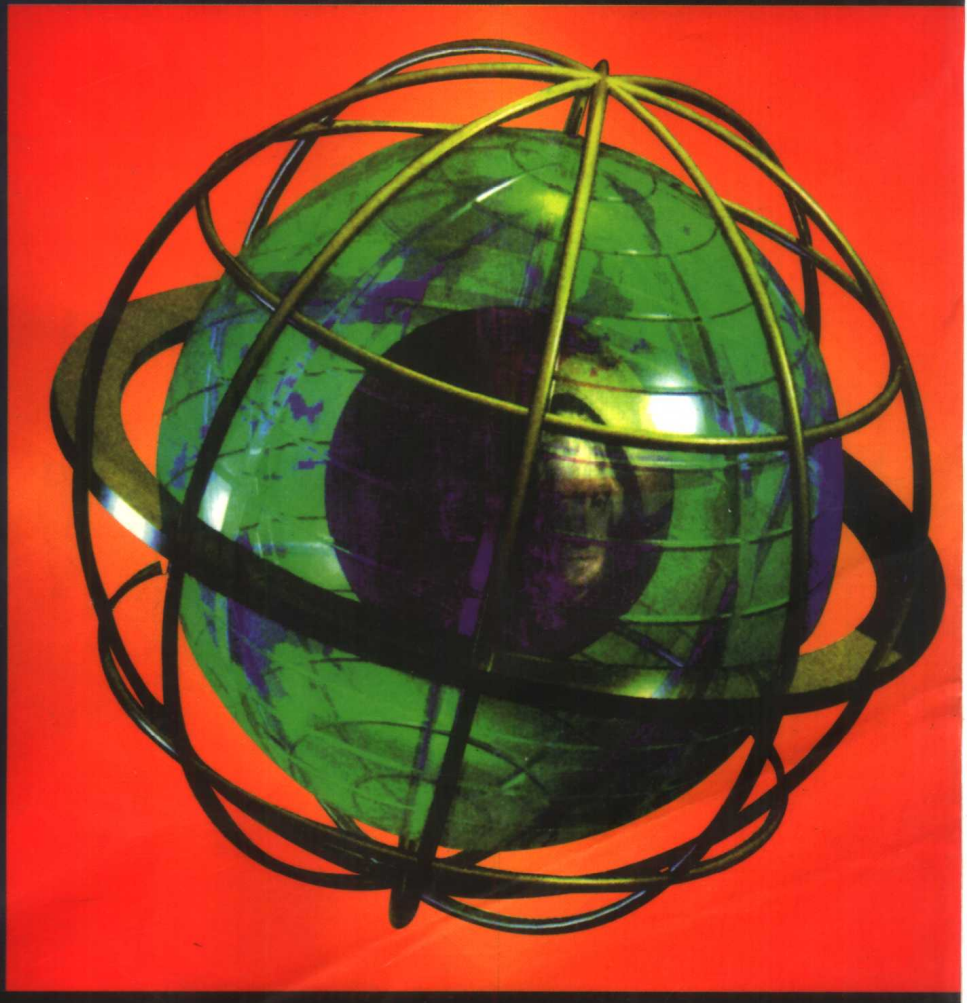
CMP

Oracle How-To

ORACLE

开发人员指南

(美) Edward Honour 著
译友翻译组 译



计算机软件开发与程序设计系列丛书

机械工业出版社
西蒙与舒斯特国际出版公司

本书是为 Oracle 开发人员及数据库管理人员而编写的，旨在帮助读者解决日常开发应用程序中遇到的难题。

本书共分 20 章，每一章都集中讨论 Oracle 数据库及其工具的某个方面，涉及的主题范围很广，从最基本的到非常高级的，这些内容对读者开发或支持 Oracle 应用程序时都很有帮助。

Edward Honour:Oracle How-to.

Authorized translation from the English language edition pubished by The Waite Group,Inc.

