



PART I

Oracle Database 10g Administrative I Exam

- **Chapter 1** Basic Oracle Concepts
- **Chapter 2** Installing Oracle Database 10g
- **Chapter 3** Creating an Oracle Database
- **Chapter 4** Interfacing with the Oracle Database
- **Chapter 5** Managing Oracle Processes
- **Chapter 6** Managing Oracle Storage Structures
- **Chapter 7** Administering Users
- **Chapter 8** Managing Database Objects
- **Chapter 9** Manipulating Database Data
- **Chapter 10** Programming Oracle with PL/SQL
- **Chapter 11** Securing the Database
- **Chapter 12** Configuring Oracle Networking
- **Chapter 13** Managing Shared Servers
- **Chapter 14** Managing Database Performance
- **Chapter 15** Monitoring Oracle
- **Chapter 16** Managing Undo
- **Chapter 17** Dealing with Locking
- **Chapter 18** Configuring the Database for Backup and Recovery
- **Chapter 19** Backing Up Oracle Databases
- **Chapter 20** Recovering Oracle Databases
- **Chapter 21** Managing Globalization in Oracle Databases





CHAPTER 1

Basic Oracle Concepts

In this chapter you will learn

- What a database is and what makes a database relational
- What SQL is
- Which database objects are supported in Oracle 10g
- What a database administrator does
- How the Oracle database fits into the Oracle product family

Someone once said that the best place to start is at the beginning. With Oracle, that means understanding where the idea of a relational database management system (RDBMS) came from and what a database is—in computer and everyday terms. Even though the material presented here may not be directly tested on the exam, this is assumed knowledge, however, so a quick read is probably a good idea.

Introduction to Databases and the Relational Model

In one form or another, databases have always been around, though their exact shape was not always easily recognizable. As long as some form of data had to be stored, there was always a method of storing it.

Databases, in their most simple form, are a mechanism for storing data. The data can be logical, like the values stored in a computer program, or may be physical, like a file or receipt. You probably have databases in existence all around you, but you may not see them as such. For example, the shoebox in which you've placed your tax receipts for the accountant is a database of your annual expenses. When you open a file cabinet and take out a folder, you are accessing a database. The content of the file folder is your data (e.g., your credit card statements, your bank statements, invoices, purchase orders, etc.). The file cabinet and drawers are your data storage mechanisms.

Before the advent of computers, all data was stored in some easily recognizable physical form. The introduction of computers simply changed the data from a physical form that you can touch and feel to a digital form that is represented by a series of 1's and 0's. Does the information that you display for an expense report on the computer screen differ greatly from the same information in the hard-copy version of the expense form? Perhaps the information is laid out differently than on the screen, but the key elements—who was paid, what amount, how much was the tax, what was the purpose of the expense, and so on—are all the same.

In looking at a database and its most basic set of characteristics, the following points hold true:

- A database stores data. The storage of data can take a physical form, such as a filing cabinet or a shoebox.
- Data is composed of logical units of information that have some form of connection to each other. For example, a genealogical database stores information on people as they are related to each other (parents, children, etc.).
- A database management system (DBMS) provides a method to easily retrieve, add, modify, or remove data. This can be a series of filing cabinets that are properly indexed, making it easy to find and change what you need, or a computer program that performs the same function.

When data began to move from a physical form to a logical form using computers, different theoretical versions of systems to manage data evolved. Some of the more common database management systems in use over the last 50 years include the *hierarchical*, *network*, and *relational*. Oracle is a relational database management system (RDBMS).

The Relational Model of Databases

The relational model for database management systems was proposed in the June 1970 issue of *Communications of the ACM*—the Association of Computing Machinery journal—by Dr. E.F. Codd, an IBM researcher, in a paper called “A Relational Model of Data for Large Shared Data Banks.” For its time it was a radical departure from established principles because it stated that tables that have related data need not know where the related information is physically stored. Unlike previous database models, including the hierarchical and network models, which used the physical location of a record to relate information between two sets of data, the relational model stated that data in one table needed to know only the name of the other table and the value on which it is related. It was not necessary for data in one table to keep track of the physical storage location of the related information in another.



