ALICE RISCHERT

Oracle SQL by Example

fourth edition

- The simplest, fastest way to master Oracle SQL!
- Build your skills with hundreds of hands-on exercises and self-review questions
- Focuses on real-world tasks
- Fully updated for Oracle 11g
- Covers all the key concepts and techniques every developer, analyst, and designer needs

Foreword by Peter Koletzke, Technical Director and Principal Instructor, Quovera, Oracle Certified Master, and Oracle ACE Director

THE ORACLE SERIES THE
INDEPENDENT VOICE ON ORACLE

The publisher offers excellent discounts on this book when ordered in quantity for bulk purchases or special sales, which may include electronic versions and/or custom covers and content particular to your business, training goals, marketing focus, and branding interests. For more information, please contact:

U.S. Corporate and Government Sales
(800) 382-3419
corpsales@pearsontechgroup.com

For sales outside the United States please contact:

International Sales
international@pearson.com

Visit us on the Web: informit.com

Library of Congress Cataloging-in-Publication Data:

Rischert, Alice.

Oracle SQL by example / Alice Rischert.

p. cm.

ISBN 978-0-13-714283-5 (pbk. : alk. paper) 1. SQL (Computer program language) 2. Oracle (Computer file) I. Title.

QA76.73.S67R57 2009

005.13'3—dc22

2009022733

Pearson Education, Inc.
Rights and Contracts Department
501 Boylston Street, Suite 900
Boston, MA 02116
Fax (617) 671-3447

ISBN-13: 978-0-137-14283-5
ISBN-10: 0-137-14283-8
Text printed in the United States on recycled paper at Edwards Brothers in Ann Arbor, Michigan
First printing August 2009

Editor-in-Chief: Mark Taub

Acquisitions Editor: Trina MacDonald

Development Editor: Michael Thurston

Managing Editor: Kristy Hart

Technical Editors: Oleg Voskoboynikov
Robert Vollman

Project Editors: Julie Anderson
Jovana San Nicolas-Shirley

Copy Editor: Kitty Wilson

Indexer: Alice Rischert

Proofreader: Sheri Cain

Publishing Coordinator: Olivia Basegio

Interior Designer: Gary Adair

Cover Designer: Anne Jones

Compositors: Nonie Ratcliff
Gloria Schurick

FOREWORD

The Ancient Problem

The year was 1680; the place was the Levant—a region on the eastern shores of the Mediterranean. The political climate was stable enough to support peaceful trade among nations, and seaports were bustling with merchants from many countries. These merchants found trade outside their native land profitable, but they faced many challenges in travel, as well as in adapting to foreign cultures and seemingly strange customs. One major challenge was communication. As a seller of goods, a key sales technique is verbally extolling the virtues of your product to a potential customer, but how do you market to a target audience whose language you have not mastered? Sign language and written symbols can be quite effective but are not nearly as personal and understandable (and beneficial to sales) as well-chosen words. The problem with spoken language in a melting pot like the Levant is that there are many languages to master.

The Ancient Solution

The solution that emerged to this problem nearly a half-century ago has been used by people of many lands throughout history: When gathered in a multilingual environment, speak a common language. In the Levant, the common language that evolved was a combination of Italian, Spanish, French, Greek, Arabic, and Turkish. This language, called *Lingua Franca*, became a standard in many ports, as merchants used it to successfully communicate with their customers and fellow traders. Those who learned and mastered this language became the most effective and successful businesspeople. Although Lingua Franca is now extinct, the term *lingua franca* is still applied to any common language used for communication between people of diverse native languages.

The Lingua Franca of IT

Today's information technology (IT) industry uses the concept of *lingua franca*. Businesses solve their data management requirements by using relational databases. The challenges that companies face when using relational databases are mainly in creating flexible and efficient human interfaces. Developing customized user interfaces often takes much in the way of effort and resources if it is to properly fulfill the requirements of the business.

IT professionals who are tasked with creating modern database systems must rely on their training and expertise, both in designing the proper database storage objects and in programming the most efficient application programs. In the recent past, developers of relational database systems have created application programs using procedural programming languages such as COBOL, Fortran, and C. Regardless of the procedural language used, there is an additional language embedded in these procedural programs. This language offers a standard way to define and manipulate the data in a relational database and is, therefore, the lingua franca of database technology: Structured Query Language (SQL).

With SQL, developers using diverse programming languages, database software, hardware, and operating systems can effectively communicate with the database in a completely standard way.

A standard language means that the written code is understood easily and can be supported fully and enhanced quickly. This is the promise of SQL, and this promise has been fulfilled successfully for decades, in countless relational database application programs.

The Modern-Day Problem

The problem today is not the lack of a common language, as was the case in the days before a lingua franca. SQL is the common language. The problem today is in the assimilation and proper use of this language. As the computer industry continues its logarithmic growth, the number of application developers increases similarly. Each new developer who writes programs that need to access a relational database must be trained in the lingua franca. In the past, with a small number of new developers, this was a manageable feat for most organizations. When a new developer was in training, she or he would learn both a procedural programming language and the database language embedded in it—SQL.

Today, IT solutions are leaning toward object orientation. Object-oriented analysis, design, and programming have come of age and, according to popular opinion, are now the best tools for creating computer systems. As a result, object-oriented languages, such as Java, C#, and C++, are replacing the traditional procedural languages for building new systems. From the perspective of a database management system, this is merely a shift in the main programming language that the developer uses to create the application's user interface. The core database language, SQL, is still required and is of key importance.

However, something fundamental has been lost in this paradigm shift to object orientation. That something is a solid background in SQL. This problem persists for many reasons. Unfortunately, IT management professionals everywhere are employing object-oriented programmers who do not have a solid grasp of SQL. This arrangement works up to a point, but there is, eventually, a collision with a brick wall. In many situations, as a consultant, I have had to break into the SQL code used in a Java program and found inefficient or incorrect use of SQL. Misunderstanding or misuse of SQL can have adverse effects on the program's efficiency and can drag down the performance of production data systems, which are critical to the functioning of the business. On the other hand, proper use of SQL results in application programs being simpler to create and maintain and being most efficient in data storage and data access.

The Modern-Day Solution

The solution to the problem of misunderstanding the lingua franca of databases is simple: an increased focus on learning the foundations and abilities of SQL and the correct methods for coding SQL programs. Time and money spent for training on the basics of SQL are time and money well spent. There are many ways to learn a subject like SQL. In my experience of over 25 years as a trainer in the IT and other industries, the best learning experience results from a multifaceted approach. Human beings learn in different ways, and using different approaches to present the same subject ensures that the subject will be understood and assimilated. In addition, the repetition of concepts and material in different formats ensures mastery of the subject. Research has proven that repetition and variety are key learning techniques.

What Ms. Rischert has accomplished in this book is the epitome of the solution for the correct understanding of SQL. This book will be useful for seasoned IT professionals who need to study the language more completely or who just need a refresher on the concepts and thought processes of using SQL. It will also be useful for those who are new to the subject and are interested in learning it thoroughly and in the right way.

The author applies the key teaching principles of variety and repetition in a multifaceted approach to the subject. This approach allows the student of the subject to fully grasp the basics as well as the best ways to use the language. All core SQL subjects are treated with this approach, which includes a brief description of the concept followed by simple and easy-to-grasp examples. Examples that you can try using an Oracle database make the concepts real and allow you to quickly grasp the subject. In addition, exercises and quizzes make you think about the concepts you just learned and therefore help you assimilate them.

The very best way to learn the concepts of any technology is to be tasked with an application development project that uses that technology. However, a project in the real-world workplace is not the right place to learn; the stakes are too high and the pressure too great. This book gives you a taste of real-world application development needs and processes by assigning workshops where you further apply the knowledge gained from the conceptual discussions, examples, exercises, and quizzes. The book then provides the solutions to all questions presented so that you can check your work. Proper use of these follow-up sections will lead you to a solid mastery of the language and give you the ability to use this lingua franca successfully to solve real-world business problems. This type of mastery will serve you well in whatever type of database programming environment you find yourself.

Peter Koletzke
Quovera
San Carlos, California
May 2009

PREFACE

SQL is the de facto standard language for relational databases, and Oracle's database server is the leading relational database on the market today. *Oracle SQL by Example,* 4th edition, presents an introduction to the Oracle SQL language in a unique and highly effective format. Rather than being a reference book, it guides you through basic skills until you reach a mastery of the language. The book challenges you to work through hands-on guided tasks rather than read through descriptions of functionality. You will be able to retain the material more easily, and the additional exercise and quiz questions reinforce and further enhance the learning experience.

Who This Book Is For

This book is intended for anyone requiring a background in Oracle's implementation of the SQL language. In particular, application developers, system designers, and database administrators will find many practical and representative real-world examples. Students new to Oracle will gain the necessary experience and confidence to apply their knowledge in solving typical problems they face in work situations. Individuals already familiar with Oracle SQL but wanting a firmer foundation or those interested in the new Oracle 11g features will discover many useful tips and tricks, as well as a wealth of information.

The initial audience for the book was the students of the Oracle SQL class at Columbia University's Computer Technology and Applications program. The student body typically encompassed a wide-ranging level of technology experience. Their questions, learning approaches, and feedback provided the framework for this book. Many students cited the hands-on exercises as critical to their understanding of database technology and SQL and continuously asked for more examples and additional challenging topics. This book shares much of the material presented in the classroom and looks at the various angles of many solutions to a particular issue.

The book begins with the basic concepts of relational databases, the SQL Developer and SQL*Plus tools, and SQL; it then gradually deepens the knowledge. Whether you already have some fundamental understanding of SQL or not, you will benefit from reading this book as it allows you to gain insight into writing alternative SQL statements. After performing the exercises in this book, you will harness the power of SQL and utilize much of Oracle's SQL functionality.

What Makes This Book Different

This book encourages you to learn by doing, to actively participate by performing the exercises, quizzes, and Workshop exercises. Ultimately, the reward is a thorough understanding of SQL and a high level of comfort dealing with real-world Oracle SQL topics. Performing the exercises aids in retention of the material, and the quizzes and Workshop sections further test your understanding and offer additional challenges. The companion Web site, located at www.oraclesqlbyexample.com, provides solutions to the Workshop exercises and includes additional exercises and answers.

The book's focus is on providing examples of how the SQL language is commonly used, with many exercises supporting the learning experience. Unlike other SQL books, this book discusses Oracle's specific implementation of the language. Learning the language alone is not enough. The book also teaches you how to adopt good habits and educates you about many Oracle-specific technology features that are essential to successful systems development. The examples are based on a sample database that takes you through the typical tasks you will encounter when working with an Oracle database.

This book is not a reference book; rather, it teaches SQL by illustrating its use through many examples. Take advantage of the index to look up concepts and refer to certain topics. The best way to learn the SQL language is to perform the exercises and compare your answers with the sample answers and accompanying explanations. Additional teaching points are part of the answers; the idea is that after you have performed an exercise, you are ready for additional, more advanced material.

This book does not cover the entire Oracle SQL syntax but emphasizes the essentials of the most frequently used features, with many examples to reinforce your learning. Some of Oracle's syntax options are too numerous, and many are very infrequently used; including them all would make the book swell by several hundred additional pages. Instead, I have concentrated on those that you will use most. After performing the exercises, you will also have gained sufficient knowledge to read up and understand the Oracle documentation, if needed. I hope that you will enjoy this learning experience and come away with the knowledge you hoped to gain.

How This Book Is Organized

Each chapter of this book is divided into labs covering particular topics. The objective of each lab is defined at its beginning, with brief examples that introduce you to the covered concepts.

Following the lab's introductory examples are exercises, which are the heart of the lab. They reinforce and expand your knowledge of the subject. Each exercise consists of a series of steps to follow to perform specific tasks or particular questions that are designed to help you discover the important aspects of the SQL language. The answers to these questions are given at the end of the exercises, along with more in-depth discussion of the concepts explored.

After you perform the exercises and compare the answers with the sample queries, answers, and explanations, you can move on to the multiple-choice quiz questions. These are meant to test your understanding of the material covered in the lab. The answers to these questions appear in Appendix A, "Answers to Quiz Questions." There are additional quiz questions at the book's companion Web site, located at www.oraclesqlbyexample.com.

At the end of each chapter, the Workshop section reinforces and combines all the topics learned in labs and helps you solidify your skills. The answers to these questions are provided on the companion Web site for this book.

Each chapter is laid out as follows:

Chapter introduction

Lab

Exercises

Exercise Answers (with detailed discussion)

Quiz

Lab

...

Workshop

The chapters should be completed in sequence because concepts covered in earlier chapters are required for the completion of exercises in later chapters.

What You Need

To complete the exercises, you need the following:

- The Oracle database software
- Oracle's SQL Developer or SQL*Plus software
- Access to the Companion Website

ORACLE 11*G*

Oracle 11g is Oracle's latest version of the relational database software and its flagship product. To follow along with this book, you can use either the Standard Edition or Enterprise Edition. The Enterprise Edition of Oracle version 11.1.0.6.0 was used to create the exercises for this book.

If you have a previous version of the Oracle database, you will be able to complete a large majority of the exercises; however, some syntax options and features are available only in Oracle 11g.

If you do not have the latest release of the Oracle software available, you can obtain a trial copy from Oracle's Web site, at www.oracle.com. You have the option of either downloading the Standard Edition or Enterprise Edition or purchasing a CD media pack from the Oracle store (https://oraclestore.oracle.com) for a nominal fee.

ORACLE SQL DEVELOPER AND SQL*PLUS

You can perform most of the exercises in this book with Oracle's SQL Developer or SQL*Plus software. Both software tools are included with the Oracle database and are part of the default installation.

SQL Developer is easier to use and offers a superior display of results. Because SQL Developer is a relatively new product and independent of the Oracle database release, you may find later versions of the software on Oracle's Web site, as a separate product download.

The book's screenshots and described functionality are based on SQL Developer 1.5.3. This tool sets the future product direction and has many useful features to improve your productivity and enhance your SQL knowledge and skill set.

This book is not intended as a comprehensive SQL Developer book. Rather, the focus is SQL. You use SQL Developer as part of your interaction with the Oracle database. Oracle's intended target audience for the SQL Developer tool is not only SQL users but also PL/SQL programmers. Therefore, you'll find that menu options in this tool are related to PL/SQL. The purpose of this book is not to describe and point out every menu option but to teach you the SQL language. SQL Developer is a useful tool for executing SQL statements and helping you understand the effects of the commands. The coverage of SQL Developer's core functionality is intended within the context of SQL.

Using the SQL Developer graphical user interface does not eliminate the need to know and understand the SQL language. SQL Developer is a productivity tool that makes you a more efficient and proficient user of the language. You still need to know and understand your actions, and the tools can eliminate some basic mistakes and alleviate some frustrations for a beginning user.

SQL*Plus has been part of Oracle since its early beginnings and will continue to be shipped with every installation and platform. It has such a long history and defined purpose in Oracle's strategy that you cannot ignore SQL*Plus altogether. Therefore, you will gain some basic knowledge of this tool as part of the exercises in this book. This book points out the relevant differences between SQL Developer and SQL*Plus, and you will gain an appreciation for each tool's inherent benefits. You can choose to perform most of the exercises in either tool, depending on your preference. Because the book cannot comprehensively cover both tools and the SQL language, it focuses on the SQL Developer and SQL*Plus features you will use most.

Instead of executing your statements in SQL Developer or SQL*Plus, you can also use other SQL execution environments; a list of such environments is provided in Appendix H, "Resources."

Access to the Companion Web Site

The companion Web site, located at www.oraclesqlbyexample.com, provides the following.

- Installation files you need to download before you begin reading the book
- Answers to the Workshop questions
- Additional Quiz questions
- Additional resources, such as links, tips, and errata

INSTALLATION FILES

All the exercises and quiz questions are based on a sample schema called STUDENT. You can download the required files to install the STUDENT schema and the installation instructions from the companion Web site. (The sample database is identical to the companion book of this series, *Oracle PL/SQL by Example,* 4th Edition, by Benjamin Rosenzweig and Elena Silvestrova Rakhimov; Prentice Hall, 2008.)

WORKSHOP

The answers to the Workshop sections are provided at the companion Web site.

ADDITIONAL QUIZ QUESTIONS

The Web site provides many other features, such as additional quiz questions and periodically updated information about the book.

Visit the companion Web site and download the required files before starting the labs and exercises.

About the STUDENT Schema

Throughout this book, you access data from a sample schema called STUDENT, which contains information about a computer education program. The schema was designed to record data about instructors, courses, students, and their respective enrollments and grades.