本书版权登记号：图字：01-97-0936

图书在版编目(CIP)数据

Oracle 开发人员指南/(美)奥那(Honour,E.)著；译友翻译组译. -北京：机械工业出版社，1998.1

(计算机软件开发与程序设计系列丛书)

ISBN 7-111-05891-7

Ⅰ.O… Ⅱ.①奥… ②译… Ⅲ.关系数据库-数据库管理系统,Oracle Ⅳ.TP311.13

中国版本图书馆 CIP 数据核字(97)第 17097 号

出 版 人：马九荣(北京市百万庄南街 1 号 邮政编码 100037)

责任编辑：温莉芳

昌平环球印刷厂印刷 · 新华书店北京发行所发行

1998 年 1 月第 1 版第 1 次印刷

787mm×1092mm 1/16 · 31.75 印张 · 772 千字

印数：0001－5000 册

定价：55.00 元

前　言

Oracle 关系型数据库管理系统(Relational Database Management System,简称 RDBMS)是世界上最流行的关系数据库。它是一个极其强大、灵活和复杂的系统,可以用于构造从小型的单用户系统到支持数千个并发用户的大中型应用项目。现在,市场上已经有各种支持 Oracle 的应用系统以及多种多样的相应的 Oracle 工具。应用开发人员可以使用 Oracle 的 Developer 2000、PL/SQL 以及最新的 Oracle WebServer 构造新的应用系统。Oracle 的 Designer 2000 是一个高级的 CASE 工具,它用于设计和生成复杂的应用程序并且支持结构化设计方法。Oracle 数据库管理员(Oracle database administrator)支持复杂的 Oracle 体系结构,为 Oracle 应用程序的用户提供了一个可靠的高度可重装配调整的数据库。Oracle 建筑师(Oracle architect)可以设计用于支持企业运作的数据库。Oracle 适合于各种任务,从单个数据库系统,到使用"快照"(Snapshot)和"数据复制"(replication)功能的分布式系统,以及使用"Oracle 并行服务器"(Oracle Parallel Server)的"集簇"(clustered)或大型(massively)并发系统。对于在使用 Oracle 方面所面临的问题,本书给出了面向实际应用的解决办法。

本书分为 20 章,每一章都集中讨论 Oracle 数据库及其工具的某个方面。

第 1 章,SQL * Plus。本章介绍 SQL * Plus 并说明如何使用 Oracle 系统中的这个基本工具。SQL * Plus 提供了对 Oracle 的最直接的访问,使用它可以有效地提高 Oracle 环境中的工作效率。

第 2 章,用户帐户和角色。本章介绍如何创建和管理用户帐户和角色。Oracle 系统中的每个用户必须被赋予一个用户帐户以便能够连接到该数据库。权限可以直接被授予用户帐户也可以通过角色间接进行授予。

第 3 章,安全性。本章给出的技巧可以用来对 Oracle 数据库实现数据安全性管理。这里会使用到第 2 章给出的许多技巧并将对它们进行扩展以提供有用的安全性管理技巧。

第 4 章,表格。表格是 Oracle 中数据的基本存储对象。本章给出了关于在 Oracle 数据库中创建和管理表格的技巧。

第 5 章,视图。视图用于以独立于实际数据存储方式的格式表示数据。本章给出了有关在 Oracle 中创建和管理视图的技巧。

第 6 章,序列。序列是 Oracle 中的一项强大的功能,它可以在数据库中产生唯一的数。本章给出有关序列的创建、管理和使用的技巧。

第 7 章,替代名。替代名是数据库对象的别名,它对于多用户应用程序是十分重要的。本章给出有关替代名的管理和有效使用的技巧。

第 8 章,数据管理。本章给出一些关键技巧来管理数据库中的数据。这里分别就数据库中数据的创建、修改和删除进行讨论。

第 9 章,数据查询。本章给出一些关键技巧来查询数据库中的数据。Oracle 是一个功能强大的数据库,可以使用许多高级方法检索数据。

第 10 章，日期、文本和数字。本章介绍了用于处理日期、文本和数字的 Oracle 函数。Oracle 给出了许多用于处理数据库数值的内装函数。

第 11 章，分布式数据库。用单个大型数据库处理全部的数据库请求的日子已经过去。本章给出关于在多个 Oracle 数据库之间进行分布数据的技巧。本章同时给出数据库链接与快照的管理。