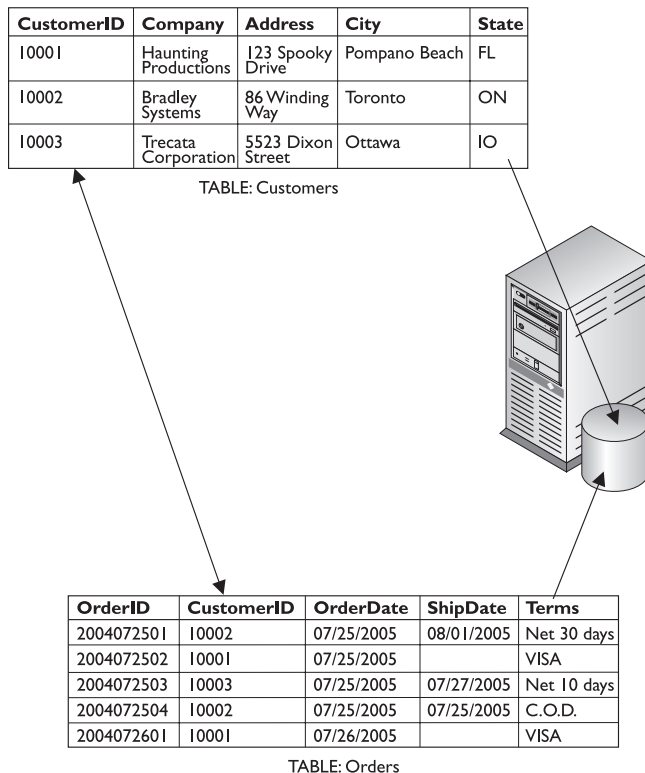
NOTE The full text of Dr. E.F. Codd’s paper “A Relational Model of Data for Large Shared Data Banks” can be found in the classics section of the ACM web site at www.acm.org/classics/nov95/toc.html.

The relational model broke all data down into collections of objects or relations that store the actual data (i.e., tables). It also introduced a set of operators to act on the related objects to produce other objects (i.e., join conditions to produce a new result set). Finally, the model proposed that a set of elements should exist to ensure data integrity so that the data would be consistent and accurate (i.e., constraints). Codd proposed a set of twelve rules that would allow designers to determine if the database management system satisfied the requirements of the relational model. Although no database today satisfies all twelve rules (because the database would run very slowly if it did, since theory is not always the same as practice), it is generally accepted that any RDBMS should comply with most of them.

The essence of the relational model is that data is made up of a set of relations. These relations are implemented as two-dimensional tables with rows and columns as shown in Figure 1-1. In this example, the Customers table stores information about clients we deal with—their customer ID, their company name, their address, and so on. The Orders table stores information about the client orders (but not the order line items—these are in another table), including the order data, the method of payment, the order date, and the ship date. The CustomerID column in both tables provides the relationship between the two tables and is the source of the relation. The tables themselves are stored in a database that resides on a computer. The physical locations of the tables need not be known—only their names.

Figure 1-1

The Customers and Orders tables are related by CustomerID.



For a database to be considered relational, and because the physical location of rows is not something that a user querying data needs to know, the table must allow for each row to be uniquely identified. The column (or set of columns) that uniquely identifies a row is known as the *primary key*. Each table in a relational database (according to database theory) must have a primary key. In this way, you are certain that the specific value appears only once in the table. In Figure 1-1, the CustomerID column of the Customers table is a primary key, ensuring that each CustomerID appears only once in the table. For the Orders table, the OrderID is the primary key.

When relating tables together (the whole point of a relational database), the value of a primary key column in one table can be placed in a column in another table. The column in the second table holding the value is known as the *foreign key*. A foreign key states that the value in this column for a row exists in another table and must continue to exist, or else the relationship is broken. In Figure 1-1, the CustomerID column of the Orders table is a foreign key to the CustomerID column in the Customers table. In order for the relationship to be valid, any value placed in the CustomerID column of the Orders table must already exist in the CustomerID column of the Customers table. In other words, in order for a client to place an order, we need to know some basic information about them. If we don't have this information, the customer cannot place an order. Oracle enforces the primary key–foreign key relationship through the use of database constraints.