After you download the installation files to create the schema within your Oracle database, you will be able to follow the exercises in the book. Chapter 1, "SQL and Data," introduces you to the relational concepts necessary to read a schema diagram. Appendix D, "STUDENT Database Schema," shows a graphical representation of the schema, and Appendix E, "Table and Column Descriptions," lists descriptive information about each table and column.

Conventions Used in This Book

Several conventions used in this book help make your learning experience easier. These are explained here.

This icon denotes advice and useful information about a particular topic or concept.

This icon flags tips that are especially helpful tricks to will save you time or trouble—for instance, a shortcut for performing a particular task or a useful method.

Warnings are provided to warn you about any potential issues regarding the safety of your database or to save you headaches in the long run.

Errata

I have made every effort to make sure there are no errors in the text and code. However, to err is human. At the companion Web site (www.oraclesqlbyexample.com), you will find corrections as they are spotted. If you find an error that has not been reported, please let me know by contacting me at ar280@yahoo.com. Your comments and suggestions are greatly appreciated.

CHAPTER 2

SQL: The Basics

CHAPTER OBJECTIVES

In this chapter, you will learn about:

- ▶ The SQL Execution Environment
- ▶ The Anatomy of a SELECT Statement
- ▶ An Introduction to SQL Developer and SQL*Plus

Now that you are familiar with the concepts of databases and schema diagrams, you are ready to start with hands-on exercises. In this chapter, you will learn the basics of the SQL language and use SQL Developer and SQL*Plus, two Oracle provided software tools that allow you to execute statements against the Oracle database.

SQL statements can range from very simple to highly complex; they can be a few words long or a few hundred words long. In this chapter, you begin by writing simple SQL statements, but you will be able to build longer, more complex SQL queries very quickly.

LAB 2.1
The SQL Execution Environment

LAB OBJECTIVES

After this lab, you will be able to:

- Understand Database Connectivity
- Execute Your First SQL Command Using SQL Developer

This lab provides you with an understanding of the basic SQL language commands. You will learn how to establish connectivity to the database server and begin executing SQL commands.

Oracle software runs on many different operating systems and hardware environments. The machine on which the Oracle database software resides is called the *Oracle database server*. A variety of tools are available to access data from the database server. In this chapter, you will be introduced to two Oracle-provided tools: SQL Developer and SQL*Plus.

The most striking difference between SQL Developer and SQL*Plus is the interface. SQL*Plus has an arcane command-line interface with old-style editing and display options.

SQL Developer is a recent addition to the Oracle tool set. It is included in the latest Oracle releases or can be downloaded free from Oracle's Web site. SQL Developer's graphical user interface greatly simplifies SQL statement execution and overall database access.

Because SQL Developer is a much easier environment to use than SQL*Plus, you start with this execution environment first to learn the basics of the SQL language.

In subsequent labs, you will explore some of the basics of SQL*Plus. While SQL*Plus seems quite outdated, you cannot ignore decades of SQL*Plus usage. It has been part of Oracle since its early beginnings and will continue to be shipped with every installation on every platform. You will find it useful to have some rudimentary knowledge of SQL*Plus. Therefore, this book describes both tools.

However, the focus of this book is learning the SQL language; the tool you use is simply the environment within which to execute the SQL language commands. Therefore, not all details of these tools are covered. Furthermore, you may also consider one of the many third-party tools available to execute your statements. No matter what tool becomes your favorite, it is beneficial to know both SQL Developer and SQL*Plus as they are found with almost every Oracle installation.

For a list of the many easy-to-use third-party tools, see Appendix H, "Resources."

Accessing the Oracle Database Server

You can access the Oracle server through various front-end tools. This lab teaches you some of the basics of SQL Developer first. This tool is Oracle's newest database query and development tool. It is also by far easier to learn and use than SQL*Plus, which is covered in Lab 2.3, "An Introduction to SQL*Plus."

The differences between SQL Developer and SQL*Plus are pointed out to you as you work through the book. You can assume, with very few exceptions, that the functionality is very similar, if not identical. You might want to use SQL Developer for execution of your SQL statements in this book because it is a more user-friendly tool than SQL*Plus for a beginning SQL user.

Getting Started with SQL Developer

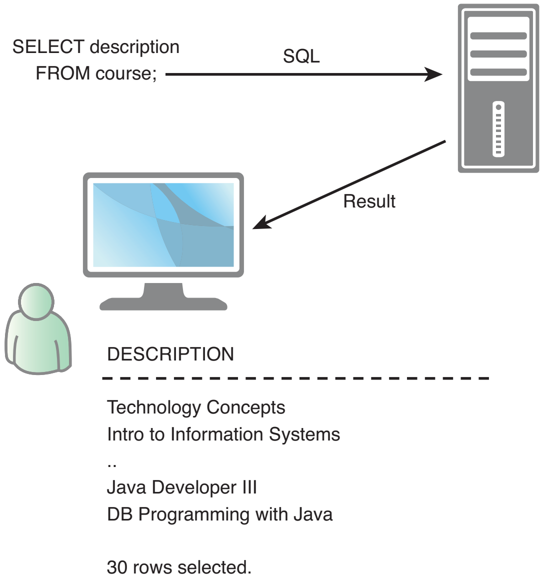
Oracle SQL Developer provides a convenient way to perform essential database tasks. The tool enhances productivity and simplifies database development tasks by providing a graphical interface for executing SQL statements, browsing, and creating and updating database objects. SQL Developer connects to any Oracle database, version 9.2.0.1 and later. You can also create database connections for non-Oracle databases.

If your Oracle software installation did not come with the tool already installed, you can download the latest version of SQL Developer from Oracle's Web site. Oracle does not charge a license fee for SQL Developer. This tool is written in Java, thus providing a uniform interface across the Windows, Linux, and MAC OS X platforms. Furthermore, SQL Developer's default database connection uses a thin Java Database Connectivity (JDBC) driver, so there is no requirement for a full Oracle client software installation involving Oracle Net. This simplifies the configuration and minimizes the footprint. With a quick unzip and execution of the file, the installation is a breeze.

SQL and the Oracle Database Server

In the midst of all this software lies the SQL language. SQL commands are sent from SQL Developer, also known as the client, or *front end,* to the server, or *back end.* These commands send instructions to the server to tell the server what services to provide. The server responds by returning a result to the client, which then displays the output. Figure 2.1 shows a SQL statement that queries the DESCRIPTION column of the COURSE table. The SQL statement is sent to the Oracle server, and the result is returned to the front end, which then formats and displays the output, as appropriate.

You may run SQL Developer and your database on the same machine. Typically, this is the case when you install both the Oracle database server and SQL Developer on your individual computer.



```
SELECT description
  FROM course;
```

```
DESCRIPTION
-----------------------
Technology Concepts
Intro to Information Systems
..
Java Developer III
DB Programming with Java

30 rows selected.
```

FIGURE 2.1
SQL and the Oracle database server

The client, whether SQL Developer or SQL*Plus, sends SQL statements to the server, and the server responds with the result set. The job of the database server involves listening and managing many clients' requests, because there are often multiple client machines involved.

The means of communication is established either via the Oracle Net software, a JDBC driver, or an ODBC driver.

Creating a Database Connection for SQL Developer

Before you can send your first SQL statement to the database, you need to create a connection to the database server. A connection consists of a username, password, and connection string or hostname. This connection authenticates you to log in to the Oracle database.

When you first evoke SQL Developer, you a see screen similar to Figure 2.2. The screen is divided into several panes. The left pane, labeled Connections, allows for a list of database connections.

The name of the displayed connection is local. This database connection refers to a database installed on the same machine as SQL Developer. You can rename this connection by right-clicking the connection name, choosing Rename Connection, and providing a new name.

To create a new database connection, right-click the Connections node and choose New Connection (see Figure 2.3).

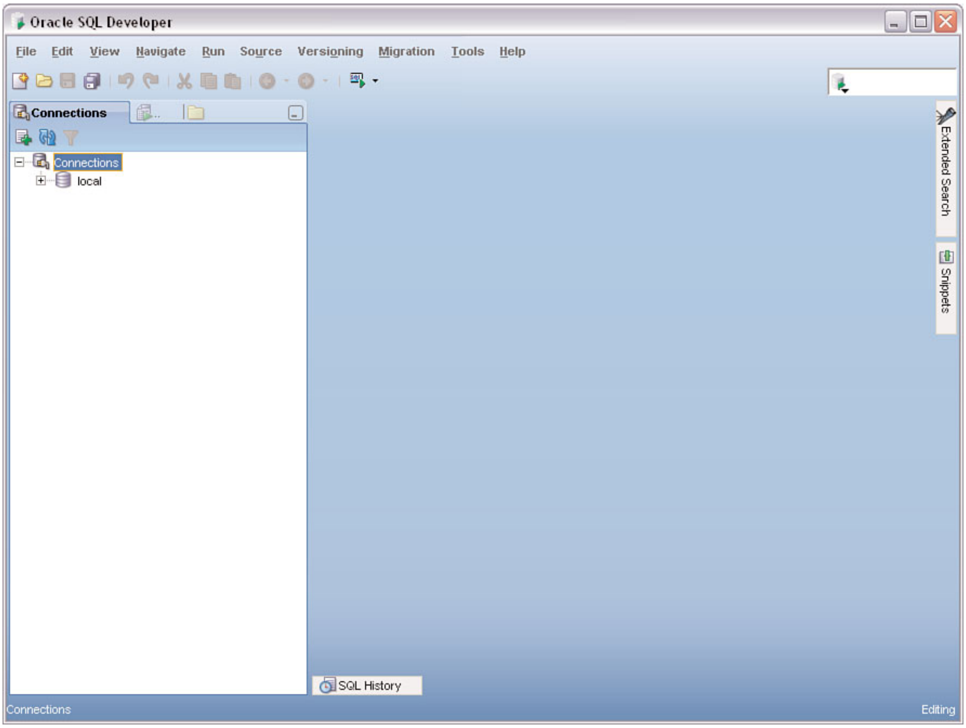


FIGURE 2.2
The Connections window in SQL Developer

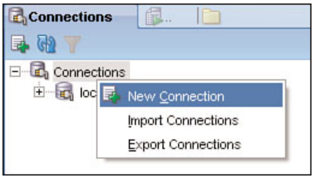


FIGURE 2.3
Creating a new database connection

You can add a new connection name, such as StudentConnection, and assign a username and password. For the purposes of the examples in this book, use the username student and the password learn.

Starting with Oracle 11*g*, the password is case-sensitive by default.

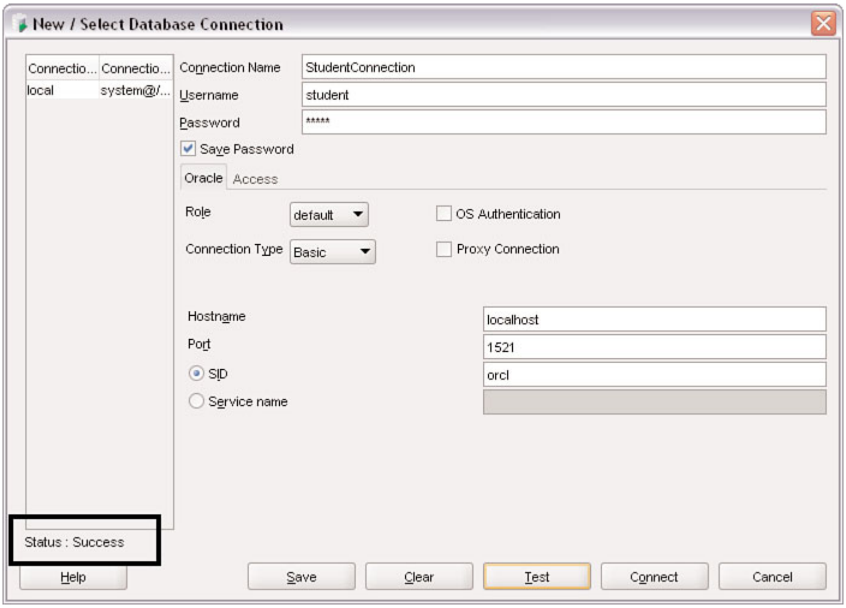
If you have not yet created the STUDENT schema (also referred to as the STUDENT user account) according to the instructions on the companion Web site, you will not be able to log in. Before you perform the lab exercises, you might want to finish reading this lab first, visit the Web site located at www.oraclesqlbyexample.com, and then create the STUDENT schema.

Choose as the Role option default. The Connection Type should be Basic, which uses the thin JDBC driver to connect; this is probably the simplest option. If you choose TNS, an entry is required in the TNSNAMES.ORA file, and the Oracle Net client must be installed. In Lab 2.3, we will discuss the TNSNAMES.ORA file as part of that lab's SQL*Plus connectivity topics.

Additional connection information consists of the name of the host (also called the *machine name*), the default port where the database will listen for connection requests (usually 1521), and either the Service name or the System ID (SID, to identify a particular database on the machine). Here the default name for the SID is orcl.

The Test button allows you determine whether this connection works. A "Success" status message appears above the Help button if the connection is successful. If your test is unsuccessful, you have probably chosen an incorrect hostname and/or SID.

The hostname is the machine name or the IP address of the machine on which the database resides. In Figure 2.4, the database is installed on the host machine called localhost. If your database resides on a computer different from the one on which you're running SQL Developer, the name of the machine on which the Oracle server is installed should be entered here.


New / Select Database Connection

Connectio...	Connectio...
local	system@/...

Connection Name: StudentConnection
Username: student
Password: *****
☑ Save Password

Oracle | Access

Role: default
Connection Type: Basic
□ OS Authentication
□ Proxy Connection

Hostname: localhost
Port: 1521
● SID: orcl
○ Service name

Status : Success

FIGURE 2.4
New / Select Database Connection dialog box

When you click the Save button, you see the connection name added to the Connections window, as shown in Figure 2.5.

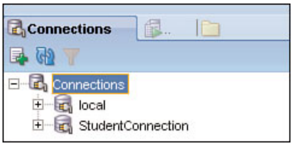


FIGURE 2.5
List of connections

When you double-click the connection name, you are connected to the database, using the appropriate user account and password. If you did not check the Save Password box when you created StudentConnection, you are prompted for it each time you open the connection.

For the majority of the exercises in this book, you will use the StudentConnection.

You can modify the connection information by right-clicking on the Student Connection node and choosing Properties from the context menu.

Exploring Database Table Objects

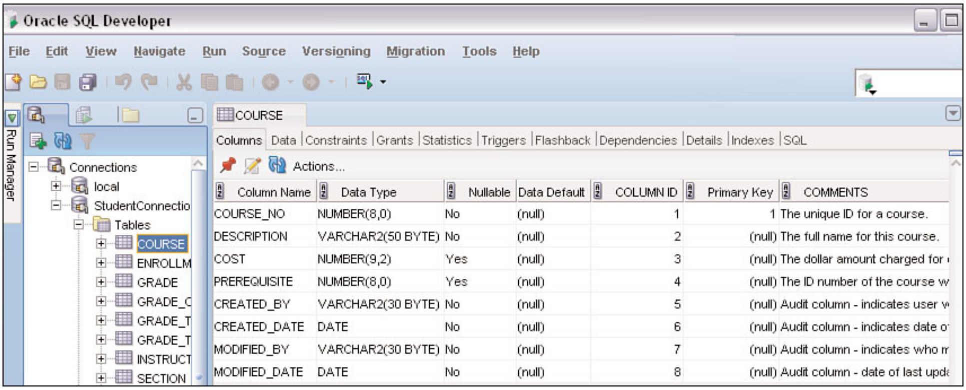
When you expand the StudentConnection node by clicking the plus sign, you see a list of database objects available to this user (see Figure 2.6). This pane, called the *Connections navigator,* is a tree-based object browser.

If you right-click on a node within the Connections navigator, a context-sensitive menu appears. For each object type, the menu varies, presenting you with choices to create, alter, drop, or manage the various objects. We will discuss the different object menus in detail in the chapters related to each object type.

For now, we will focus on the table objects. When you double-click an individual table node, you see various tabs displayed that provide details about the table.

COLUMNS TAB

The Columns tab displays a list of the columns, together with each column's data type. You can see whether the column allows null values, the primary key definition, and any column comments. In the Primary Key column, the value 1 indicates that this is the first column in the primary key. For the COURSE table, you can see in Figure 2.6 that the primary key consists of one column: COURSE_NO.


Oracle SQL Developer
File Edit View Navigate Run Source Versioning Migration Tools Help
Run Manager
Connections
local
StudentConnectio
Tables
COURSE
ENROLLM
GRADE
GRADE_C
GRADE_T
GRADE_T
INSTRUCT
SECTION
COURSE
Columns Data Constraints Grants Statistics Triggers Flashback Dependencies Details Indexes SQL
Actions...

