing data in large databases.

Progress Check Answers

1. The Global Cache Service (or Cache Fusion) connects the nodes to the shared storage.

2. The existing data is automatically redistributed among all disks in the disk group.

3. False. The database instance communicates with the ASM instance to determine which ASM files to access directly. Only the ASM instance works with the disk groups.

4. The Automatic Workload Repository contains workload and performance statistics used for self-management activities.

5. RAC, ASM, and OEM are the database components that are part of grid computing.

6. The Cluster Manager monitors the status of each database instance in the cluster and enables communication between the instances.

As we stated earlier, we as database administrators do not have to know the theory behind the functions provided by Oracle Database 10*g*, or even how to use their results. However, we should be able to let our users know what capabilities are available. Knowing this, our users will be able to take advantage of these functions and construct efficient queries. In the next section, we will be discussing a new feature in Oracle Database 10*g* SQL models.

CRITICAL SKILL 9.10
Create SQL Models

One of the more powerful data analysis features introduced in Oracle Database 10*g* is *SQL models*. SQL models allow a user to create multidimensional arrays from query results. Formulas, both simple and complex, can then be applied to the arrays to generate results in which the user is interested. SQL models allow inter-row calculations to be applied without doing expensive self-joins.

SQL models are similar to other multidimensional structures used in business intelligence applications. However, because they are part of the database, they can take advantage of Oracle Database 10*g*'s built-in features of scalability, manageability security, and so on. In addition, using SQL models, there is no need to transfer large amounts of data to external business intelligence applications.

A SQL model is defined by the **model** extension of the **select** statement. Columns of a query result are classified into one of three groups:

- **Partitioning** This is the same as the analytic partitioning we defined in the Windowing Functions section.
- **Dimensions** These are the attributes used to describe or fully qualify a measure within a partition. Examples could include product, sales rep id, and phone call type.
- **Measures** These are the numeric (usually) values to which calculations are applied. Examples could include quantity sold, commission amount, and call duration.

One of the main applications of SQL models is projecting or forecasting measures based on existing measures. Let's look at an example of the **model** clause to illustrate. The listing and its results show an aggregate query using the SALES table:

The **model** clause has many variations and allows for very powerful calculations. Let's point out some of the characteristics and/or features you should be aware of. Supported functionalities include the following:

- Looping (for example, FOR loops)
- Recursive calculations
- Regression calculations
- Nested cell references
- Dimension wildcards and ranges
- The model clause does not update any base table, although in theory you could create a table or materialized view from the results of the query using the **model** clause.

Restrictions include the following:

- The **rules** clause cannot include any analytic SQL or windowing functions.
- A maximum of 20,000 rules may be specified. This may seem like plenty, but a FOR loop is expanded into many single-cell rules at execution time.

Project 9-2 Using Analytic SQL Functions and Models

Once all the database structures have been put in place and data has been loaded, the users will want to analyze it. Knowing what functions are available is important, and so is their use as well, at least to some extent. So, in this project we will walk through a more complex analytical example that includes using the **lag** function and creating a SQL model.

Step by Step

1. Create a view of the SALES table using the following listing. (The SALES table should have been created during your Oracle installation process.) This view will calculate the percentage change (called **percent_chng**) of **quantity_sold** from one year to the next using the **lag** function, summarized by **prod_category, channel_desc**, and **calendar_year**.

APPENDIX
Mastery Check Answers

Team Fly

Previous Next

Chapter 1: Database Fundamentals

1. The _____ background process is primarily responsible for writing information to the Oracle Database 10*g* files.

The database writer, or dbwr, background process is primarily responsible for writing information to the Oracle Database 10*g* files.

2. How many online redo log groups are required to start an Oracle Database 10*g*?

B.

Explanation The minimum number of redo log groups required to start an Oracle Database 10*g* is two though many DBAs add more groups. This increases the fault tolerance of the database.

3. Of the following four items of information, which one is not stored in Oracle Database 10*g*'s control files?

B.

Explanation The creator of the database is not stored anywhere in the assortment of the Oracle Database 10*g* support files.

4. What is the function of a default temporary tablespace in the support of the Oracle Database 10*g*?

The default temporary tablespace is the location where sessions build intermediary objects to satisfy queries and perform sort operations.

5. Differentiate between an Oracle Database 10*g* and instance.

The database is a collection of files, whereas an instance is a handful of processes working in conjunction with a computer's memory to support access to those files.

6. Activities such as allocating space in the database and user management are commonly performed by the DBA. What feature in the Oracle Database 10*g* allows some of these secure operations to be carried out by non-DBA users? How are these rights given out?

System privileges allow the DBA to give out rights to perform selected secure operations to other Oracle Database 10*g* users. These privileges are given out with a grant statement.