SQL: The Structured Query Language

All of the relations in a relational database are managed by a relational database management system. As indicated earlier, an RDBMS allows you to manipulate relational tables and their contents. It provides a language that allows you to create, modify, and remove objects in the database, as well as add, change, and delete data. The language that Oracle uses is the Structured Query Language, or SQL. SQL was originally developed by IBM, for whom E.F. Codd worked, and was first called Structured English Query Language (or SEQUEL, for short). The name has been shortened to Structured Query Language, or SQL, but it is still pronounced *sequel*.

SQL is actually a collection of several different “languages,” each designed for a particular purpose. It is made up of the following:

- **Data definition language (DDL)** DDL is used to create and modify database objects. DDL statements include CREATE, ALTER, DROP, RENAME, and TRUNCATE. If you need to add a new table to the database, you use the CREATE TABLE statement to perform this task. To remove an index, you use the DROP INDEX statement, and so on.
- **Data manipulation language (DML)** DML is used to modify data in tables in the database. DML statements include INSERT, UPDATE, and DELETE, as well as extensions to control transactions in the database, including COMMIT, ROLLBACK, and SAVEPOINT. The SELECT statement used to query data in the database is not technically considered a DML command, although it is sometimes included with the definition of DML because it deals with the retrieval of data.
- **Data control language (DCL)** DCL is used to configure security to perform database tasks and manipulate database objects. DCL statements include GRANT and REVOKE. Permissions can be granted to allow a user to perform a task such as creating a table, or to manipulate or query data, as by performing an insert into a table in the database.

Another characteristic of an RDBMS is that tables in a relational database do not have their relationship represented by data in one table storing the physical location of the data in a related table. As you can see in Figure 1-1, the Customers table and the Orders table are related by the data that exists in the CustomerID column of both tables. The physical location on disk of each table does not factor into the relationship between them. As long as a user querying the two tables knows the column that relates them, he/she is able to formulate a SQL statement that will extract the data satisfying the condition of that relationship (also known as the “join condition”). Should one of the tables be moved to a different hard disk used to store data in the database, the relationship will still hold true.

A third characteristic of an RDBMS is that the language used to manipulate the database has a rich and varied set of operators that can be used to manipulate the data and explore the relationships between the various tables. The SQL language allows you to determine, through the proper use of operators, data that is related

between tables, data where the relationship does not hold true, and much more. The SQL language in its pure form does not, however, have any procedural elements of a programming language such as loops, conditional logic, and the use of variables. Oracle has extended SQL to include these elements through PL/SQL, a proprietary set of language elements that can be used to create stored procedures, triggers, and other subprograms.

RDBMSs have become popular in part for the preceding reasons. Nothing stays static for long in the database world. Oracle, the first commercially available relational database management system, has extended its database capabilities to support object features.

Object Relational Database Management System (ORDBMS)

Releases of Oracle prior to Oracle 8 were RDBMSs; that is, they followed the relational model and complied with its requirements, and often improved upon them. With the introduction of Oracle 8, Oracle was considered an object relational database management system—something that is even more true with Oracle 10g. An ORDBMS complies with the relational model but also extends it to support the newer object relational database model introduced in the 1980s.

An ORDBMS is characterized by a number of additional features, including these:

- **Support for user-defined datatypes** This means that users can create their own datatypes based upon the standard Oracle datatypes or other user-defined datatypes. This feature allows for more accurate mapping of business objects to database features and can reduce the time it takes to maintain databases after they have been implemented.
- **Support for multimedia and other large objects** Oracle 8 and subsequent releases up to 10g have full support for binary large objects, or BLOBs. This means that it is possible to store large amounts of information such as video clips, images, and large amounts of text in the column of a row. Even though earlier releases of Oracle had a similar feature, it lacked functionality and was not implemented in a way that conformed to object relational standards. The current implementation is much improved.
- **Full compatibility with relational database concepts** Even though object extensions have been added to Oracle, in order for it to remain an ORDBMS, it needs to conform to the requirements of an RDBMS. Because of Oracle's strong legacy as an RDBMS, its object features can be leveraged along with the relational features to provide robust solutions.