Column Name	Data Type	Nullable	Data Default	COLUMN ID	Primary Key	COMMENTS
COURSE_NO	NUMBER(8,0)	No	(null)	1	1	The unique ID for a course.
DESCRIPTION	VARCHAR2(50 BYTE)	No	(null)	2	(null)	The full name for this course.
COST	NUMBER(9,2)	Yes	(null)	3	(null)	The dollar amount charged for
PREREQUISITE	NUMBER(8,0)	Yes	(null)	4	(null)	The ID number of the course w
CREATED_BY	VARCHAR2(30 BYTE)	No	(null)	5	(null)	Audit column - indicates user w
CREATED_DATE	DATE	No	(null)	6	(null)	Audit column - indicates date o
MODIFIED_BY	VARCHAR2(30 BYTE)	No	(null)	7	(null)	Audit column - indicates who m
MODIFIED_DATE	DATE	No	(null)	8	(null)	Audit column - date of last upd

FIGURE 2.6
Column definition of the COURSE table

DATA TAB

A click on the Data tab displays the data stored in the table. This tab also contains functionality to modify the data. You will learn how to make changes to data in Chapter 11, "Insert, Update, and Delete."

CONSTRAINTS TAB

The Constraints tab is useful for determining foreign key relationships of the table with other tables and for showing the validation constraints that exists. Chapter 12, "Create, Alter, and Drop Tables" explains these topics in detail.

GRANTS TAB

The Grants tab provides details about who has access privileges to the object; this is discussed in Chapter 15, "Security."

STATISTICS TAB

The Statistics tab shows columns and table statistics, such as the number of rows, the number of distinct values for each column, and so on. The Oracle database uses these values to optimize the execution of SQL statements. Chapter 18, "SQL Optimization," expands on how these statistics affect performance.

TRIGGERS, DEPENDENCIES, INDEXES, FLASHBACK, AND SQL TABS

The Triggers, Dependencies, Indexes, and SQL tabs are discussed in Chapter 12 as well as Chapter 13, "Indexes, Sequences, and Views." You will find out what triggers are associated with a table and the event on which the trigger fires. Any indexes that are created for the tables display in the Index tab. The Dependencies tab shows any views or objects that are dependent on this table. The Flashback tab allows you to see the previous data values before a data manipulation occurred at a specific time in the past; this will be discussed in Chapter 11. The SQL Tab shows the SQL to re-create the DDL for the table and its associated objects.

DETAILS TAB

The Details tab lists various details of a table, such as the date and time the table was created, the last date and time statistics were collected, and so on. You will learn more about this in Chapter 14, "The Data Dictionary, Scripting, and Reporting."

Reviewing the various tabs for a table allows you to glance at the important characteristics of a table. To explore another table, you double-click that table's node to replace the COURSE table's information with the new table's information. If you do not want to replace the display, you can click the red Push Pin icon to freeze the display.

The SQL Developer Worksheet

Aside from clicking the Data tab, another way to display data is by using the SQL language. The command to retrieve rows is the SELECT command. You enter SQL statements into the SQL Worksheet. The easiest way to open a worksheet is by clicking the SQL Worksheet icon in the toolbar, as shown in Figure 2.7.

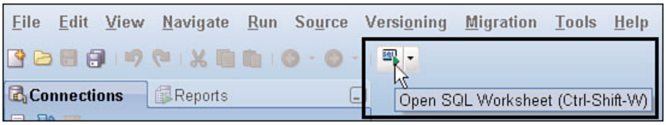


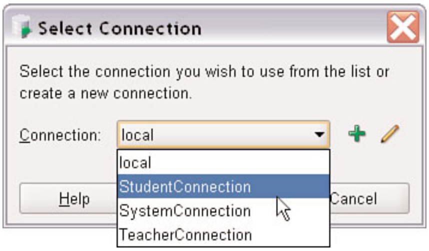
FIGURE 2.7
Open SQL Worksheet icon

Another way to open the worksheet is to right-click the connection name and choose Open SQL Worksheet, or you can choose Tools from the top menu bar and then SQL Worksheet.

The Connection dialog box (see Figure 2.8) allows you to select the database connection for this worksheet. The plus sign brings up the dialog box to create a new connection, and the Pencil icon facilitates editing of an existing database connection.

As you become more familiar with SQL Developer, you will find that there are many ways to perform the same action, using different menu choices. In addition to the menu on the top of the screen, there are context-sensitive menus and icons for frequently performed tasks.

When a connection is selected, the SQL Worksheet tab description shows the name of the connection on the top. You can execute SQL statements using the StudentConnection by entering a command in the SQL Worksheet.



Select Connection

Select the connection you wish to use from the list or create a new connection.

Connection: local

FIGURE 2.8

Select Connection dialog box

You can open multiple worksheets within SQL Developer by clicking the Open SQL Worksheet icon again. Each additional worksheet can hold different SQL statements and result sets. The worksheet tab will display the unique name of the connection on the top. For example, in Figure 2.9, a second worksheet for this connection is shown as StudentConnection~1.

The StudentConnection and StudentConnection~1 worksheets share the same database session. A *session* is an individual connection to the Oracle database server, which starts as soon as the user is logged in and authenticated. The session ends when the user disconnects or exits. Chapter 11 provides a more detailed discussion on sessions and their effect on the read consistency and locking of data during data manipulations.

Another tab, such as the TeacherConnection tab in Figure 2.9, represents another connection that may use a different database and/or login name.

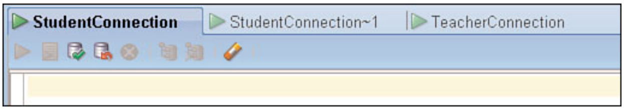


FIGURE 2.9

Multiple worksheets and their respective connections

Figure 2.10 shows the different panes within SQL Developer. You are already familiar with the Connections navigator on the left of the screen. As a separate tab next to it, you see the Reports navigator, which contains many supplied data dictionary reports, as discussed in Chapter 14.

The result of your SQL statement execution displays in the Results window, which shows the effect of the SQL statement execution. You can see a number of tabs, and SQL Developer displays most of your statement results in the Results tab. The Script Output tab shows the result of a script run (a collection of SQL statements). The Explain and the Autotrace tabs show the execution plan of a SQL statement and give an indication of how efficiently Oracle may execute your command; we discuss these tabs in Chapter 18. The DBMS Output and OWA

(Oracle Web Agent) Output tabs are relevant if you execute PL/SQL statements. (See the companion book *Oracle PL/SQL by Example,* 4th edition, by Benjamin Rosenzweig and Elena Silvestrova Rakhimov; Prentice Hall, 2008.)

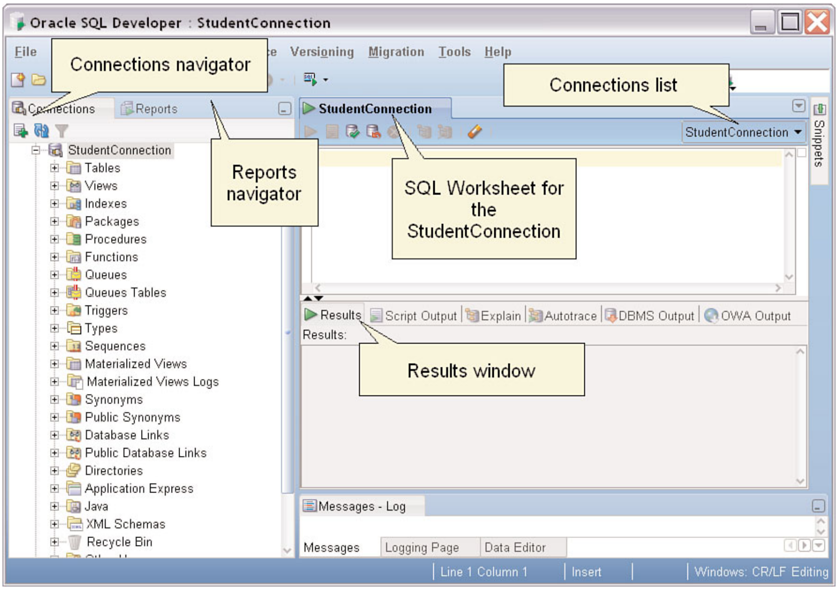


FIGURE 2.10
The various SQL Developer panes

The Connections list on the right of the SQL Worksheet allows you to switch to another connection for the current worksheet. You can execute the same statement against a different connection by choosing the connection name from the Connections list drop-down menu.

Below the Results window, you may see tabs such as Messages, Logging Page, and Data Editor. Depending on the action you are taking, you see feedback information in these tabs. You will see examples of messages in the Data Editor when you manipulate data via SQL Developer in Chapter 11.

ENTERING A SQL STATEMENT

You enter a SQL statement in the SQL Worksheet window. The following SELECT statement retrieves all the columns and rows from the COURSE table.

```
SELECT *
  FROM course
```

To execute the command, you click the green triangle. When your mouse hovers over the triangle, a ToolTip displays a description and alternative F9 function (see Figure 2.11).

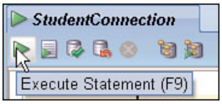
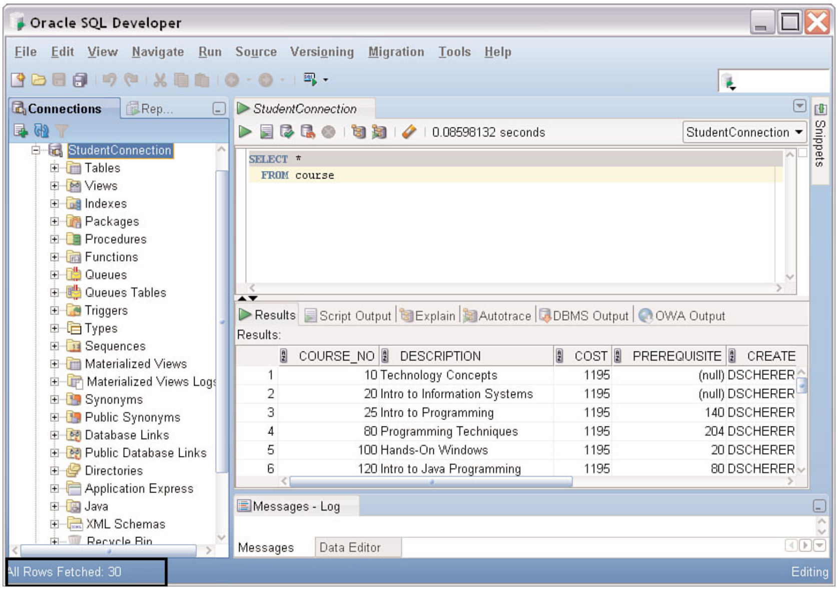


FIGURE 2.11
The Execute Statement icon

THE RESULTS TAB

The Results tab (see Figure 2.12) displays the data of the COURSE table. The left side of the Results tab shows an ordered listing of numbers, which represent the order of the rows in the Results window. These row numbers are not part of the database table; they are only for display within this window. On top of the Results tab are the column names from the COURSE table. You can scroll to the right to see any additional columns and scroll down to all the rows. You can adjust the width of individual columns and drag the column order around without having to change the SQL statement.



```
SELECT *
  FROM course
```

	COURSE_NO	DESCRIPTION	COST	PREREQUISITE	CREATE
1	10	Technology Concepts	1195	(null)	DSCHERER
2	20	Intro to Information Systems	1195	(null)	DSCHERER
3	25	Intro to Programming	1195	140	DSCHERER
4	80	Programming Techniques	1195	204	DSCHERER
5	100	Hands-On Windows	1195	20	DSCHERER
6	120	Intro to Java Programming	1195	80	DSCHERER

All Rows Fetched: 30

FIGURE 2.12
The SQL statement and corresponding result

On the bottom left of the screen, the status indicates how many records the statement returned to SQL Developer. If the bar is not visible, you can display it by choosing View, Status Bar.

Commonly Used Data Types

As you saw on the Data tab in SQL Developer, every column in Oracle has a data type, which determines what type of data can be stored. You need to know about the data types in order to use some of the comparison operators discussed in the next chapter.

DATE

The DATE data type stores date and time information. Depending on your setup, the default display format for a date may be DD-MON-YY. For example, July 4, 2009, displays as 04-JUL-09. There are a number of functions you can use to change the display format or to show the time. You also have menu options in SQL Developer for customizing the display. You will learn more about these topics in Chapter 5, "Date and Conversion Functions."

NUMBER

Columns with the data type NUMBER allow only numeric data; no text, hyphens, or dashes are permitted. A column defined as NUMBER(5,2) can have a maximum of three digits before the decimal point and two digits after the decimal point. The first digit (5) is called the *precision;* the second digit (2) is referred to as the *scale*. The smallest allowed number is – 999.99, and the largest is 999.99. A column definition with a zero scale, such as NUMBER(5) or NUMBER(5,0), allows integers in the range from – 99,999 to 99,999.

VARCHAR2 AND CHAR

The VARCHAR2 and CHAR data types store alphanumeric data (for example, text, numbers, special characters). VARCHAR2 is the variable-length data type and the most commonly used alphanumeric data type; its maximum size is 4,000 characters. The main difference between VARCHAR2 and CHAR is that the CHAR data type is a fixed-length data type, and any unused room is blank padded with spaces.

For example, a column defined as CHAR(10) and containing the four-character-length value JOHN in a row will have six blank characters padded at the end to make the total length 10 spaces. (If the column is stored in a VARCHAR2(10) column instead, it stores four characters only.) A CHAR column can store up to 2,000 characters.

If you want to store data containing more than 4,000 characters, you need to consider the CLOB data type, which allows you to store large amounts of textual data. It replaces the formerly used LONG data type, which is supported only for backward compatibility.

OTHER DATA TYPES

The data types BLOB and BFILE are binary data types that deal with access to multimedia content such as movies, images, or music. The main difference between these two data types is how the data is stored within the Oracle database. The BLOB data type stores the content inside the Oracle database, whereas the BFILE data type stores only a reference to the file location directory and the file name.

In order to access the binary content of the data, you need to use highly specific functions that go beyond the objectives of this book. In addition to the data types mentioned, Oracle provides data types to support specific national character sets (for example, NCLOB, NVARCHAR2), intermedia (image, audio, video) data types, and spatial (geographic) data. Oracle also gives you the ability to create your own customized object data types.

Refer to Appendix I, "Oracle Data Types," for a detailed list of the various data types. For most SQL operations, you typically use the NUMBER, VARCHAR2, and various DATE-related data types. They are the most commonly used data types, where the vast majority of data is stored.

Now that you know how to log on to the Oracle database, this is a good time to read the readme file you downloaded from the Web site located at www.oraclesqlbyexample.com and create the STUDENT schema if you have not already done so.

LAB 2.1 EXERCISES

a) How does the Oracle server communicate with the client?

b) In SQL Developer, expand the Tables node below the StudentConnection to reveal the different tables available to the STUDENT user. Double-click the INSTRUCTOR table. Then double-click the GRADE table. Is the information regarding the INSTRUCTOR table still visible?

c) What happens when you type DESCRIBE student in the SQL Worksheet pane and then click the Execute Statement icon?

LAB 2.1 EXERCISE **ANSWERS**

a) How does the Oracle server communicate with the client?

ANSWER: SQL Developer and SQL*Plus are examples of client programs, and the Oracle database is the server. Various protocols, such as Oracle Net and JDBC, facilitate communication between the server and the client.

The client issues SQL commands, telling the server to perform specific actions. The server sends back the results of those instructions to the client software, where they are displayed.

b) In SQL Developer, expand the Tables node below StudentConnection to reveal the different tables available to the STUDENT user. Double-click the INSTRUCTOR table. Then double-click the GRADE table. Is the information regarding the INSTRUCTOR table still visible?

ANSWER: The GRADE table information replaces the INSTRUCTOR tab. A click on the Push Pin icon (see Figure 2.13) keeps the object's information displayed.

The icon next to the Push Pin is the Edit icon. Clicking the Edit icon allows you to modify the table's column definitions, add and modify constraints, and so on. You will learn about these options in Chapter 12, which explores the different choices and their effects on the entry and storage of the data.

Next to the Edit icon is the Refresh icon, which re-queries the database for the latest updates on the given object. The Actions menu provides additional options to modify the table and column properties.