第 12 章，PL/SQL。本章讨论 Oracle 的过程化语言 PL/SQL。在本章的主题中会涉及到 PL/SQL 的高级特性。本书全部内容都依赖于本章中所介绍的技巧。通过学习第 12 章给出的代码，读者会对 PL/SQL 有一个彻底的了解。

第 13 章，过程、函数和包。本章会讨论 Oracle 数据库的一些最强有力的特性。内嵌过程、函数和包允许应用程序使用数据库服务器处理与应用程序相关的代码。本章给出的技巧将有助于开发功能强大的内嵌过程、函数和包。

第 14 章，内装包。本章介绍使用与 Oracle 一同提供的功能强大的内装包的技巧。在本章中会讨论由 Oracle 提供的内装包如何执行用户通常用编码来实现的许多任务。

第 15 章，数据库触发器。本章介绍数据库触发器的创建和管理。数据库触发器允许在数据库级别上增强数据库的商业法则。

第 16 章，应用程序性能调整。本章介绍的技巧用于提高运行在 Oracle 数据库上的应用程序的性能。既使是设计得最好的应用程序也可能有一些性能问题，这可以使用本章给出的技巧来解决。

第 17 章，数据库性能调整。本章介绍的技巧用于提高运行于 Oracle 数据库之上的所有应用程序的性能。尽管本章给出的主题是专门面向数据库管理员的，但也同时为应用程序开发者提供了执行 Oracle 数据库性能调整的一个很好的基础。

第 18 章，用于 OLE 的 Oracle Objects。本章介绍了用于 OLE 的 Oracle Objects，这会将 Oracle 数据库的功能扩展到支持 OLE 自动操作的任何开发工具上。

第 19 章，用 Oracle WebServer 建立 Web 应用。本章提供了开发使用 Oracle WebServer 的 Intranet 和 Internet 应用程序的技巧。Web 很快会成为企业计算的一个组成部分，同时 Oracle 的 Web 产品会为产业界带来一个主要的冲击。

第 20 章，WebServer 管理。本章提供了用于 Oracle WebServer 管理的技巧。Oracle WebServer 可以使用任何浏览器通过 Web 文档来管理。本章给出的技巧对于有效管理 WebServer 环境是十分重要的。

附录，错误消息处理。本附录使读者直接查阅在开发和维护应用程序遇到的错误消息的原因及相应的解决方法。

本书是写给初级、中级和高级 Oracle 开发者以及数据库管理员的。本书旨在解决使用 Oracle 数据库应用程序的通常开发过程中发生的问题。本书提供给读者许多基础的知识，这有助于读者解决其他类似的问题。本书涉及的主题包含范围很广，从最基本的到非常高级的。这些对读者在开发或支持 Oracle 应用程序时都很有帮助。本书同时给出了关于程序设计问题和技巧的特别示例。

另外，与本书配套的 CD 盘中包含了 Oracle Corporation 提供的产品及原作者开发的产品。有对此盘感兴趣的读者可与机械工业出版社华章图文信息有限公司联系。

目　录

第 1 章　SQL * Plus

在开发用于 Oracle 数据库的应用程序方面，SQL * Plus 是最强有力的工具之一。SQL * Plus 提供了对 Oracle 的最直接的访问。尽管看上去 SQL * Plus 不象图形查询工具那样与用户友好，但是它提供了更大的灵活性，并且可以运行于支持 Oracle 的各种平台。与大多数图形查询工具不同，SQL * Plus 可以用于操纵数据和创建数据库对象。

SQL * Plus 在应用程序开发过程中扮演着很重要的角色。它是一个功能强大的原型构造(Prototyping)工具，用户可以使用它开发和测试 SQL 语句，然后再将这些语句集成到应用程序中，因此 SQL * Plus 对于使用 Oracle 的任何开发者而言都是很有帮助的。本章将介绍如何使 SQL * Plus 的工作成为整个开发过程的一部分，这里的示例将说明 SQL * Plus 如何有效地帮助开发者开发 Oracle 应用程序。