The one thing that defines Oracle as an ORDBMS is its capability to allow you to create a user-defined datatype, which becomes an object in Oracle. For example, if you wanted to use a common definition for a telephone number in several tables (Customers, Suppliers, Employees, etc.) and wanted to be sure that any changes to its characteristics would be inherited by all tables using it, you could create a new

datatype called “PhoneNumber” with the proper characteristics and then create the tables using the PhoneNumber datatype as one of the column definitions. If the rules for area codes, for example, changed, you could modify the attributes and methods of the PhoneNumber datatype and all tables would inherit the change.

Database Objects

Every RDBMS needs to support a minimum number of database objects in order to comply with the basic requirements for a relational database. Oracle supports these and many more. This chapter presents only a listing of those objects, while subsequent chapters will allow you to create and manipulate many of these objects.

Oracle’s collection of database objects includes all of those that are needed for it to be called a relational database (tables, views, constraints, etc.) as well as others that go beyond what is required and are included because they provide additional functionality (packages, object types, synonyms, sequences, etc.). The full list of database objects that Oracle 10g supports appears in Table 1-1.

Object	Description
Table	A collection of columns and rows representing a single entity (e.g., customers, orders, employees, etc.).
Column	A single attribute of an entity stored in a table. A column has a name and a datatype. A table may have, and typically does have, more than one column as part of its definition.
Row	A single instance of an entity in a table, including all columns. For example, a student row will store all information about a single student, such as that student’s ID, name, and address.
Cell	The term cell is used to refer to the intersection of a single column in a single row. For example, the CompanyName column for CustomerID 10002 in our example would be a cell holding that data—Bradley Systems.
Constraint	A database object that is used to enforce simple business rules and database integrity. Examples of constraints are PRIMARY KEY, FOREIGN KEY, NOT NULL, and CHECK.
View	A view is a logical projection of data from one or more tables as represented by a SQL statement stored in the database. Views are used to simplify complex and repetitive SQL statements by assigning those statements a name in the database.
Index	An index is a database object that helps speed up retrieval of data by storing logical pointers to specific key values. By scanning the index, which is organized in either ascending or descending order according to the key value, you are able to retrieve a row quicker than by scanning all rows in a table.

Table 1-1 Oracle 10g Database Objects

Object	Description
Index-organized table	A table whose physical storage is organized like an index. Unlike a regular table, where rows are inserted in no particular order and querying all rows will retrieve the data in random order, index-organized tables store data organized according to the primary key defined on the table. The difference between a table (referred to as storing data on a heap) and an index-organized table is like the difference between storing all of your receipts in a shoebox (i.e., in no specific order) and storing it chronologically according to the date the expense was incurred. Taking the receipts out of the shoebox will result in no specific logic in their retrieval, while doing the same when the receipts are organized chronologically will allow you to predict that the June 2 receipt will appear before the August 1 receipt.
Partition	Tables in Oracle 10g can be cut into pieces for more efficient physical storage. A partition (or subpartition) holds a subset of the table's data, typically on a separate physical disk, so that data retrieval is quicker either by allowing reads from more than one physical disk simultaneously (multipartition parallel reads) or by not reading a partition's data at all if it is not required to satisfy the query (partition elimination).
Cluster	A storage mechanism object that allows rows from more than one table to be physically stored together for quicker retrieval. For example, if you store the Order information (customer, payment info, delivery details, etc.) in one table and the line items (item, cost, sale price, quantity, etc.) in a different table, you will need to perform at least two reads to retrieve information about an order: one for the order info and the second for line item info. Creating both tables on the cluster organized by the order ID will allow Oracle to place the order and line item data for the same order ID on the same physical block, thereby reducing retrieval of that order's information to a single read. The downside of clusters is that they force you to preallocate a certain portion or all of the disk space they require when rows are added or the cluster is created.
Sequence	A sequence allows you to create and increment a counter that can be used to generate numerical values to be used as primary key values for a table.
Synonym	As in the English language, a synonym is another name for an existing object. Synonyms are used in Oracle as shorthand for objects with long names, or to make it easier to remember a specific object.
Stored procedure	A stored procedure is a collection of SQL and PL/SQL statements that perform a specific task, such as to insert a row into a table or to update data.
Trigger	A trigger is a special kind of stored procedure that cannot be invoked manually but rather is automatically invoked whenever an action is performed on a table. Triggers can be associated with a table and a corresponding action such as INSERT, UPDATE, or DELETE as well as system events such as user logon and logoff, or database STARTUP and SHUTDOWN.
Function	A function is a stored program that must return a value. Unlike stored procedures, which can have parameters passed to them and do not need to return any value as output, a function must return a value.