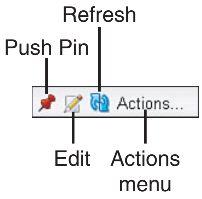
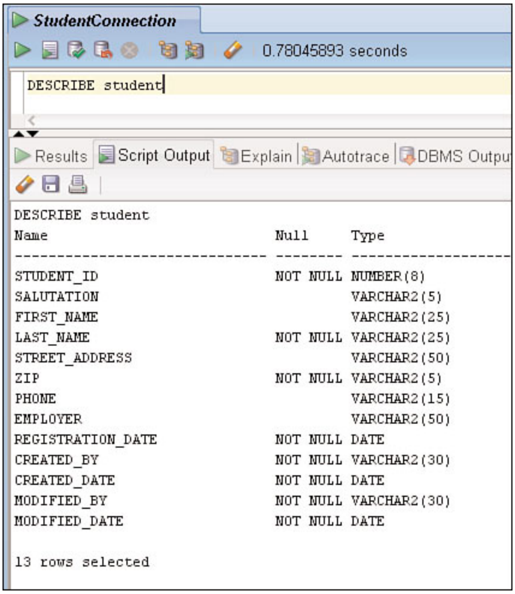


FIGURE 2.13
The Column tab icons

c) What happens when you type DESCRIBE student in the SQL Worksheet pane and then click the Execute Statement icon?

ANSWER: The DESCRIBE command displays the structure of the STUDENT table, listing the columns, data types, and null allowed characteristics. The result of the command displays in the Scripts Output tab, not the Results tab (see Figure 2.14).



```
DESCRIBE student
```

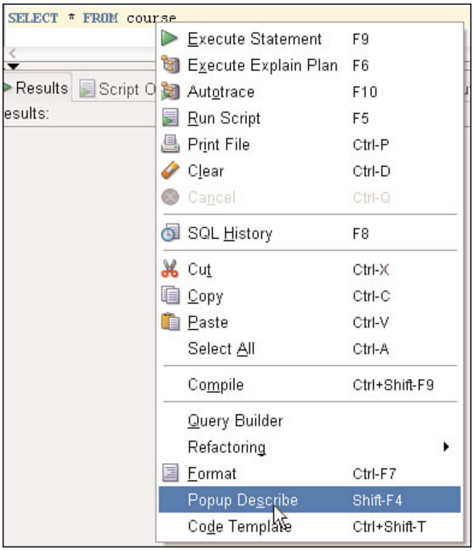
```
DESCRIBE student
Name                           Null     Type
------------------------------ -------- ------------------
STUDENT_ID                     NOT NULL NUMBER(8)
SALUTATION                              VARCHAR2(5)
FIRST_NAME                              VARCHAR2(25)
LAST_NAME                      NOT NULL VARCHAR2(25)
STREET_ADDRESS                          VARCHAR2(50)
ZIP                            NOT NULL VARCHAR2(5)
PHONE                                   VARCHAR2(15)
EMPLOYER                                VARCHAR2(50)
REGISTRATION_DATE              NOT NULL DATE
CREATED_BY                     NOT NULL VARCHAR2(30)
CREATED_DATE                   NOT NULL DATE
MODIFIED_BY                    NOT NULL VARCHAR2(30)
MODIFIED_DATE                  NOT NULL DATE

13 rows selected
```

FIGURE 2.14
The DESCRIBE command

The DESCRIBE command is actually a SQL*Plus command, not a command in the SQL language. It lets you quickly show the structure of a table. SQL Developer accepts and executes many of the SQL*Plus commands.

Because this is a SQL*Plus command, the Script Output tab, not the Results tab, shows the output. The Scripts Output tab displays the result in a similar fixed-character fashion to SQL*Plus. You also get results in this tab if you click the Run Script icon (F5); this functionality tries to emulate SQL*Plus as much as possible.

Compared to the SQL*Plus DESCRIBE command, SQL Developer's Columns tab provides significantly more detailed information at once. Another way to display the Columns tab is by using the SQL Developer's Popup Describe menu option. You access the Popup Describe menu option by placing your cursor on a table in the SQL Worksheet and then right-click for the context menu (see Figure 2.15).



```
SELECT * FROM course
```

FIGURE 2.15
The Popup Describe menu option

Lab 2.1 Quiz

In order to test your progress, you should be able to answer the following questions.

1) Anyone can connect to an Oracle database, as long as he or she has the SQL Developer or SQL*Plus software.

_______ a) True
_______ b) False

2) When you establish a connection using SQL Developer, the hostname is the machine name or IP address where the database resides.

_______ a) True
_______ b) False

3) SQL*Plus is available with every version of Oracle.

_______ a) True
_______ b) False

4) More than one user can be connected to a database at the same time.

_______ a) True
_______ b) False

5) The COST column of the COURSE table is defined as NUMBER(9,2). The maximum cost of an individual course is 9,999,999.99.

_______ a) True
_______ b) False

6) You can store at most 4,000 characters in a VARCHAR2 column.

_______ a) True
_______ b) False

ANSWERS APPEAR IN APPENDIX A.

LAB 2.2

The Anatomy of a SELECT Statement

LAB OBJECTIVES

After this lab, you will be able to:

- Write a SQL SELECT Statement
- Use DISTINCT in a SQL Statement
- Execute Statements in SQL Developer

When you write a SQL query, it is usually to find an answer to a question such as "How many students live in New York?" or "Where, and at what time, does the UNIX class meet?" A SQL *SELECT statement,* or SQL *query,* is used to find answers to these questions. A SELECT statement can be broken down into a minimum of two parts: the *SELECT list* and the *FROM clause*. The SELECT list usually consists of the column or columns of a table or tables from which you want to display data. The FROM clause states on what table or tables this column or columns are found. Later, you will learn some of the other clauses that can be used in a SELECT statement.

How to Write a SQL Query

Before formulating the SELECT statement, you must first determine in which table the information is located. A study of the schema diagram for the STUDENT database reveals that the COURSE table provides descriptions related to courses. (You can also refer to Appendix E, "Table and Column Descriptions.")

The following SELECT statement provides a list of course descriptions. SQL does not require a new line for each clause, but using this formatting convention makes for easy readability.

```
SELECT description
  FROM course
```

The SELECT list shows the single column called DESCRIPTION, which contains this information. The DESCRIPTION column is found on the COURSE table as specified in the FROM clause. When the statement is executed, the result set is a list of all the values found in the DESCRIPTION column of the COURSE table.

```
        DESCRIPTION
        -------------------------
        Technology Concepts
        Intro to Information Systems
        ...
        Java Developer III
        DB Programming with Java

30 rows selected.
```

Many of the result sets displayed throughout this book show both the SQL statement and the resulting data in a fixed-width font. At times, you may also find screenshots of the output in SQL Developer. However, typically the result is shown in a fixed-width font for easy readability.

The output of the command is displayed in bold font to easily distinguish between the output from the commands you enter. Not all the returned rows may be listed. A line in the output that shows ... indicates that some of the output has been omitted. Typically, you see the beginning and the ending rows of the result set and the number of rows returned.

RETRIEVING MULTIPLE COLUMNS

To retrieve a list of course descriptions and the cost of each course, include the COST column in the SELECT list.

```
        SELECT description, cost
          FROM course
        DESCRIPTION                   COST
        ----------------------------- ----
        Technology Concepts           1195
        Intro to Information Systems  1195
        ...
        Java Developer III            1195
        DB Programming with Java

30 rows selected.
```

When you want to display more than one column in the SELECT list, separate the columns with commas. It is good practice to include a space after the comma for readability. The order of columns in a SELECT list determines the order in which the columns are displayed in the output.

SELECTING ALL COLUMNS

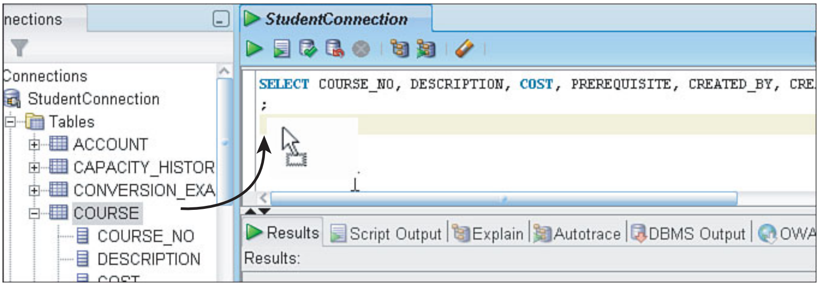
You can select all columns in a table with the asterisk (*) wildcard character. This is handy because it means you don't have to type all columns in the SELECT list. The columns are displayed in the order in which they are defined in the table. This is the same order you see when you click the Columns tab in SQL Developer or issue the DESCRIBE command.

```
SELECT *
  FROM course
```

Constructing the SQL Statement in SQL Developer

You can drag tables listed in the Connections navigator into the SQL Worksheet. When you do this, you construct a SELECT statement with all columns in the table. If desired, you can then edit the statement further. Figure 2.16 shows an example.

To define the type of statement that will be generated, select Tools, Preferences, Database: Worksheet Parameter, Drag and Drop Effect.



```
SELECT COURSE_NO, DESCRIPTION, COST, PREREQUISITE, CREATED_BY, CRE
;
```

FIGURE 2.16
Result of dragging a table into the SQL Worksheet

The SQL Worksheet Icons

Figure 2.17 shows the SQL Worksheet toolbar. You are already familiar with the Execute Statement icon.

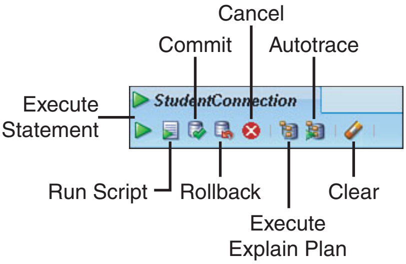


FIGURE 2.17
The SQL Worksheet icons toolbar

RUN SCRIPT

The Run Script icon allows you to execute multiple statements and emulates SQL*Plus as much as possible; the result is displayed in the Script Output tab instead of the Results tab.

COMMIT

The Commit icon looks like a database icon with the check mark. Any modifications to the data become permanent and visible to all users.

ROLLBACK

The Rollback icon looks like a database icon with an undo arrow. It undoes database changes, provided that they have not yet been committed. The COMMIT and ROLLBACK commands are discussed in Chapter 11.

CANCEL, EXECUTE EXPLAIN PLAN, AND AUTOTRACE

The Cancel icon stops a running statement that is currently executing. The Execute Explain Plan icon and the Autotrace icons are useful for optimizing SQL statements. You will learn about them in Chapter 18.

CLEAR

The eraser icon (Ctrl-D) at the end of the toolbar clears any statements in the SQL Worksheet.

Eliminating Duplicates with DISTINCT or UNIQUE

The use of the DISTINCT or UNIQUE keyword in the SELECT list eliminates duplicate data in the result set. The following SELECT statement retrieves the last name and the corresponding zip code for all rows of the INSTRUCTOR table.

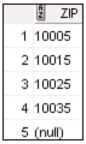
```
     SELECT last_name, zip
       FROM instructor
     LAST_NAME                 ZIP
     ------------------------- -----
     Hanks                     10015
     Wojick                    10025
     Schorin                   10025
     Pertez                    10035
     Morris                    10015
     Smythe                    10025
     Chow                      10015
     Lowry                     10025
     Frantzen                  10005
     Willig

10 rows selected.
```

There are 10 rows, yet only nine instructors have zip codes. Instructor Willig has a NULL value in the ZIP column. If you want to show only the distinct zip codes in the table, you write the following SELECT statement. In this example, the last row shows the NULL value.

```
     SELECT DISTINCT zip
       FROM instructor
     ZIP
     -----
     10005
     10015
     10025
     10035


5 rows selected.
```

By definition, a NULL is an unknown value, and a NULL does not equal another NULL. However, there are exceptions: If you write a SQL query using DISTINCT or UNIQUE, SQL considers a NULL value equal to another NULL value.

The output in SQL Developer shows the existence of the null much more obviously with a "(null)" display in the column (see Figure 2.18). Furthermore, the numbers to the left of the ZIP column display how many rows are returned.



```
  ZIP
1 10005
2 10015
3 10025
4 10035
5 (null)
```

FIGURE 2.18
Display of a null value in SQL Developer

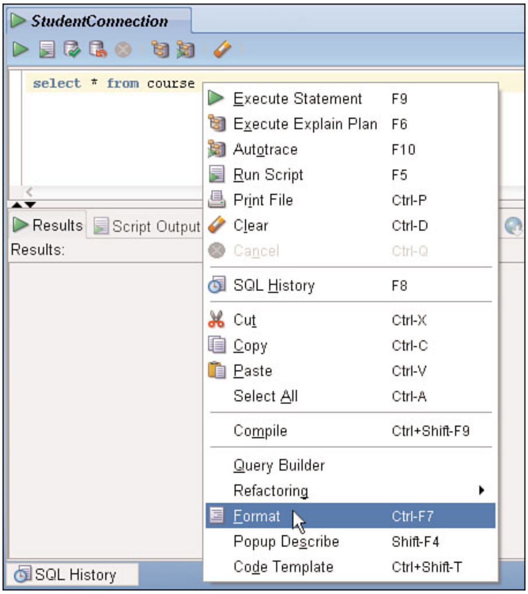
From Chapter 1, "SQL and Data," you already know that a primary key is always unique or distinct. Therefore, the use of the DISTINCT or UNIQUE keyword in a SELECT list containing the primary key column(s) is unnecessary. The ZIP column in the INSTRUCTOR table is not the primary key and can therefore contain duplicate or null values.

Formatting a SQL Statement in SQL Developer

The SQL statements presented in this and all other books in this series follow a common format. The use of uppercase for SELECT, FROM, and other Oracle keywords is for emphasis only and distinguishes them from table and column names in SQL statements, which appear in lowercase letters. A standard format enhances the clarity and readability of your SQL statements and helps you detect errors more easily. Refer to Appendix B, “SQL Formatting Guide,” for the formatting guidelines used throughout this book.

SYNTAX FORMATTING

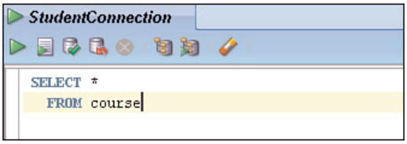
SQL Developer provides many ways to help you achieve consistency. When you right-click within the SQL Worksheet, the menu shows a Refactoring, To Upper/Lower/Initcap menu option that lets you toggle between the different cases. The shortcut to remember is Ctrl-Quote. Another useful feature is the Format menu (Ctrl-F7); it automatically reformats your SQL statement to fit a given standard. You highlight the statement, right-click, and choose Format (see Figure 2.19) from the context menu.



```
select * from course
```

FIGURE 2.19
Format feature

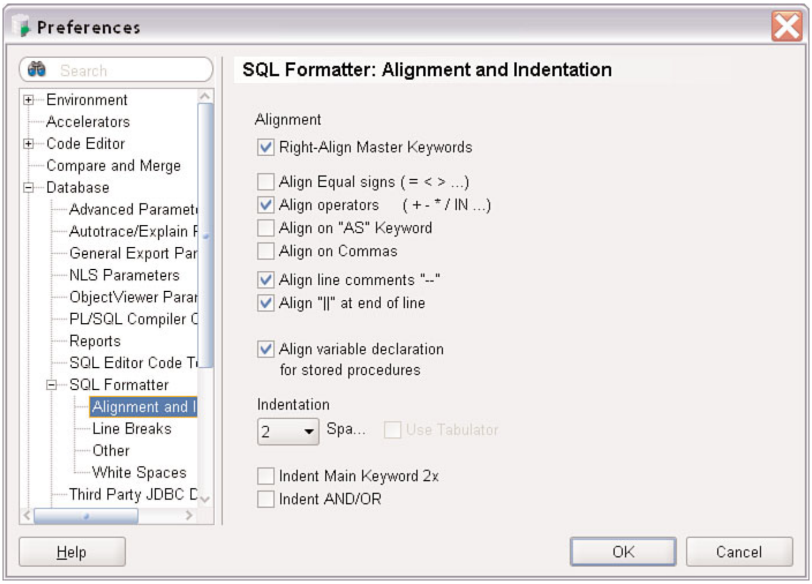
Figure 2.20 shows the result of this selection. The Oracle keywords are in uppercase and right aligned, and the name of the COURSE table is in lowercase.



```
SELECT *
  FROM course
```

FIGURE 2.20
Format results

The Tools, Preference, SQL Formatter menu option allows you to customize the formatting to your standards (see Figure 2.21).



SQL Formatter: Alignment and Indentation

Alignment

☑ Right-Align Master Keywords

□ Align Equal signs (= < > ...)

☑ Align operators (+ - * / IN ...)