Previous Next

14. There are many ways to replicate data from one node to another. What main feature does Oracle Streams provide that is missing from many other methods?

Oracle Streams can keep one node's data in sync with another, an almost insurmountable chore in many do-it-yourself solutions.

15. What does the acronym SQL stand for?

A.

Explanation SQL is the industry standard for many relational database systems. Interestingly enough as you become more familiar with its syntax, it reads very much like plain and simple English. This makes it even more attractive as a data retrieval language.

Chapter 2: SQL: The Structured Query Language

1. DDL and DML translate to _____ and _____, respectively.

DDL stands for Data Definition Language. DML stands for Data Manipulation Language.

2. Which of the following descriptions is true about insert **statements?**

B.

Explanation Insert statements can *never* have a **where** clause. Every **insert** will create a row, providing it doesn't violate any constraints.

3. In addition to the two mandatory keywords required to retrieve data from the database, there are three optional keywords. Name them.

The optional parts of a **select** statement are the **where** clause, an **order by** statement, and a **group by** statement.

4. Write a SQL statement to select the customer last name, city, state, and amount sold for the customer represented by customer ID 100895.

Possible solutions include any of the Oracle and ANSI join options. The following SQL statement joins the CUSTOMERS and SALES tables using a simple Oracle join:

9. Create a view that contains all products in the Electronics category.

The DDL to create a view with only Electronics products could be created this way:

is granted to a user. Granting select on an individual table is an example of an object privilege.

15. How would you implement your corporate password policy in Oracle?

A profile can be used to implement a password management policy as well as to limit resources for a user. You can specify the number of days after which a user must change their password. You can also establish a policy where a password cannot be used again before a specified number of days has elapsed, and/or the new password must differ from the old by a certain number of changes. A function can also be used to ensure that the user will create a complex password.

Chapter 4: Networking

1. The _____ background process registers the service information to the listener.

The PMON background process registers the service information to the listener.

2. True or False: The LOCAL_LISTENER **parameter should be set to work with port 1521.**

False.

Explanation The **LOCAL_LISTENER** parameter is defined when port 1521 is not used.

3. The _____ is used during installation to configure Oracle Net Services.

The Oracle Net Configuration Assistant is used during installation to configure Oracle Net Services.

4. The _____ file can be used to define grant or deny access to the Oracle database server.

The sqlnet.ora file can be used to define, grant, or deny access to the Oracle database server.

5. The _____ utility can also be used to test a service.

The tnsping utility can also be used to test a service.

6. A _____ contains a set of parameters that define Oracle Net options on the remote or database server.

A profile contains a set of parameters that define Oracle Net options on the remote or database server.

Previous Next

7. The ldap.ora file location can be manually specified with the _____ or TNS_ADMIN **environmental variables.**

The ldap.ora file location can be manually specified with the ***LDAP_ADMIN*** or **TNS_ADMIN** environmental variables.

8. True or False: The easy naming method is a valid naming method.

True.

Explanation The easy naming method is a valid naming method.

9. The Oracle LDAP directory is called the _____.

The Oracle LDAP directory is named the Oracle Internet Directory.

10. True or False: The Oracle Management Service is a repository of information generated by the Management Agent.

False.

Explanation The Oracle Management Service interfaces with the management agents to process and monitor information. It is not the repository.

Chapter 5: Backup and Recovery

1. What are some advantages of cold backups, and when would you use them?

Cold backups are consistent and very simple to implement. They are useful in situations where the database can be brought down to perform a backup. No recovery is required after a restore from a cold backup, if you choose. You can also perform a cold backup of a database in archivelog mode and perform a recovery of that database. These deliver simplicity and flexibility in cases where you can take a database down to back it up.

2. What are disadvantages of cold backups?

The database is not available during the backup.

3. Describe the difference between a logical and a physical backup.

A physical backup is performed by utilities such as RMAN or by a hot or cold backup, and operates on the underlying database data files. A logical backup, on the other hand, is performed by utilities such as Data Pump Export or Import and allows for a backup or restore to be performed on logical database structures such as tables or indexes.

4. Name three different types of backups.

Three different types of backups are physical hot and cold backups, logical backups as implemented by Data Pump Export/Import, and RMAN backups.

10. Are there any disadvantages to an RMAN recovery catalog?

One disadvantage to the recovery catalog is that it must be backed up itself. This is simple since it is a very small schema.

11. How can default settings be set up by you for future runs of RMAN?

Default settings can be set up by you with the RMAN **configure** command. Channels can be configured to achieve optimal performance and other backup configurations can be created to simplify backup and recovery scripts.

12. What is an RMAN backup-set and how does it relate to a backup-piece?

A backup-set is made up of one or more backup-pieces and is what constitutes a full or incremental backup. A backup-piece is a file that is managed by RMAN.