Table 1-1 Oracle 10g Database Objects (*continued*)

Object	Description
Package	A package is a collection of stored procedures and functions grouped under a common name. This allows you to logically group all program elements for a particular part of the database under a single name for maintenance and performance reasons.
User-defined datatype	A user-defined datatype is a database object that can be used in any table or another object definition. Using user-defined datatypes allows you to ensure consistency between tables and also lets you apply methods (i.e., actions that can be performed by the object) as part of the definition.
BLOB	A BLOB is a binary large object used to store video, images, and large amounts of text. BLOBs are defined as a column in a table and can be one of several datatypes: BLOB, CLOB, NCLOB, or BFILE.

Table 1-1 Oracle 10g Database Objects (*continued*)

Oracle also includes other objects that are beyond the scope of this book, including dimensions, directories, materialized views, and more. These objects are created for a specific purpose such as to facilitate data warehousing and their full coverage is beyond the scope of this book.

Each object in an Oracle database is owned by a user. A user defined in an Oracle database does not have to own any objects, but those that do are known as *schema users*. A *schema* is a collection of all objects owned by a particular user, including tables, indexes, views, and so on.

The Oracle Data Dictionary

As you may well imagine, a database may contain hundreds and even thousands of objects. Keeping track of all this information is the job of the Oracle data dictionary. A *data dictionary* in any database contains metadata information. Metadata is “data about data,” or a set of tables and other database objects that store information about your own tables and database objects.

The data dictionary in Oracle is a set of tables, called base tables, which contain the most basic information about user-created database objects. These base tables are owned by an Oracle user called SYS, which is created when the database itself is created. The base tables are never accessed directly, as their names are cryptic by design to discourage users from querying and modifying them. To make it easier to access the data dictionary and get information on objects in the database, a series of views are created during the database creation process. These views are commonly referred to as data dictionary views.

Oracle has three sets of data dictionary views. They are as follows:

- **USER_ views** These views allow users to get information on objects that are in their schema (i.e., objects that they have created and own).

- **ALL_ views** These views allow users to get information on objects that they own or that they have been given access to. The ALL_ views contain a subset of the information presented in the USER_ views for the same object and allow users to find out what other objects they are allowed to reference or manipulate in the database, in addition to the objects that they own.
- **DBA_ views** The DBA_ views, designed to be used by the database administrator (DBA) of the database, provide a full set of information for objects in the database, i.e., any object created by any user. Normal users do not have access to these views, as special privileges are needed to SELECT from them.

As you delve further in this book, you will be introduced to many DBA_ views to help you in your duties as a database administrator. But, what does a database administrator do in the Oracle world?

Responsibilities of a Database Administrator

One of my colleagues likes to comment that users of the databases for which he is responsible think they control the databases. The reality, as he quite correctly puts it (if in a control-freakish sort of way), is quite different. As a DBA, he can do whatever he wants in any database he is responsible for, so he's the one with control. Database administrators do have a great deal of power, but it is important to remember that with great power also comes great responsibility. Ultimately, the success and failure of a database to respond to user requirements and satisfy corporate objectives rests with the DBA. The DBA must take the blame and the praise for good or bad database management.

The kinds of tasks DBAs in the Oracle world are responsible for include the following:

- **Sizing and evaluating server hardware** As the individual responsible for the smooth operation of databases in your organization, you will be called upon to suggest the configuration of the server that will be used to run Oracle. Your experience will play a key role here in determining the amount of memory, hard disk, CPU, and other resources required to support the target database's operations. Understanding the architecture of Oracle and the data needs of the business and the application will help you perform this task.
- **Installing Oracle software and updates** After you buy the software, the first thing you need to do is bring it up. Installation of the Oracle software on the target platform is the job of the DBA. It usually involves more than putting in the CD and answering the prompts of the setup program because Oracle is a very powerful system comprising a complex piece of software that has many hooks and interactions with the operation system. Ensuring that the software is installed and working properly is a key to being a successful DBA.