□ Align on "AS" Keyword

□ Align on Commas

☑ Align line comments "--"

☑ Align "||" at end of line

☑ Align variable declaration for stored procedures

Indentation

2 Spa... □ Use Tabulator

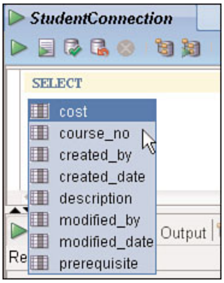
□ Indent Main Keyword 2x

□ Indent AND/OR

FIGURE 2.21
Preferences window

CODE COMPLETION

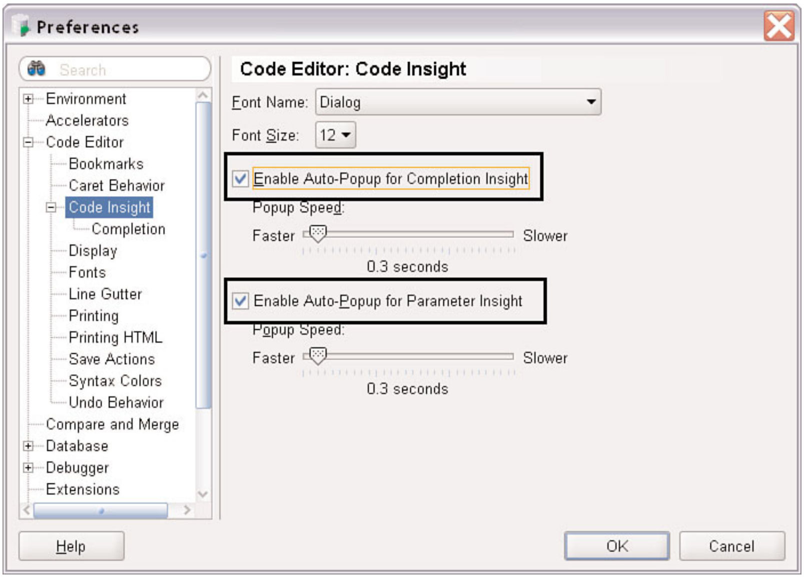
Another useful feature of Oracle Developer is code completion, which helps you complete your SQL statements easily. When you pause on your statement, the program prompts you for appropriate commands, column names, or table names, which you can then select from the list. Figure 2.22 shows an example. When you remove the asterisk from the statement and enter a space, you see a list of possible choices. You can then choose the DESCRIPTION column and then enter a comma to get the list of relevant columns.



```
SELECT
cost
course_no
created_by
created_date
description
modified_by
modified_date
prerequisite
```

FIGURE 2.22
Code completion feature in SQL Developer

If you find the code completion feature confusing, you can turn it off by unchecking both of the Enable Auto-Popup check boxes in the Tools, Preference menu (see Figure 2.23).



Code Editor: Code Insight

Font Name: Dialog

Font Size: 12

☑ Enable Auto-Popup for Completion Insight

Popup Speed: Faster — Slower, 0.3 seconds

☑ Enable Auto-Popup for Parameter Insight

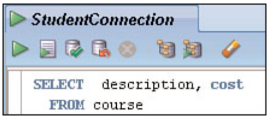
Popup Speed: Faster — Slower, 0.3 seconds

FIGURE 2.23
Code Insight preferences

SYNTAX HIGHLIGHTING

SQL Developer offers syntax highlighting, which helps distinguish the SQL language keywords with a different color. This way, you can easily identify and distinguish between the SQL language commands and any table or column names. The column and table names appear in black; SQL language commands appear in blue. This color-coding improves the readability of a statement and helps you spot syntax errors easily.

Notice that the COST column in Figure 2.24 is not colored black. Even though this is the name of the column in the table, COST also happens to be an Oracle keyword.

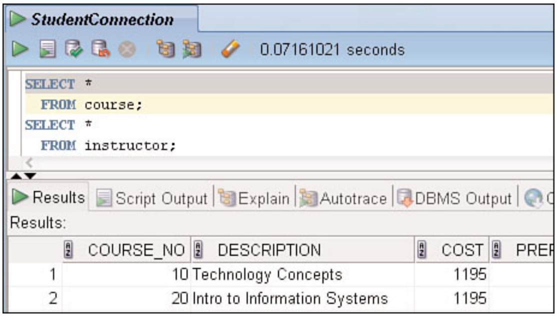


```
SELECT  description, cost
  FROM course
```

FIGURE 2.24
Syntax highlighting

Writing Multiple Statements in the SQL Worksheet

You can enter multiple statements in the SQL Worksheet and execute them individually by placing the cursor on the line of the statement (see Figure 2.25). You need to end each SQL statement with a semicolon (;) or type a forward slash (/) on a new line; otherwise, SQL Developer displays an error.



```
SELECT *
  FROM course;
SELECT *
  FROM instructor;
```

Results:

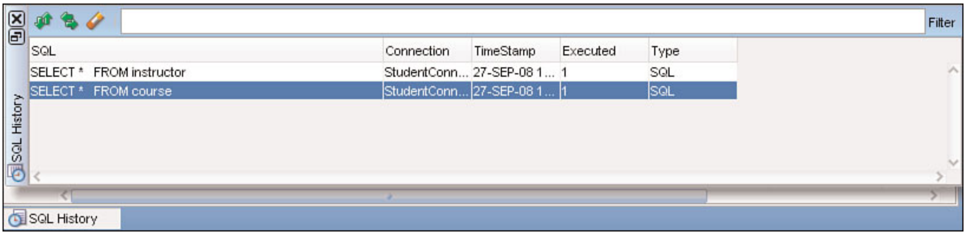
	COURSE_NO	DESCRIPTION	COST	PREF
1	10	Technology Concepts	1195	
2	20	Intro to Information Systems	1195	

FIGURE 2.25
Executing multiple SQL statements in SQL Developer

If you want to run both statements at once, you need to run the statements as a script by clicking the Run Script icon (F5). The output is then displayed in the Script Output tab in a matter much like the SQL*Plus command-line version.

SQL Developer's Statement History

SQL Developer keeps track of your most recently executed commands in the SQL History window (see Figure 2.26) below the Results pane. If the SQL History tab is not visible, you can click View, SQL History or press F8. The SQL commands are saved even after you exit SQL Developer.



SQL	Connection	TimeStamp	Executed	Type
SELECT * FROM instructor	StudentConn...	27-SEP-08 1...	1	SQL
SELECT * FROM course	StudentConn...	27-SEP-08 1...	1	SQL

FIGURE 2.26
SQL History window

To place a command from the History window back into the SQL Worksheet, you can simply double-click the statement. If you choose the up/down arrows icon on the left, the statement is appended to any existing statements in the SQL Worksheet window. The left/right arrows icon replaces any existing SQL statement(s) in the Worksheet.

You are able to search for text within the historical SQL statements by entering the information in the box and clicking the Filter button on the right. The eraser icon clears all the statements from the SQL history. If you do not choose a statement, you can exit the SQL History window by pressing the Esc key.

▼ LAB 2.2 EXERCISES

a) Write a SELECT statement that lists the first and last names of all students.

b) Write a SELECT statement that lists all cities, states, and zip codes.

c) Why are the results of the following two SQL statements the same?

```
SELECT letter_grade
  FROM grade_conversion

SELECT UNIQUE letter_grade
  FROM grade_conversion
```

d) Explain what happens, and why, when you execute the following SQL statement.

```
SELECT DISTINCT course_no
  FROM class
```

e) Execute the following SQL statement. Then, in the Results window, right-click and choose the menu option Single Record View. Describe your observation.

```
SELECT *
  FROM student
```

▼ LAB 2.2 EXERCISE ANSWERS

a) Write a SELECT statement that lists the first and last names of all students.

ANSWER: The SELECT list contains the two columns that provide the first and last names of students; the FROM clause lists the STUDENT table where these columns are found. You can examine the rows by scrolling up and down. The rows are not returned in any particular order; you will learn about ordering the result set in Chapter 3, "The WHERE and ORDER BY Clauses."

```
SELECT first_name, last_name
  FROM student
FIRST_NAME                LAST_NAME
------------------------- ----------
George                    Eakheit
Leonard                   Millstein
...
```

```
Kathleen                  Mastandora
Angela                    Torres

268 rows selected.
```

b) Write a SELECT statement that list all cities, states, and zip codes.

ANSWER: The SELECT list contains the three columns that provide the city, state, and zip code; the FROM clause contains the ZIPCODE table where these columns are found.

```
SELECT city, state, zip
  FROM zipcode
CITY                      ST ZIP
------------------------- --------
Santurce                  PR 00914
North Adams               MA 01247
...
New York                  NY 10005
New York                  NY 10035

227 rows selected.
```

c) Why are the results of the following two SQL statements the same?

```
SELECT letter_grade
  FROM grade_conversion

SELECT UNIQUE letter_grade
  FROM grade_conversion
```

ANSWER: The result sets are the same because the data values in the LETTER_GRADE column of the GRADE_CONVERSION table are not repeated; the LETTER_GRADE column is the primary key of the table, so by definition its values are unique. The UNIQUE and DISTINCT keywords can be used interchangeably.

d) Explain what happens, and why, when you execute the following SQL statement.

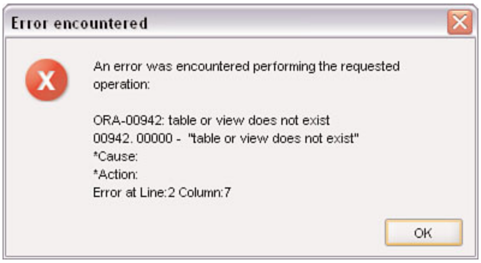
```
SELECT DISTINCT course_no
  FROM class
```

ANSWER: Oracle returns an error because a table named CLASS does not exist.

The error message indicates the error in the query. In SQL Developer, you see a message box similar to Figure 2.27, which indicates the line and column number where the error occurs.

You can review your cursor's exact position by referring to the bottom of the screen (see Figure 2.28).

SQL is an exacting language. As you learn to write SQL, you will inevitably make mistakes. It is important to pay attention to the error messages the database returns to you so you can learn from and correct your mistakes. For example, the Oracle error message in Figure 2.27 informs you that you referenced a nonexistent table or view within the database schema. (Views are discussed in Chapter 13. You can correct your SQL statement and execute it again.



An error was encountered performing the requested operation:

ORA-00942: table or view does not exist
00942. 00000 - "table or view does not exist"
*Cause:
*Action:
Error at Line:2 Column:7

FIGURE 2.27
Error message in SQL Developer

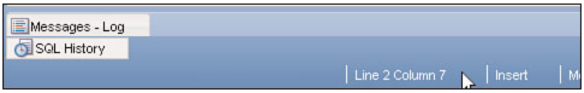
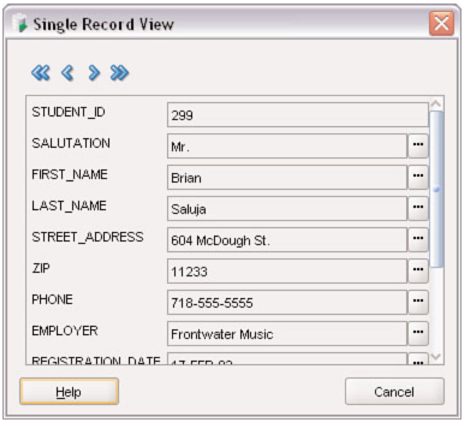


FIGURE 2.28
Line and column indicator

e) Execute the following SQL statement. Then, in the Results window, right-click and choose the menu option Single Record View. Describe your observation.

```
SELECT *
  FROM student
```

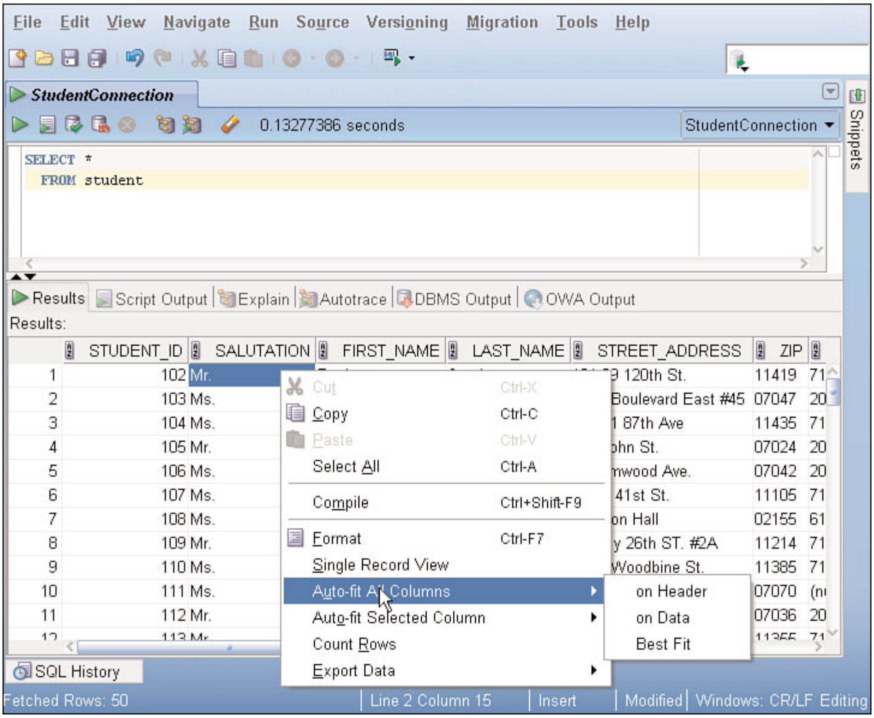
ANSWER: The Single Record View window allows you to examine one record at a time and scroll through the records using the arrows at the top (see Figure 2.29). If there are many columns in a table, you can expand the window by dragging its sides.



STUDENT_ID 299
SALUTATION Mr.
FIRST_NAME Brian
LAST_NAME Saluja
STREET_ADDRESS 604 McDough St.
ZIP 11233
PHONE 718-555-5555
EMPLOYER Frontwater Music
REGISTRATION_DATE 17-FEB-03

FIGURE 2.29
Single Record View window

As you can see, there are many menu options available when you right-click the Results window. The Auto-fit menu options (see Figure 2.30) are very useful for formatting the Results window according to the length of the data cells or the length of the column name.



0.13277386 seconds

```
SELECT *
  FROM student
```

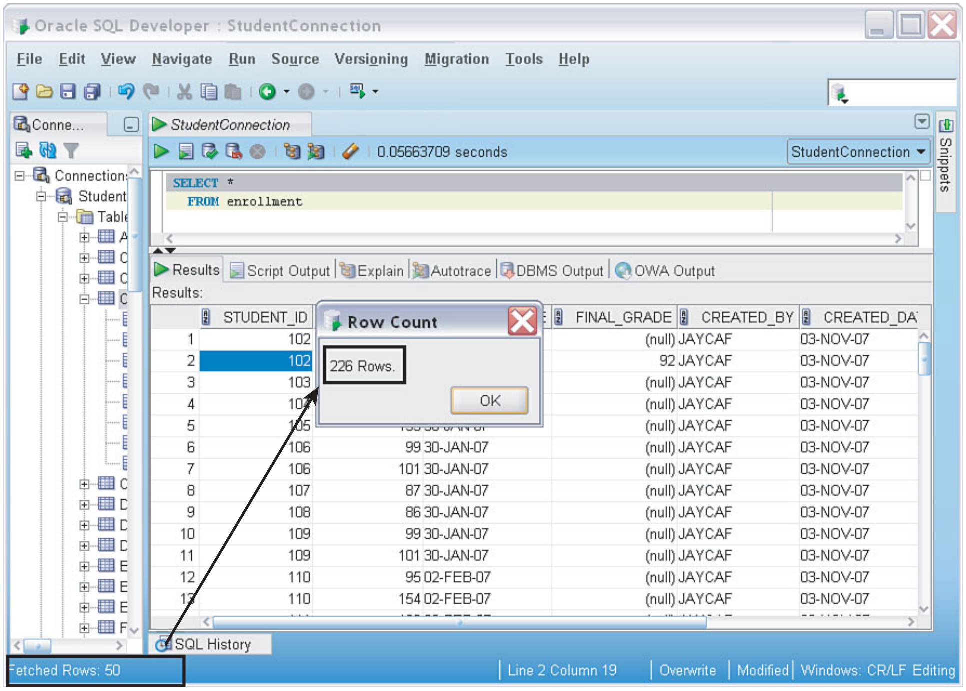
Results:

	STUDENT_ID	SALUTATION	FIRST_NAME	LAST_NAME	STREET_ADDRESS	ZIP	
1	102	Mr.			9 120th St.	11419	71
2	103	Ms.			Boulevard East #45	07047	20
3	104	Ms.			1 87th Ave	11435	71
4	105	Mr.			ohn St.	07024	20
5	106	Ms.			nwood Ave.	07042	20
6	107	Ms.			41st St.	11105	71
7	108	Ms.			on Hall	02155	61
8	109	Mr.			y 26th ST. #2A	11214	71
9	110	Ms.			Woodbine St.	11385	71
10	111	Ms.				07070	(nu
11	112	Mr.				07036	20
12	113	Mr.				11355	71

Fetched Rows: 50 | Line 2 Column 15 | Insert | Modified | Windows: CR/LF Editing

FIGURE 2.30
The Results window menu options

The Count Rows menu option returns the number of rows in the table. Not all the rows may be displayed at a given time in SQL Developer, as indicated in the Fetched Rows message on the status bar (see Figure 2.31). SQL Developer fetches additional rows, as needed, when you scroll down.



```
SELECT *
  FROM enrollment
```

Results:

	STUDENT_ID			FINAL_GRADE	CREATED_BY	CREATED_DA
1	102			(null)	JAYCAF	03-NOV-07
2	102			92	JAYCAF	03-NOV-07
3	103			(null)	JAYCAF	03-NOV-07
4	104			(null)	JAYCAF	03-NOV-07
5	105	[illegible]	[illegible]	(null)	JAYCAF	03-NOV-07
6	106	99	30-JAN-07	(null)	JAYCAF	03-NOV-07
7	106	101	30-JAN-07	(null)	JAYCAF	03-NOV-07
8	107	87	30-JAN-07	(null)	JAYCAF	03-NOV-07
9	108	86	30-JAN-07	(null)	JAYCAF	03-NOV-07
10	109	99	30-JAN-07	(null)	JAYCAF	03-NOV-07
11	109	101	30-JAN-07	(null)	JAYCAF	03-NOV-07
12	110	95	02-FEB-07	(null)	JAYCAF	03-NOV-07
13	110	154	02-FEB-07	(null)	JAYCAF	03-NOV-07

226 Rows.

Fetched Rows: 50

FIGURE 2.31
Row Count and Fetched Rows comparison

Lab 2.2 Quiz

In order to test your progress, you should be able to answer the following questions.