13. Describe the ways in which corrupt blocks can be recovered.

Individual data blocks can be recovered with the database online and available to users. You need to specify which block can be recovered and these can be found in the alert log or the v$backup_corruption or v$copy_corruption views. Block media recovery is only available with RMAN.

14. What are some advantages to incremental image copies?

Incremental image copies save disk space and network resources since the backup files are smaller. They can also speed up recovery when compared to applying archive logs.

15. When performing a recovery from a hot backup, do all files and tablespaces need to be brought forward to the same point in time?

All files and tablespaces (except for files in read-only tablespaces) need to be brought forward to the same point in time.

Chapter 6: PL/SQL

1. Where is PL/SQL executed?

PL/SQL is executed within the confines of the database. It will receive parameters and return data, but all program execution is performed in the database.

2. Which type of PL/SQL statement would you use to increase the price values by 15 percent for items with more than 1500 in stock and by 20 percent for items with fewer than 500 in stock?

Previous Next

B.

Explanation To be able to perform a validation of your data and then perform one task if the condition is met and another when the condition is not met, you must use the IF-THEN-ELSE construct.

3. What is the fetch **command used for?**

The **fetch** command is used to retrieve data from an open cursor that had been opened previously.

4. What will the following command perform?

It will set the value of the variable to 500.

5. What is wrong with this function definition?

The END clause needs to have the same name as the name of the procedure. This value should be raise_price, not lower_price.

6. What is the advantage of using the %TYPE attribute when defining PL/SQL variables?

The advantage of using the %TYPE attribute when defining PL/SQL variables is that it links the definition of the variable to the database definition. It also allows a program to adjust for changes to database structures while impacting the execution of the PL/SQL program.

7. What Oracle Database 10*g* facility besides PL/SQL supports exception handling based on error numbers?

No other Oracle Database 10*g* facility besides PL/SQL supports exception handling based on error numbers.

8. A commit that is issued in a PL/SQL program will commit what?

All transactions that are currently pending will be committed, unless the commit is issued within an autonomous transaction.

Chapter 7: Java

1. True or False: One of the benefits of EJBs is that they combine business and user interface logic.

False.

Explanation EJBs contain business logic, not user interface logic. User interface logic is contained in servlets.

Previous Next

10. The _____ script is used to configure the Oracle JVM.

The **initjvm.sql** script is used to configure the Oracle JVM.

Explanation The **initjvm.sql** script loads the Java classes into the Oracle database and sets up the Oracle JVM in the Oracle database.

Chapter 8: XML

1. XML stands for Extensible _____ Language.

XML stands for Extensible Markup Language.

2. XML documents are in what type of data structure?

B.

Explanation XML documents tag data using a hierarchical structure.

3. What is the notation used when querying XML documents, which is implemented with Oracle Database 10*g*?

XPath is the notation used when querying XML documents.

Explanation There are plans to base queries using the XQuery standard.

4. What is a well-formed XML document?

A well-formed XML document is syntactically correct (in other words, there are corresponding end tags for every start tag).

5. Using the existsnode() **function within a SQL query should return what value if the node exists?**

D.

Explanation existsnode() returns a Boolean value: 0 if the node is not found, 1 if it is.

6. Of the following four methods listed, which is used to store XML documents in Oracle XML DB?

D.

Explanation For XML document management, Oracle has implemented numerous methods that can access Oracle XML DB.

7. What is used to ensure that an XML document is valid?

XML Schema is associated to an XML document that defines the structure and data types the XML document must conform to.

8. What is the defined data type that is used to store XML documents in Oracle Database 10*g*?

XMLType is the defined data type used to store XML documents in Oracle Database 10*g*.

9. When using xmlagg() **function to build an XML type, what SQL clause must it be accompanied by?**

When using **xmlagg()** function to build an XML type, it must be accompanied by a **group by** clause.

Explanation The **group by** clause is used to group the result set into a hierarchical structure.

10. What does the acronym XSLT stand for?

C.

Explanation XSLT, or Extensible Stylesheet Language, is used to automatically transform XML-tagged documents into multiple display formats such as HTML, or into another XML document.

11. What does the acronym SQLX stand for?

A.

Explanation SQLX is the combination of the Structured Query Language used to access relational databases and XPath, which is used to query XML documents.

Chapter 9: Large Database Features

1. Which of the following is *not* a valid combination for composite partitioning?

C.

Explanation Range partitioning with list partitioning is *not* a valid combination for composite partitioning.

2. What data population methods can be used on a compressed table that result in the data being compressed?

For data to be compressed, it must be bulk loaded into the table or you can issue an **alter table** statement to compress existing data.

3. _____ partitioned indexes are defined independently of the data partitions, and _____ partitioned indexes have a one-to-one relationship with the data partitions.

Previous Next

Index

Symbols

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Previous Next

Q

R

S

T

U

V

W

X

Y

Z