- **Planning and designing the database structure** Once the software is installed, you need to make sure that the layout of the physical data structures and logical elements of Oracle is done in an optimal way. If this is not the case, performance will suffer and users will make their displeasure known. If you have properly sized the hardware, this should be an easy task, since you should have taken the database size and structure into account; if you inherited the environment, you may need to use your expertise to determine the optimal configuration.
- **Creating databases** As you will see in Chapter 3, this is a somewhat anticlimactic task. Creation of the database is the first step to administering it. Although relatively straightforward, the process can run into problems; with experience, however, you should grow well equipped to fix these problems.
- **Backing up databases and implementing other ways to safeguard the data** Once a database is in production and users are connecting to it, they may not take it well if the database becomes unavailable. Even worse, if data is lost it could mean lost productivity, sales, and customers. Ensuring that a database is always available to users, that data loss is minimized, and that recovery is quick and complete is perhaps one of the most important responsibilities of the DBA.
- **Creating and maintaining database users** Once a new user needs to gain access to the database or when the requirements and permissions of another user change, the DBA must be able to make the necessary security modifications to ensure appropriate access. In some cases, application developers may not make use of Oracle's built-in security fully, so being able to recognize these situations and take appropriate action is also necessary.
- **Implementing application and database designs** Organizations may purchase third-party software applications or hire database architects to design a database to suit a specific database requirement when in-house expertise is lacking. However, the actual implementation of these designs will be undertaken by the DBA, since the DBA will be responsible for ensuring that the database continues to work properly after the software vendor or database architect leaves.
- **Restoring and recovering databases** Sometimes things go wrong. Hardware fails, users improperly modify or delete data, or a natural disaster or some other calamity befalls the data center. Being able to recover from a variety of scenarios is critical. This is when the fault tolerance disaster recovery strategy is tested for real—but it should also be tested in mock scenarios to ensure it works. The DBA is the one that is answerable for their success or failure.
- **Monitoring and tuning database performance** In *Star Trek: The Next Generation* there is an episode where the Enterprise assists a stranded vessel. The vessel's occupants are somewhat lacking in both engineering and communication skills, but they do ask Captain Picard and Giordi to *make us go fast*. Your users will frequently comment that the database is not fast

enough. Keeping those comments to a minimum and solving performance problems when (or before) they occur will reduce your stress level and increase job satisfaction.

As you can tell, the duties of a DBA are wide and varied. It is an exciting position whose goal can sometimes be summed up in one phrase: reduce the number of times the phone rings in a day. If you get a lot of phone calls from users or management, this is probably not a good sign and can make your day an unhappy one. Well-tuned databases on appropriate hardware with good disaster recovery and backup strategies will reduce your phone calls, make happy users, and increase your job satisfaction. Sounds simple, right?

The Oracle Product Family

As an Oracle database administrator, you may be responsible for a database that is actually part of another Oracle product or that complements or is relied upon by that other product. Oracle has long since become more than just a database company—though the Oracle database is at the heart of what Oracle Corporation does best. In addition to the database, Oracle Corporation also offers these products, among others:

- **Oracle E-Business Suite** Also referred to as Oracle Applications, the E-Business Suite is the premier enterprise resource planning (ERP) suite in the industry. Encompassing many modules, including financials, human resources, distribution, manufacturing, and many others, the E-Business Suite allows organizations to leverage Oracle technology to better run their businesses and increase profitability and customer satisfaction. Oracle E-Business Suite makes use of the Oracle database, Application Server, and Developer Suite to run and enhance the application.
- **Oracle Collaboration Suite** One of Oracle's newest products, Collaboration Suite encompasses e-mail, fax, calendaring, web conferencing, a files repository, voice mail, and other modules to provide a single integrated platform for an organization's collaboration and communication infrastructure. Like Oracle E-Business Suite, Collaboration Suite also incorporates the Oracle database and Application Server as its foundation.
- **Oracle Application Server** A Java 2 Enterprise Edition (J2EE)-certified server platform, Oracle Application Server integrates everything you need to deploy and develop web-based applications. Oracle Application Server includes a portal server and tools to develop portlets, Web Services capabilities, content management, support for forms-based applications using Oracle Forms, reporting using Oracle Reports, development using PL/SQL or Java, and many other features. It also includes the capability to configure and use a single sign-on server that interfaces with other third-party directory services such as Microsoft's Active Directory, Novell's NDS, and Sun's iPlanet.