1.1　怎样将 SQL 语句保存到文件中

使用 Oracle 越多，开发功能强大的 SQL 语句也就越容易。但是如何避免在每次需要时重复键入相同的 SQL 语句呢？SQL * Plus 可以让开发者将 SQL 语句保存到文件中，这样可以在下次需要时再使用它们。本节将指导如何完成将 SQL 语句保存到文件中的任务。

问题的提出

通常，用户需要多次执行 SQL 语句。当书写要重复执行的很长的 SQL 语句或一般的 SQL 语句时，用户需要将它保存到文件中以便以后再次执行。怎样将 SQL 语句保存到文件中呢？

实现技术

SQL * Plus 的 SAVE 命令用于将在 SQL 缓冲区中当前存放的 SQL 语句保存到文件中。SQL 缓冲区(SQL buffer)中存放了输入到 SQL * Plus 中或从文件中检索出的最新 SQL 语句。只要 SQL * Plus 将一条语句识别为 SQL 语句，便会将 SQL 缓冲区中存放的当前语句替换为新识别的语句。可以将 SQL 缓冲区理解为 SQL * Plus 中的 Windows 剪贴板。SQL 缓冲区中的数据可以用 SAVE 命令保存到文件中，用 GET 命令从文件中取出，并且通过各种 SQL * Plus 命令进行编辑。本章 1.4 节会详细探讨如何编辑 SQL 缓冲区中的内容。

SAVE 命令的语法是

SAV[E] filename[.ext][CRE[ATE]|REP[LACE]|APP[END]]

用括号[]标明的字符和选项都是可选的。用竖线分隔的选项是互斥的。SAVE 命令的缺省动作是创建一个新文件。如果该文件存在，就需要 REPLACE 或 APPEND 关键字。在缺省情况下，SQL * Plus 会使用.SQL 扩展名保存文件以将它识别为一个 SQL * Plus 文件。如果想要用不同的扩展名进行保存，可以在执行 SAVE 命令时进行指定。

注意：SUFFIX 系统变量可以用于更改缺省扩展名，可以将.SQL 扩展名指定为其它的取值。关于设置系统变量的更多信息，可以参见本章 1.7 节。

实现步骤

1)输入想要保存的 SQL 语句。不要用分号结束语句。如果 SQL 语句以分号结尾，SQL * Plus 会立即执行它。

```
SQL>  SELECT EMPNO,ENAME
```

```
2  FROM EMP
3  ORDER BY EMPNO
4  ……
```

与 SQL * Plus 命令不同,SQL 语句可以包含许多行而无需行连续字符。SQL * Plus 命令的行连续字符是连字符-。可以通过按下 ENTER 键开始新的一行。SQL * Plus 自动按照按下的 ENTER 键个数进行行的编号。如果在一个空行上按下 ENTER,SQL * Plus 会回到 SQL 提示行。输入的 SQL 语句会保留在 SQL 缓冲区中直到被清除,由另一个 SQL 语句替换,或者退出 SQL * Plus。

2)使用 SAVE 命令保存文件。如果处理成功,SQL * Plus 会提示文件被创建。

```
SQL> SAVE EMPQRY
Created file EMPQRY
```

创建的文件自动放在当前工作目录中,并以.SQL 为扩展名。用户可以指明一个完整的路径,并使用不同的扩展名保存文件。如果该文件存在,会反馈一条错误消息。图 1-1 给出 SQL * Plus 显示错误消息的情形。