1) The SELECT clause specifies the columns you want to display, and the FROM clause specifies the table that contains these columns.

_______ a) True
_______ b) False

2) The column names listed in the SELECT list must be separated by commas.

_______ a) True
_______ b) False

3) The asterisk can be used as a wildcard in the FROM clause.

_______ a) True
_______ b) False

4) The following statement contains an error:

```
SELECT courseno
  FROM course
```

_______ a) True
_______ b) False

5) The Cancel icon stops a long-running SQL statement.

_______ a) True
_______ b) False

6) The Ctrl-Quote keystroke allows you to toggle the case of text entered in the SQL Worksheet.

_______ a) True
_______ b) False

7) Syntax highlighting in SQL Developer helps you distinguish between Oracle keywords and table/column names.

_______ a) True
_______ b) False

8) All SQL commands must be entered in uppercase only.

_______ a) True
_______ b) False

9) When you click on the Execute Statement icon, the output always displays in the Results tab.

_______ a) True
_______ b) False

ANSWERS APPEAR IN APPENDIX A.

LAB 2.3
An Introduction to SQL*Plus

LAB OBJECTIVES

After this lab, you will be able to:

- Understand the Essentials of SQL*Plus
- Execute Commands in SQL*Plus
- Name the Major Differences between SQL Developer and SQL*Plus

All Oracle databases include an installation of SQL*Plus by default. SQL*Plus is an Oracle software tool that allows you to execute SQL statements and SQL*Plus commands. It has been around since Oracle's early beginnings, and this command-line interface is available with every Oracle version and operating system. You can clearly see the age of SQL*Plus in its outdated interface, but this tool still serves many useful purposes.

Why Learn About SQL*Plus?

You might wonder what is the rationale of learning to use the command-line SQL*Plus when SQL Developer's graphical user interface is so much more intuitive. All the SQL statements and many SQL*Plus-specific commands work the same way in SQL Developer. Unquestionably, SQL*Plus seems quite arcane compared to SQL Developer, but knowing this old-style tool may come in handy when you're working with Oracle versions that do not support SQL Developer (such as versions prior to 9.2.0.1).

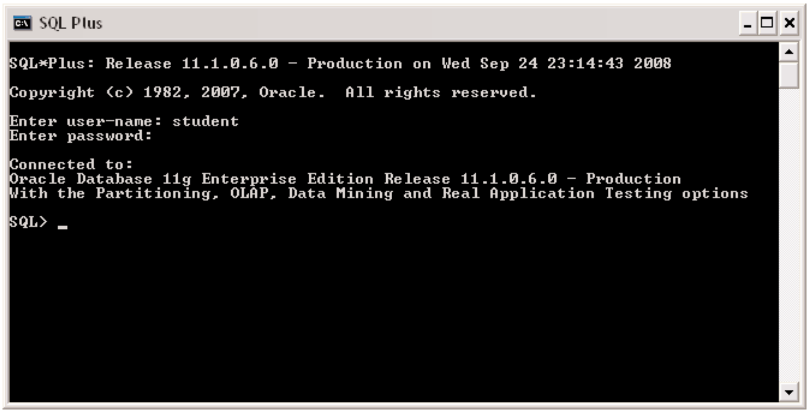
Furthermore, SQL*Plus is very suitable for executing scripts from the operating system prompt. A script is a saved file that contains one or more statements that allows you to rerun a command without retyping. This is useful when you need to rerun the same statements. You will learn about this in Chapter 14.

Starting SQL*Plus

If SQL*Plus program is installed on your Windows machine, you can access it by choosing Programs, Oracle, Application Development, SQL*Plus. This launches the program and displays the Log On dialog. Enter student as the username and learn as the password (both in lowercase) and press the Enter key. The password does not display onscreen.

Figure 2.32 illustrates a successful login with the correct username and password. Effectively, you have established a connection with the Oracle database as the user STUDENT. The client and the server can now communicate with each other.

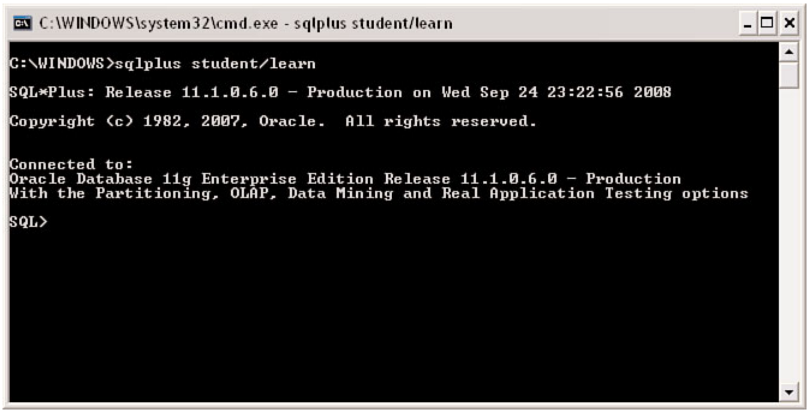
The screen shows the version of SQL*Plus and the Oracle database. The SQL> command prompt indicates that SQL*Plus is ready to accept your commands, and you can begin to type. This is the default prompt for SQL*Plus.



```
SQL*Plus: Release 11.1.0.6.0 - Production on Wed Sep 24 23:14:43 2008

Copyright (c) 1982, 2007, Oracle.  All rights reserved.

Enter user-name: student
Enter password:

Connected to:
Oracle Database 11g Enterprise Edition Release 11.1.0.6.0 - Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL> _
```

FIGURE 2.32
The SQL*Plus prompt

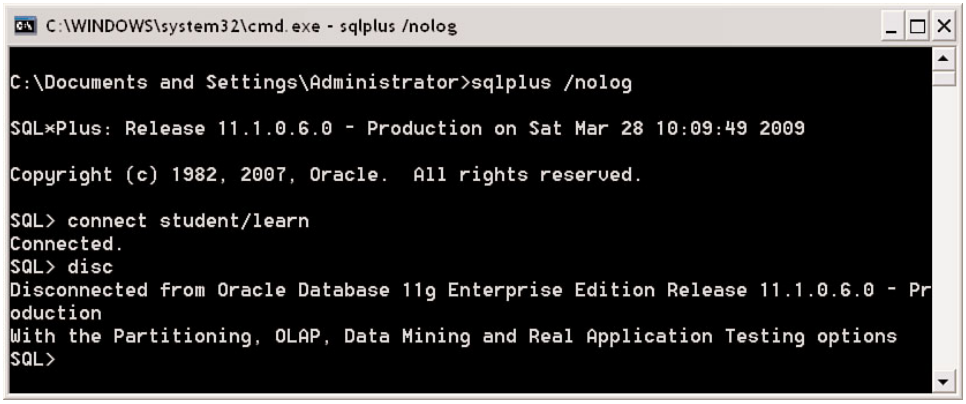
You can also invoke SQL*Plus by typing sqlplus at your operating system's command prompt and entering the username and password when prompted to do so. Or you can include the login username and password directly on the operating system prompt, as shown in Figure 2.33.



```
C:\WINDOWS>sqlplus student/learn

SQL*Plus: Release 11.1.0.6.0 - Production on Wed Sep 24 23:22:56 2008

Copyright (c) 1982, 2007, Oracle.  All rights reserved.


Connected to:
Oracle Database 11g Enterprise Edition Release 11.1.0.6.0 - Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL>
```

FIGURE 2.33
Invoking SQL*Plus from the Windows operating system command prompt

You can also invoke SQL*Plus without logging in to the database by using the NOLOG option (see Figure 2.34). To connect to the database, you use the CONNECT command. The DISCONNECT (or DISC) command disconnects the session but does not exit SQL*Plus. Issuing a CONNECT command disconnects you from any previously connected session.



```
C:\Documents and Settings\Administrator>sqlplus /nolog

SQL*Plus: Release 11.1.0.6.0 - Production on Sat Mar 28 10:09:49 2009

Copyright (c) 1982, 2007, Oracle.  All rights reserved.

SQL> connect student/learn
Connected.
SQL> disc
Disconnected from Oracle Database 11g Enterprise Edition Release 11.1.0.6.0 - Pr
oduction
With the Partitioning, OLAP, Data Mining and Real Application Testing options
SQL>
```

FIGURE 2.34
The NOLOG option and the CONNECT and DISCONNECT commands

Exiting SQL*Plus

To log out of SQL*Plus, either type EXIT or QUIT and press Enter. Alternatively, you can simply use your mouse to close the window. In the Windows operating system, you can also press Ctrl+C or Ctrl+Z, and in UNIX you can use Ctrl+D.

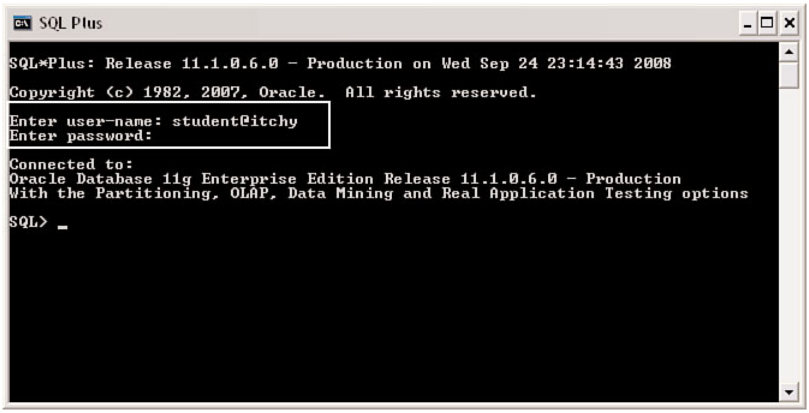
Exiting ends the session, and the STUDENT user is no longer connected to the database. However, there may be other client machines connected to the Oracle database; the server software continues to run, regardless of whether a client is connected to it.

The Remote Database and SQL*Plus

Often, a database resides on a machine other than your local client machine, or you have a choice of accessing different databases. In these cases, you need to supply a *connect identifier*, which directs SQL*Plus to the appropriate database.

Furthermore, you need to have Oracle's connectivity software, called SQL Net, installed. Typically when you perform a SQL*Plus installation, the SQL Net software is automatically installed for you. This is different from SQL Developer, which works with both a JDBC connection and SQL Net.

To use SQL*Plus to connect to a remote database called ITCHY, you enter the username, followed by the @ symbol followed by the connect identifier. Figure 2.35 shows such a logon example.



```
SQL*Plus: Release 11.1.0.6.0 - Production on Wed Sep 24 23:14:43 2008

Copyright (c) 1982, 2007, Oracle.  All rights reserved.

Enter user-name: student@itchy
Enter password:

Connected to:
Oracle Database 11g Enterprise Edition Release 11.1.0.6.0 - Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL> _
```

FIGURE 2.35
Using a connect identifier in SQL*Plus

The simplified syntax for the logon is as follows. The first syntax line prompts you for the password.

```
username@connect_identifier
username/password@connect_identifier
```

The connect identifier matches either an entry in a file called TNSNAMES.ORA or follows the Easy Connect syntax.

USING A TNSNAMES ENTRY

Essentially, the TNSNAMES.ORA file is a file that contains a list of databases with their respective technical connection information. It lists the database's IP address (or the machine name) and database instance name. Your database administrator can help you with the configuration and setup of this file if you have a remote database setup.

Following is an excerpt of a TNSNAMES.ORA file. The entries in your file will obviously vary. If you supply the host string ITCHY at login, SQL*Plus looks up the ITCHY entry in the TNSNAMES.ORA file. The HOST entry shows the machine name or IP address. The service name, or SID, entry identifies the name of the Oracle instance; here the instance is called ORCL. When you install Oracle with the default options, you are asked to supply an SID (system identifier). A common default name is ORCL.

The terms *SID* and *service name* are often used interchangeably, but they can be different, particularly in environments running Oracle RAC (Real Application Cluster) for fault-tolerant replication of data.

```
ITCHY =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS =
        (PROTOCOL = TCP)
        (HOST =ibmt41)
        (PORT = 1521)
      )
    )
    (CONNECT_DATA =
      (SERVICE_NAME = ORCL)
    )
  )
SCRATCHY =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS =
        (PROTOCOL = TCP)
        (HOST = 169.254.147.245)
        (PORT = 1521)
      )
    )
    (CONNECT_DATA =
      (SID = ORCL)
    )
  )
```

USING EASY CONNECT SYNTAX

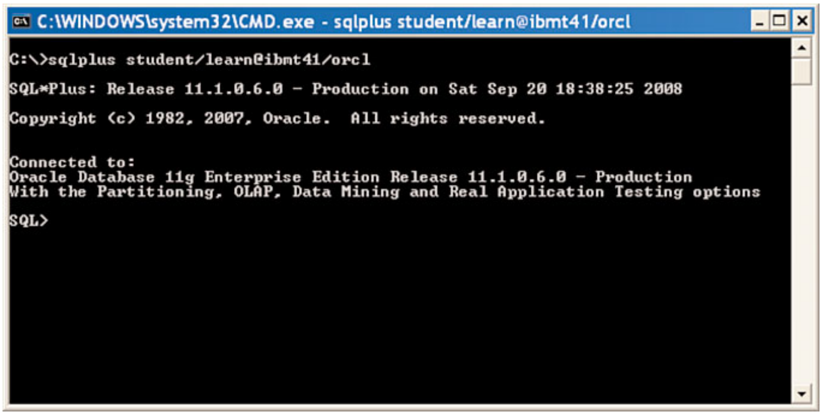
The Easy Connect feature allows you to make a connection without the ITCHY entry being present in the TNSNAMES.ORA file. For example, you can connect to this database by supplying all the connection information.

```
student/learn@ibmt41:1521/ORCL
```

Or you can use the following.

```
student/learn@ibmt41/ORCL
```

This syntax shows the machine name called ibmt41 followed by the port number (the default port of the Oracle database is typically 1521), followed by the SID ORCL. Figure 2.36 shows how this connection is established in SQL*Plus.


C:\WINDOWS\system32\CMD.exe - sqlplus student/learn@ibmt41/orcl

```
C:\>sqlplus student/learn@ibmt41/orcl

SQL*Plus: Release 11.1.0.6.0 - Production on Sat Sep 20 18:38:25 2008

Copyright (c) 1982, 2007, Oracle.  All rights reserved.


Connected to:
Oracle Database 11g Enterprise Edition Release 11.1.0.6.0 - Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL>
```

FIGURE 2.36
Connection to SQL*Plus using the Easy Connect syntax

The syntax for the connect identifier using Easy Connect follows.

```
Host[:Port]/service_name
```

The host is the machine or IP address of the database server computer. The port specifies the listening port of the database server; if it is not specified, it defaults to 1521. The service name is the name of the database instance you want to connect to on this database server.