- **Oracle Developer Suite** If you need to develop applications for the Oracle database or Application Server, Oracle Developer Suite provides tools for the development of Java, PL/SQL, business intelligence, data warehousing, and other applications.
- **Oracle Consulting** When you need assistance in your Oracle deployment, Oracle Consulting can provide the expertise and individuals to help your projects succeed.
- **Oracle University** The training of customers and partners on Oracle products and technologies is the responsibility of Oracle University and its partners. Instructor-led and online courses are available.

More information on the products available from Oracle Corporation can be found on the Oracle web site at www.oracle.com.

Chapter Review

In this chapter you learned a bit about the history of the relational database management system (RDBMS) concept. You also learned that SQL is actually three languages: a data control language (DCL) for managing permissions on database objects, a data definition language (DDL) for creating and managing those objects, and a data manipulation language (DML) for adding, updating, or deleting data from the database, as well as controlling those transactions. You saw what objects may appear in an Oracle 10g database and read a brief description of each object, and you were briefed on your responsibilities as a database administrator. Finally, you learned that Oracle is not just a database company but one with a wealth and breadth of products and services. At this point, you're ready to start working with Oracle (after a few questions).

Questions

1. You need to recommend a platform for the deployment of your web-based application written in Java. You need to make the management of web page content easy as well as integrate security with your Active Directory infrastructure. Which Oracle product will satisfy your requirements? (Choose the best answer.)
 - A. Oracle Database
 - B. Oracle Application Server
 - C. Oracle Collaboration Suite
 - D. Oracle E-Business Suite
 - E. Oracle Developer Suite
2. For which of the following types of information stored in a database would the use of a sequence be appropriate? (Choose two correct answers.)

- A. Invoice line item
 - B. Invoice number
 - C. Employee name
 - D. Atomic element
 - E. Customer identifier
3. What is a key benefit to making use of user-defined datatypes in Oracle? (Choose the best answer.)
- A. Ability to rename Oracle built-in datatypes
 - B. Inheritance
 - C. Polymorphism
 - D. Consistency of similar data structures across multiple tables
 - E. Easier maintenance of databases
4. Your organization has outgrown its hosted e-mail system. You also need to implement web conferencing. Development of an interface to your in-house telephone system will take place and needs to be integrated with a new voice mail platform. Your developers are versed in Java on a Linux platform. Which Oracle product provides the best fit for your organization's requirements? (Choose the best answer.)
- A. Oracle Database
 - B. Oracle Application Server
 - C. Oracle Collaboration Suite
 - D. Oracle E-Business Suite
 - E. Oracle Developer Suite
5. Which of the following is not typically a responsibility of an Oracle database administrator? (Choose the best answer.)
- A. Creating new users
 - B. Creating database objects
 - C. Installing Oracle software
 - D. Application development to manipulate database data
 - E. Backing up the database

Answers

1. **B.** Oracle Application Server provides a Java 2 Enterprise Edition (J2EE)-certified server for deploying your web-based application. It also includes a portal server for creating new content and a content management server for managing changes to content. Finally, the Oracle Internet Directory component allows for integration with third-party directory services to provide single sign-on functionality.
2. **B and E.** A sequence is an automatically generated chronological number, ideal for invoice numbers or unique identifiers for database data, such as a customer ID. Because a sequence is numeric in nature, it is not appropriate for text-based data such as an order line item, employee name, or atomic element.
3. **D.** Using a user-defined datatype, you can ensure that the same data is always stored the same way in all tables. In other words, a phone number will always have a consistent structure no matter what table it is in. It can make maintenance of the database easier as well, but this is a secondary benefit.
4. **C.** Oracle Collaboration Suite provides e-mail, fax, and voice mail functionality, as well as web conferencing, calendaring, and other features. Because it also includes Oracle Application Server, it is the perfect product for the development and deployment of a custom application to interface with the phone system.
5. **D.** The development of the application that will interface with the database and manipulate the data within it is not something that a DBA does. This task is performed by an application developer. The DBA is responsible for all other tasks listed in the question, including database security, software installation, backup and recovery, and the creation and management of database objects.