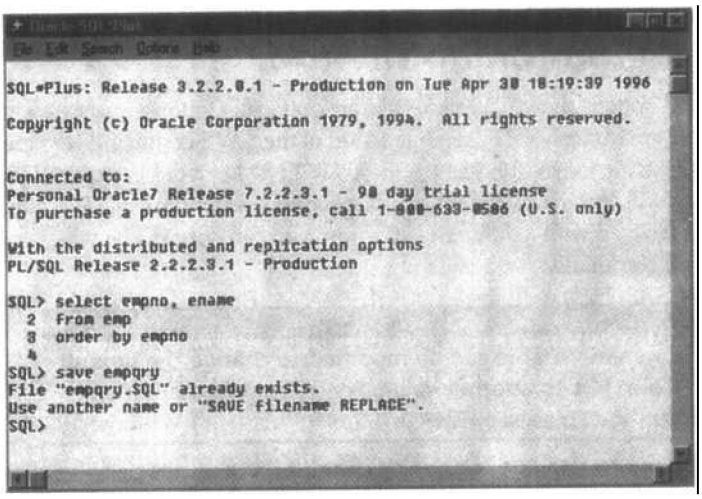


图 1-1　SQL * Plus 在试图创建文件时显示错误消息

3)因为一个 SQL * Plus 命令文件可以包含多个 SQL 语句,APPEND 选项可使你在现有文件的末尾添加保存在 SQL 缓冲区中的内容。使用 SAVE 命令和 APPEND 选项可以将 SQL 缓冲区中的内容保存在同一个文件中。

```
SQL>  SAVE EMPQRY APPEND
Appended file to empqry
```

如果在试图向一个不存在的文件添加数据时,该文件会被创建。

工作原理

SQL * Plus SAVE 命令是一条 SQL * Plus 命令(不是一条 SQL 语句)。执行该命令并不会替换 SQL 缓冲区中的内容。如果使用其它的工具执行该 SAVE 命令,会发生一个错误。步骤 1)创建了一条 SQL 语句但并未执行它。步骤 2)将在步骤 1)中使用 SAVE 命令创建的 SQL 缓冲区内容保存到一个文件中。步骤 3)将 SQL 缓冲区中的内容附加到现有文件的末尾,这要使

用 SAVE 命令的 APPEND 选项。

说明

在向 SQL 缓冲区输入语句时，SQL * Plus 只识别 SQL 或 PL/SQL 语句。SQL * Plus 命令不会保存在 SQL 缓冲区中。如果想要用 SQL 脚本命令包含格式化命令，请使用 INPUT 命令，该命令会放置在 SQL 缓冲区中的所有语句并且在进行输入时不分析这些语句。在使用 INPUT 命令时请不要在 SQL 语句末尾使用分号，否则会发生错误。

1.2 怎样运行 SQL * Plus 命令文件

SQL * Plus 可以用于执行许多任务，而大多数任务包含不只一个步骤。SQL * Plus 可以将一系列的 SQL、PL/SQL 和 SQL * Plus 命令保存到一个文件中，并将它们作为一个单一语句运行。SQL * Plus 命令文件可以进行嵌套以创建一个复杂的事件序列。本节将探讨如何产生和运行 SQL * Plus 命令文件。

问题的提出

可以使用 GET 命令从磁盘向 SQL 缓冲区装入一条 SQL 语句。用户可能想要在 SQL * Plus 中运行一条 SQL * Plus 命令而无需先将它装入 SQL 缓冲区中。这是十分重要的，因为用户的 SQL * Plus 命令文件中包含了格式化命令和多个语句。怎样才能运行一个 SQL 命令文件呢？

实现技术

START 命令执行一个 SQL * Plus 命令文件。START 命令可以用@符号替换，这两个命令是等价的。当执行一个 START 命令时，该文件中的每条语句会顺序执行。文件中出现的错误会显示出来，而文件会继续进行。SQL * Plus 命令文件可以嵌套。START 命令可以是 SQL * Plus 命令文件的一部分，这样便可以运行其它文件的内容了。嵌套命令文件的技巧可以用来将系统变量设置为查询结果，详见 1.9 节所示。