Generally, you create a TNSNAMES entry when you use the same connection frequently; it's far quicker to enter than a long Easy Connect string. Your organization may even have a dedicated Oracle Names Server that manages the connectivity of many servers without the need to maintain a TNSNAMES entry on client machines.

Logon Problems

Although this book cannot possibly list all the errors and solutions to all logon problems, let's look at are two very common Oracle error messages you may encounter.

TNS ERROR

A TNS error is usually related to the connectivity between the server and the client, using the Oracle Net client software. The following message is displayed if the connect identifier could not be resolved. This may be due to an invalid hostname or service name. You need to check the values and retry.

```
ORA-12154: TNS: could not resolve the connect identifier specified
```

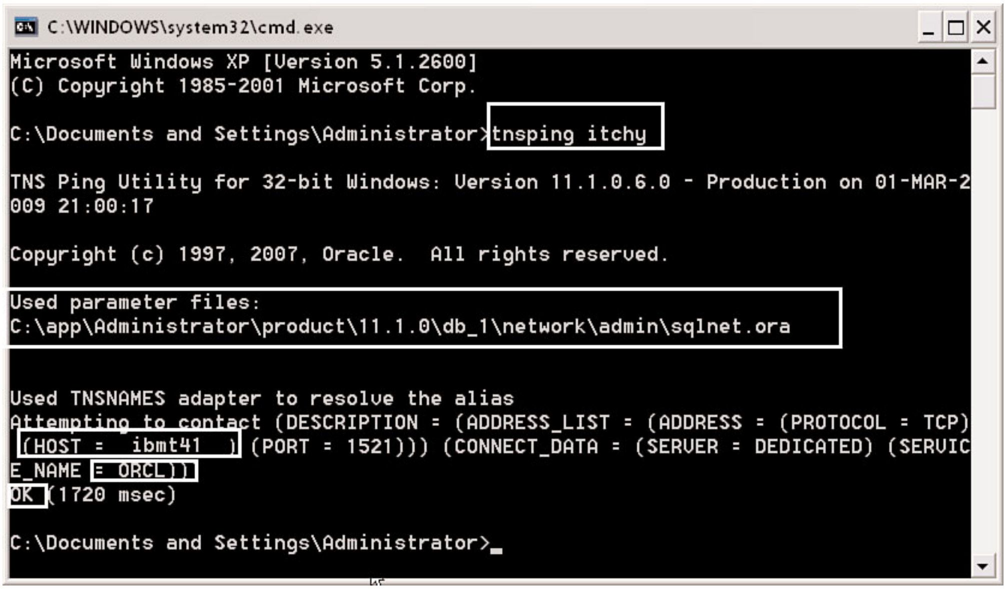
INCORRECT USERNAME OR PASSWORD

Another error occurs if you entered the wrong username or password when the Oracle server attempted to authenticate you as a valid user. You need to double-check the spelling of your username, which is student, and your password, which is learn (both in lowercase). Starting with Oracle 11g, the password is case-sensitive by default. (If you cannot log on with this ID and password, check the readme file regarding the installation of the STUDENT schema.)

```
ORA-01017: invalid username/password; logon denied
```

If you are connecting to a remote Oracle database, be sure to enter the Oracle Net connection string supplied to you by your Oracle database administrator and recorded in your TNSNAMES.ORA file.

If you want to test whether the TNSNAMES entry is resolved correctly, you can ping the database with the TNSPING command from the operating system prompt. Figure 2.37 shows the execution and result of the command to determine whether the TNSNAMES entry ITCHY is valid and whether the server's listener program is running. From the output, you can see the file location of the TNSNAMES.ORA that was used to resolve the ITCHY name. Furthermore, you can see the host or machine name value and the service or instance name. The OK message indicates that the database's listener process is ready to accept your connection request.



```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Administrator>tnsping itchy

TNS Ping Utility for 32-bit Windows: Version 11.1.0.6.0 - Production on 01-MAR-2
009 21:00:17

Copyright (c) 1997, 2007, Oracle.  All rights reserved.

Used parameter files:
C:\app\Administrator\product\11.1.0\db_1\network\admin\sqlnet.ora


Used TNSNAMES adapter to resolve the alias
Attempting to contact (DESCRIPTION = (ADDRESS_LIST = (ADDRESS = (PROTOCOL = TCP)
(HOST =  ibmt41  ) (PORT = 1521))) (CONNECT_DATA = (SERVER = DEDICATED) (SERVIC
E_NAME = ORCL)))
OK (1720 msec)

C:\Documents and Settings\Administrator>_
```

FIGURE 2.37
TNSPING command result

Executing SQL Commands Using SQL*Plus

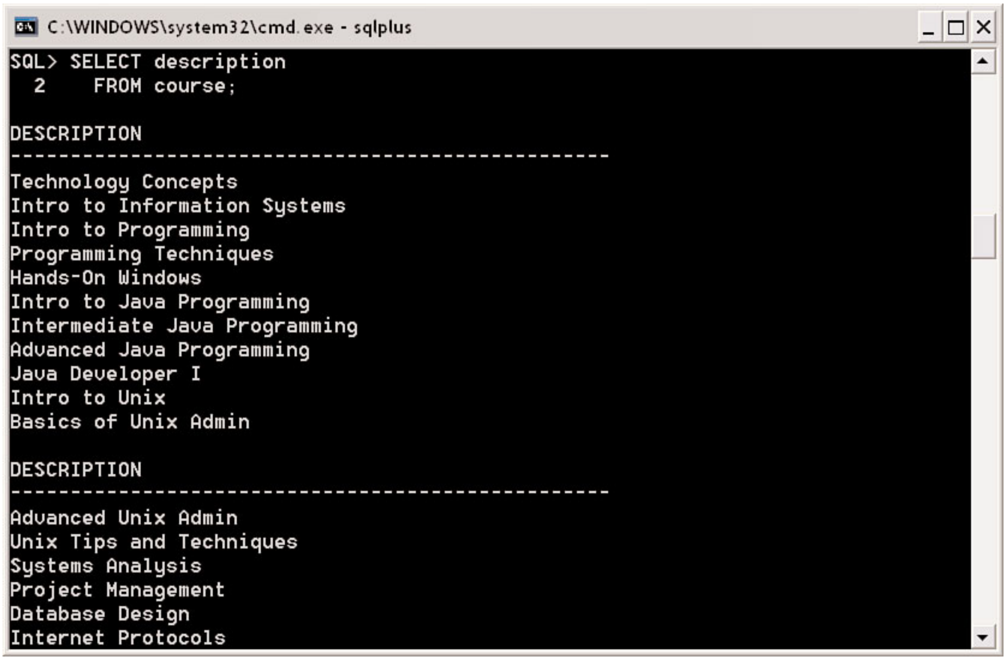
SQL*Plus requires the use of a semicolon (;) at the end of each SQL statement to execute the statement. Alternatively, the forward slash (/) can be used on a separate line to accomplish the same thing. In the following statement, we want to show only the DESCRIPTION column of the COURSE table.

```
SQL> SELECT description
  2    FROM course;
```

Another way to do the same thing is to use the following statement.

```
SQL> SELECT description
  2    FROM course
  3  /
```

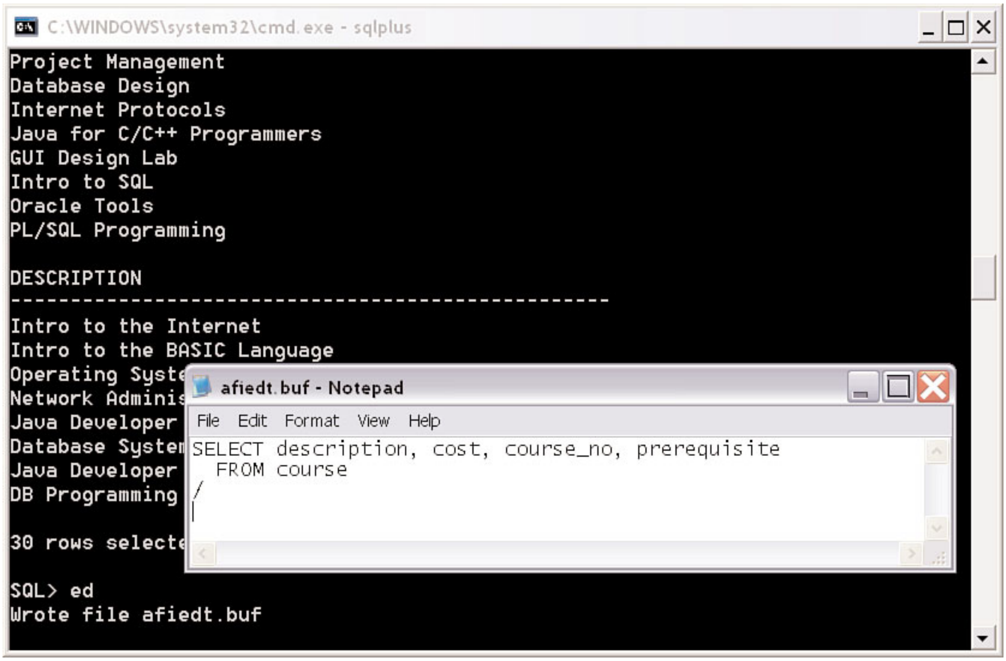
Figure 2.38 shows the result of the execution of this query in SQL*Plus. You can scroll up and down to see the results.



```
SQL> SELECT description
  2    FROM course;

DESCRIPTION
--------------------------------------------------
Technology Concepts
Intro to Information Systems
Intro to Programming
Programming Techniques
Hands-On Windows
Intro to Java Programming
Intermediate Java Programming
Advanced Java Programming
Java Developer I
Intro to Unix
Basics of Unix Admin

DESCRIPTION
--------------------------------------------------
Advanced Unix Admin
Unix Tips and Techniques
Systems Analysis
Project Management
Database Design
Internet Protocols
```

FIGURE 2.38
Executing a SELECT statement in SQL*Plus

If you want to edit the statement, you can type the EDIT or ED command at the SQL> prompt. This evokes the Notepad editor in Windows (see Figure 2.39) or the default editor currently set in SQL*Plus. When you use the EDIT command at the SQL prompt, SQL*Plus stays open in the background, and your text editor is in the foreground, automatically displaying the SQL statement in the buffer.


C:\WINDOWS\system32\cmd.exe - sqlplus
afiedt.buf - Notepad
File Edit Format View Help

```
Project Management
Database Design
Internet Protocols
Java for C/C++ Programmers
GUI Design Lab
Intro to SQL
Oracle Tools
PL/SQL Programming

DESCRIPTION
--------------------------------------------------
Intro to the Internet
Intro to the BASIC Language
Operating Syste
Network Adminis
Java Developer
Database Systen
Java Developer
DB Programming

30 rows selecte

SQL> ed
Wrote file afiedt.buf
```

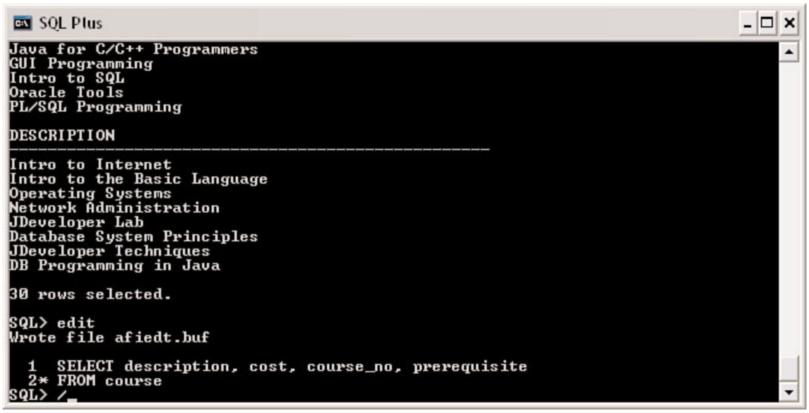
```
SELECT description, cost, course_no, prerequisite
  FROM course
/
```

FIGURE 2.39
Using the Notepad editor to edit a SQL statement in SQL*Plus

For quick editing of statements, simply make your changes here, save the file, and exit Notepad, which brings you back to SQL*Plus. (In this example, additional columns were added to the query.)

When the changes are saved, you exit Notepad, and the revised SQL statement is placed in the buffer area of SQL*Plus. You can then execute the statement with the forward slash on a new line, as shown in Figure 2.40.

When you invoke an editor, the SQL statement ends with a forward slash on a separate line at the end. SQL*Plus adds this character to the file so the file can be executed within SQL*Plus. Also, when you invoke the editor from SQL*Plus, you can't go back to the SQL*Plus screen until you close the editor.



```
Java for C/C++ Programmers
GUI Programming
Intro to SQL
Oracle Tools
PL/SQL Programming

DESCRIPTION
--------------------------------------------------
Intro to Internet
Intro to the Basic Language
Operating Systems
Network Administration
JDeveloper Lab
Database System Principles
JDeveloper Techniques
DB Programming in Java

30 rows selected.

SQL> edit
Wrote file afiedt.buf

  1  SELECT description, cost, course_no, prerequisite
  2* FROM course
SQL> /
```

FIGURE 2.40
SQL statement that is about to be executed with the forward slash command

The SQL*Plus Buffer

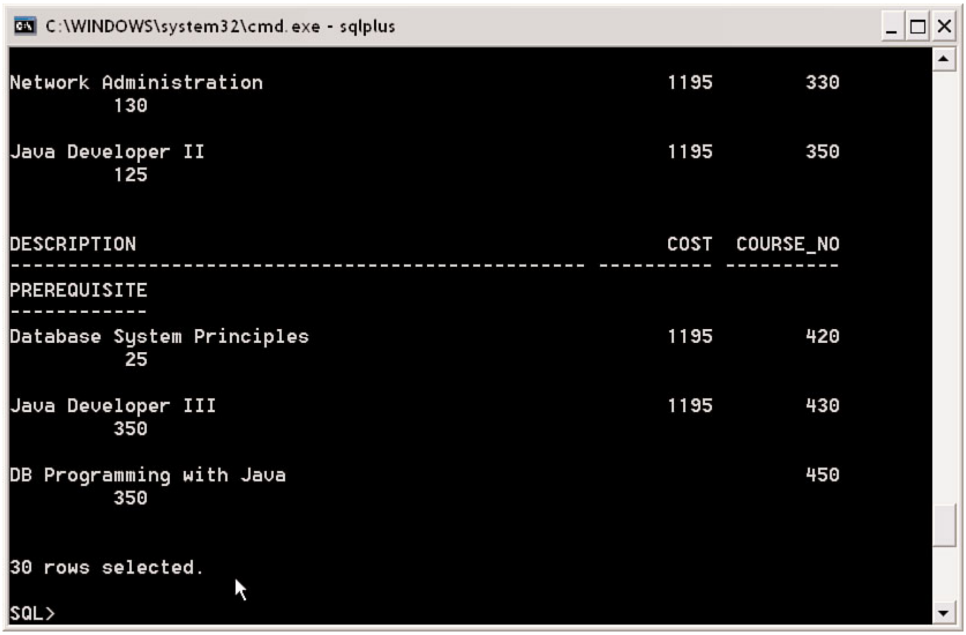
SQL*Plus stores the last SQL command you typed in what is referred to as the *SQL*Plus buffer.* You can re-execute a statement by just pressing the / key or typing the SQL*Plus RUN command. The most recent statement stays in the buffer until you enter another SQL command. You can use the SQL*Plus LIST command, or simply the letter L, to list the contents of the buffer. The semicolon or the slash, either of which executes the statement, is not stored in the buffer. The asterisk next to the number 2 indicates that this is the current line in the buffer. (Aside from using Notepad or any other editor, you can also use SQL*Plus's arcane Line Editor commands; these commands are described in Appendix C, "SQL*Plus Command Reference.")

```
SQL>LIST
  1 SELECT description, cost, course_no, prerequisite
  2*  FROM course
```

In the Windows operating system, you can use the up and down arrow keys to recall previous SQL and SQL*Plus statements.

Formatting SQL*Plus Results

The result set is difficult to read when data "wraps" itself onto the next line. The result may look similar to the screen shown in Figure 2.41. This wrapping often occurs when your SELECT statement contains multiple columns. To help you view the output more easily, SQL*Plus offers a number of formatting commands. Note that these commands are not SQL commands but commands specific only to SQL*Plus.



```
Network Administration                                   1195       330
        130

Java Developer II                                        1195       350
        125


DESCRIPTION                                              COST  COURSE_NO
-------------------------------------------------- ---------- ----------
PREREQUISITE
------------
Database System Principles                               1195       420
         25

Java Developer III                                       1195       430
        350

DB Programming with Java                                            450
        350


30 rows selected.

SQL>
```

FIGURE 2.41
SQL*Plus output wrapped

FORMATTING COLUMN ATTRIBUTES

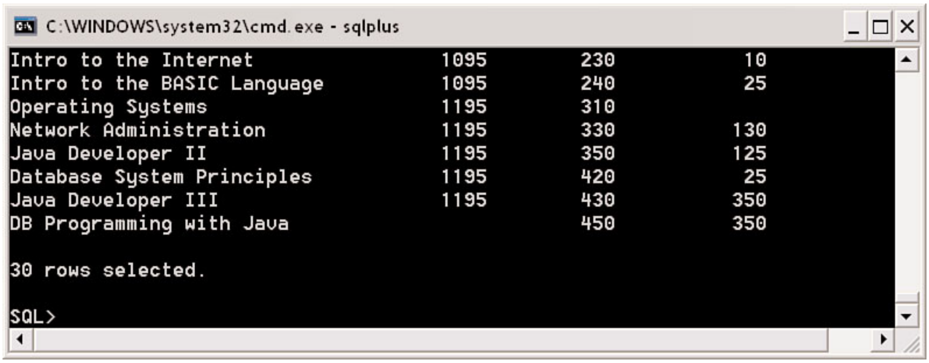
The SQL*Plus COLUMN command allows you to specify format attributes for a column.

The following statement formats the DESCRIPTION column to display a maximum of 30 characters. If the values in the columns do not fit into the space allotted, the data wraps within the column. The column headings are truncated to the specified length.

```
COL description FORMAT A30
```

When you re-execute the SQL statement, the result is more readable, as you see in the result set shown in Figure 2.42. The format for the column stays in place until you either re-specify the format for the column, specifically clear the format for the column, or exit SQL*Plus. To clear all the column formatting, execute the CLEAR COLUMNS command in SQL*Plus.

The SQL*Plus commands such as the FORMAT command are not SQL commands and therefore do not require a semicolon or forward slash.



```
Intro to the Internet                   1095         230           10
Intro to the BASIC Language             1095         240           25
Operating Systems                       1195         310
Network Administration                  1195         330          130
Java Developer II                       1195         350          125
Database System Principles              1195         420           25
Java Developer III                      1195         430          350
DB Programming with Java                             450          350

30 rows selected.

SQL>
```

FIGURE 2.42
SQL*Plus output formatted

FORMATTING NUMBERS

If the column is a NUMBER data type column, you can change the format with a *format model* in the COLUMN command. For example, the 9 in the format model 999.99 represents the numeric digits, so the number 100 is displayed as 100.00. You can add dollar signs, leading zeros, angle brackets for negative numbers, and round values to format the display as you like.

```
COL cost FORMAT $9,999.99
        SELECT DISTINCT cost
          FROM course
              COST
        ----------
         $1,095.00
         $1,195.00
         $1,595.00


        4 rows selected.
```

One row in the COURSE table contains a null value in the COST column. As mentioned previously, DISTINCT recognizes one or more null values in a column as one distinct value when returning a result set.

If you do not allot sufficient room for numbers to fit in a column, SQL*Plus shows # symbols instead of the numbers.

```
COL cost FORMAT 999.99
           COST
        -------
        #######
        #######
        #######


        4 rows selected.
```

For more SQL*Plus COLUMN FORMAT commands, see Appendix C.

Displaying the Number of Rows Returned

SQL*Plus sometimes does not show the number of rows returned by a query but rather depends on the feedback settings for your SQL*Plus session. Typically, the feedback is set to six or more rows. In the previous example, the feedback was set to 1, which displays the feedback line even when there is only one row returned. You will find this setting useful if your result set returns less than the default six rows and if any of the rows return nulls, which display as blanks by default. Otherwise, you might think it is not a row or value. To display the exact number of rows returned until you exit SQL*Plus, enter the SET FEEDBACK SQL*Plus command.

```
SET FEEDBACK 1
```

To display your current settings, use the SHOW ALL command or simply SHOW FEEDBACK. If you want to retain certain SQL*Plus settings, you can create a login.sql file for your individual computer in a client/server setup. You can also create a glogin.sql file for all users if you want them all to have identical settings (see Appendix C, "SQL*Plus Command Reference.")

SQL*Plus Commands versus SQL Statements

A SQL*Plus command is specific to the SQL*Plus execution environment. Unlike a SQL statement, a SQL*Plus command does not require a semicolon or backslash in order to be executed. SQL*Plus commands are commonly used for formatting query and report results, setting environment variables and runtime options, describing table and object definitions, executing batch scripts, and performing database administration tasks.

SQL*Plus commands come in handy when you have to create repeatable scripts; you will learn more about some of the useful SQL*Plus commands in Chapter 14.

Saving and Running SQL Statements in SQL*Plus

You can save your SQL statements within SQL*Plus. Type the following statement.

```
SELECT *
  FROM course
```

Now edit the file in Notepad and select Save As to save it with the name C:\examples\myfile.sql. Exit Notepad and type and execute a new, different SQL statement.

```
SELECT state
  FROM zipcode
```

This new statement is now in the buffer; however, you can execute a different SQL statement, such as the one you saved in myfile.sql, with the START or @ command.

```
SQL>@c:\examples\myfile
```

The statement in the file runs, producing a result set. Because the file already contains a forward slash, the SQL statement is executed automatically. If you save myfile with an extension other than .sql, you must type the file name and extension. If you want to change myfile again, simply type the following. Notepad will open, with myfile.sql containing your SQL statement.

```
SQL>ED c:\examples\myfile
```

Discontinuation of the SQL*Plus for Windows GUI Version

So far, you have learned how to use the Windows command-line version of SQL*Plus. In prior Oracle versions, a SQL*Plus Windows GUI version for the Windows Desktop was available. The functionality of the Windows command-line version and the Windows GUI version was almost identical. Starting with Oracle 11g, Oracle no longer ships the Windows version of the product and replaced it with the SQL Developer software.

Differences Between SQL Developer and SQL*Plus

Throughout this book you will see both SQL*Plus and *SQL Developer* mentioned. For the most part, the basic functionality of the two products is identical with respect to the SQL language.

One of the most obvious differences is the visual display of the result set and the user interface. Furthermore, instead of typing and then executing commands, SQL Developer allows you to perform many operations with a few mouse clicks. Table 2.1 highlights a number of the notable differences.

While SQL Developer simplifies many tasks, it can also allow a novice user to perform some potentially damaging actions using the menus. A good understanding of the effects of the underlying SQL operations is essential for making SQL Developer a productive tool. As you learn more about the SQL language and get more experienced in SQL Developer, you will appreciate many of its advanced features.

TABLE 2.1 Key Differences Between SQL Developer and SQL*Plus

SQL DEVELOPER	SQL*PLUS
Graphical user interface.	Command-line interface
Editing in the SQL Worksheet text box.	Editing from the SQL> prompt via the command-line editor or an invoked editor.
SQL Developer automatically handles the formatting of columns to fit the width of the screen.	Columns may not fit the whole width of your screen. Use various SQL*Plus formatting commands to make them display on one line.
Allows connectivity to some non-Oracle databases.	Executes only against an Oracle database.
Works with Oracle versions 9.01 and above.	All Oracle versions are supported.

TABLE 2.1 Continued

SQL DEVELOPER	SQL*PLUS
Has an auto-completion syntax feature.	Requires knowledge of exact syntax and object names.
Many of the typical SQL actions can be performed through the GUI, without writing an explicit SQL statement.	Requires typing of the SQL command.
A null value is easily distinguishable as "(null)" in the Results tab.	There is no special display of null values unless you issue the SQL*Plus command SET NULL text.

▼ LAB 2.3 EXERCISES

a) After you have logged in to SQL*Plus with the user ID student and the password learn, what information does the SQL*Plus screen show you?

b) What do you learn when you type the command DESCRIBE instructor and press Enter?

c) Describe the result set you get when executing the following SQL statement. Format the result to make it more readable.

```
SELECT *
  FROM grade_type
```

d) Explain what happens, and why, when you execute the following SQL statement.

```
SELECT instructor_id, instructor_name
  FROM instructor
```

▼ LAB 2.3 EXERCISE ANSWERS

a) After you have logged in to SQL*Plus with the user ID student and the password learn, what information does the SQL*Plus screen show you?

ANSWER: The screen shows which version of SQL*Plus you are using, the current date and time, Oracle copyright information, and the version of the Oracle database software you are connected to. After this information is displayed, you see the SQL> command prompt. You can enter commands at this prompt.

b) What do you learn when you type the command DESCRIBE instructor and press Enter?

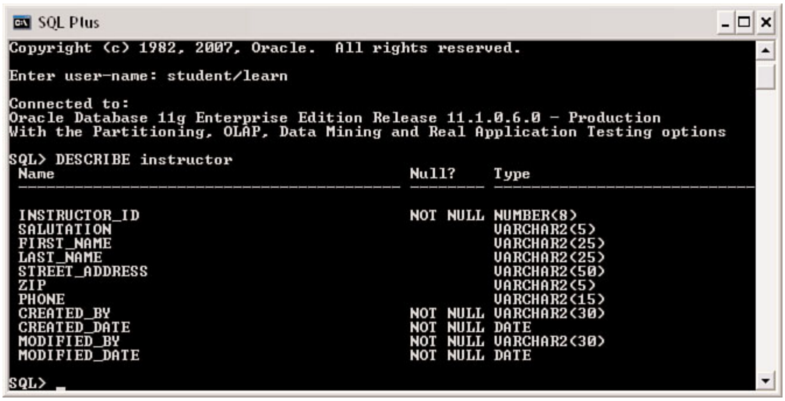
ANSWER: You can display the structure of your table with the SQL*Plus DESCRIBE command (see Figure 2.43). You can abbreviate the command as DESCR.

You can execute the same DESCRIBE command in SQL Developer, with the same result.

c) Describe the result set you get when executing the following SQL statement. Format the result to make it more readable.

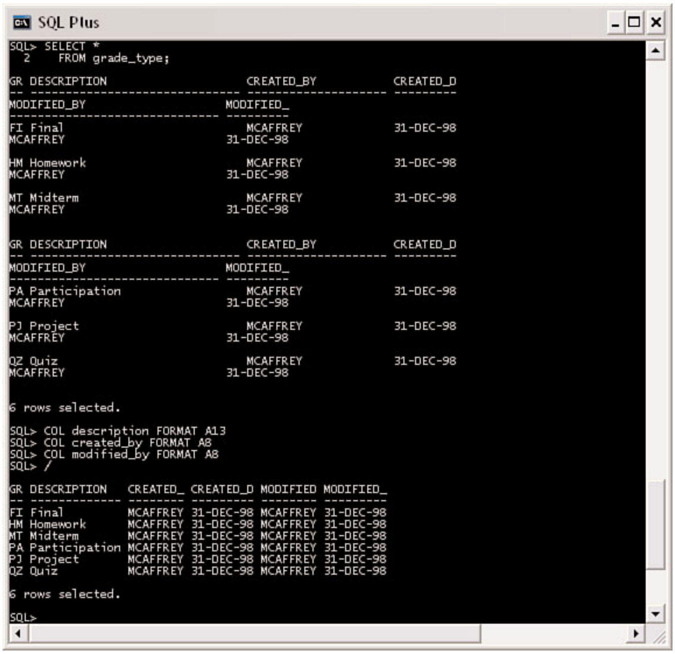
```
SELECT *
  FROM grade_type
```

ANSWER: All columns and rows of the GRADE_TYPE table are returned in the result set. Your result may resemble the first listing of SQL output in Figure 2.44, displaying the wrapped columns. The second result shows the output nicely formatted after the SQL*Plus COLUMN commands are issued.



```
Copyright (c) 1982, 2007, Oracle.  All rights reserved.

Enter user-name: student/learn

Connected to:
Oracle Database 11g Enterprise Edition Release 11.1.0.6.0 - Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL> DESCRIBE instructor
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------

 INSTRUCTOR_ID                             NOT NULL NUMBER(8)
 SALUTATION                                         VARCHAR2(5)
 FIRST_NAME                                         VARCHAR2(25)
 LAST_NAME                                          VARCHAR2(25)
 STREET_ADDRESS                                     VARCHAR2(50)
 ZIP                                                VARCHAR2(5)
 PHONE                                              VARCHAR2(15)
 CREATED_BY                                NOT NULL VARCHAR2(30)
 CREATED_DATE                              NOT NULL DATE
 MODIFIED_BY                               NOT NULL VARCHAR2(30)
 MODIFIED_DATE                             NOT NULL DATE

SQL> _
```

FIGURE 2.43
Executing the SQL*Plus DESCRIBE command



```
SQL> SELECT *
  2    FROM grade_type;

GR DESCRIPTION                    CREATED_BY           CREATED_D
-- ------------------------------ -------------------- ---------
MODIFIED_BY                    MODIFIED_
------------------------------ ---------
FI Final                          MCAFFREY             31-DEC-98
MCAFFREY                       31-DEC-98

HM Homework                       MCAFFREY             31-DEC-98
MCAFFREY                       31-DEC-98

MT Midterm                        MCAFFREY             31-DEC-98
MCAFFREY                       31-DEC-98


GR DESCRIPTION                    CREATED_BY           CREATED_D
-- ------------------------------ -------------------- ---------
MODIFIED_BY                    MODIFIED_
------------------------------ ---------
PA Participation                  MCAFFREY             31-DEC-98
MCAFFREY                       31-DEC-98

PJ Project                        MCAFFREY             31-DEC-98
MCAFFREY                       31-DEC-98

QZ Quiz                           MCAFFREY             31-DEC-98
MCAFFREY                       31-DEC-98


6 rows selected.

SQL> COL description FORMAT A13
SQL> COL created_by FORMAT A8
SQL> COL modified_by FORMAT A8
SQL> /

GR DESCRIPTION   CREATED_ CREATED_D MODIFIED MODIFIED_
-- ------------- -------- --------- -------- ---------
FI Final         MCAFFREY 31-DEC-98 MCAFFREY 31-DEC-98
HM Homework      MCAFFREY 31-DEC-98 MCAFFREY 31-DEC-98
MT Midterm       MCAFFREY 31-DEC-98 MCAFFREY 31-DEC-98
PA Participation MCAFFREY 31-DEC-98 MCAFFREY 31-DEC-98
PJ Project       MCAFFREY 31-DEC-98 MCAFFREY 31-DEC-98
QZ Quiz          MCAFFREY 31-DEC-98 MCAFFREY 31-DEC-98

6 rows selected.

SQL>
```

FIGURE 2.44
SQL*Plus output, both unformatted and formatted with SQL*Plus commands

d) Explain what happens, and why, when you execute the following SQL statement.

```
SELECT instructor_id, instructor_name
  FROM instructor
```

ANSWER: Oracle returns an error because the column INSTRUCTOR_NAME does not exist.

```
SELECT instructor_id, instructor_name
                      *
ERROR at line 1:
ORA-00904: "INSTRUCTOR_NAME": invalid identifier
```

If you use SQL*Plus, the asterisk in the error message indicates where the error occurs in the SQL statement.

The following is the correct SQL query.

```
SELECT instructor_id, last_name
  FROM instructor
```

Lab 2.3 Quiz

In order to test your progress, you should be able to answer the following questions.

1) A TNSNAMES entry is always required when you are using SQL*Plus to access an Oracle database.

_______ a) True
_______ b) False

2) SQL*Plus commands can be executed against non-Oracle databases.

_______ a) True
_______ b) False

3) SQL*Plus works with all versions of Oracle.

_______ a) True
_______ b) False

4) A SQL*Plus command must be ended with either a semicolon or a backslash.

_______ a) True
_______ b) False

ANSWERS APPEAR IN APPENDIX A.

▼ WORKSHOP

The projects in this section are meant to prompt you to utilize all the skills you have acquired throughout this chapter. The answers to these projects can be found at the companion Web site to this book, located at www.oraclesqlbyexample.com.

1) Use SQL Developer to retrieve all the STATE values from the ZIPCODE table, without repeating the values.

2) Recall one of the statements you executed in Lab 2.2, using the SQL History tab.

3) What happens if you try to log on to SQL*Plus with the uppercase version of the password learn?

4) Execute the following statements in SQL*Plus and record your observations.

```
SET NULL 'NULL'
SELECT DISTINCT cost
  FROM course
```

INDEX