实现步骤

1)与本书配套的 CD 上的 CHP1-1.sql 是一个 SQL * Plus 命令文件。执行该文件可以创建或替换本章中使用的示例表格。文件的内容说明了 SQL * Plus 命令文件的结构。

```
SET TERMOUT OFF
SPOOL CHP1-1.LOG
DROP TABLE DEPT1_1;
DROP TABLE EMP1_1;
CREATE TABLE DEPT1_1(
    DEPTNO      NUMBER(6),
    DNAME       VARCHAR2(30));
CREATE TABLE EMP1_1(
    EMPNO       NUMBER(6),
    ENAME   VARCHAR2(30),
    SALARY      NUMBER(12,2),
    DEPTNO      NUMBER(6));
INSERT INTO DEPT1_1
    VALUES (1,'MARKETING');
INSERT INTO DEPT1_1
```

```
    VALUES (2,'SALES');
INSERT INTO DEPT1_1
    VALUES (3,'ACCOUNTING');
INSERT INTO EMP1_1
    VALUES (1,'SMITH,JOHN',24000,1);
INSERT INTO EMP1_1
    VALUES (2,'JONES,MARY',42000,1);
INSERT INTO EMP1_1
    VALUES (3,'BROWN,BILL',36000,2);
INSERT INTO EMP1_1
    VALUES (4,'CONWAY,JIM',52000,3);
INSERT INTO EMP1_1
    VALUES (5,'HENRY,JOHN',22000,3);
INSERT INTO EMP1_1
    VALUES (6,'SMITH,GARY',43000,2);
INSERT INTO EMP1_1
    VALUES (7,'BLACK,WILMA',44000,2);
INSERT INTO EMP1_1
    VALUES (8,'GREEN,JOHN',33000,2);
INSERT INTO EMP1_1
    VALUES (9,'JONHSON,MARY',55000,3);
INSERT INTO EMP1_1
    VALUES (10,'KELLY,JOHN',20000,2);
COMMIT;
SPOOL OFF
SET TERMOUT ON
```

第 1 条命令是一条 SQL * Plus 命令，当使用 START 命令运行脚本程序时，它禁止语句的输出。第 2 行将语句产生的输出结果写入由 SPOOL 命令指定的文件中。有关创建输出文件的详细信息，请参见 1.6 节和 1.10 节。两行 DROP TABLE 语句会删除示例表格(如果它们已经存在)。CREATE TABLE 语句会构建示例表格，而 INSERT 语句会向表格添加数据。有关构建和修改表格的详细信息，请参见第 4 章。COMMIT 语句将事务保存到数据库中。SPOOL OFF 语句会停止向文件的写操作，而最后一条语句会将 TERMOOT 系统变量复位为 ON。

2)使用 START 命令执行 CHP1-2.sql。但在操作以前，必须先要连接到数据库并拥有创建表格所需的权限。第 2 章“用户帐户和角色”会讨论用户帐户的创建，1.3 节会讨论连接到不同用户帐户的处理。

```
SQL> START CHP1_1
SQL>
```

由于文件的第 1 行禁止文件处理结果的输出，所以在运行文件时在 SQL * Plus 中未显示任何信息。

工作原理

用@符号替代 START 命令运行一个文件时该文件可以包含 SQL、PL/SQL 和 SQL * Plus 命令。文件中的语句和命令由 SQL * Plus 执行，就如同它们被顺次输入 SQL * Plus 一样。如果要打开一个 SQL * Plus 命令文件并输入每一行，这会象运行该文件一样。命令文件

CHP1-1. sql 包含了 SQL 语句和 SQL * Plus 命令。上面第一行将脚本命令的输出写入文件 CHP1-1. LOG。下面两行删除存在的示例文件。如果这些文件不存在，会有错误信息显示出来，但可以忽略它们。两行 CREATE TABLE 语句会创建两个示例表格。INSERT 命令会将示例数据放入表格中。COMMIT 语句将事务保存到数据库中，并且最后一条语句关闭输出文件。

说明

当要使用很长的 SQL * Plus 脚本命令时，通常使用诸如 Windows Notepad 这样的编辑器并用 START 命令运行会更容易。尽管 SQL * Plus 具有文本编辑功能，但是处理长的语句时这些功能会显得并不容易使用。如果要运行 CHP1-1. sql，就必须连接到一个 Oracle 数据库并拥有创建表格的权限。如果读者不清楚在自己的组织机构中具备什么数据库访问权限，请与数据库管理员进行联系。

1.3 怎样作为另一个用户连接到 SQL * Plus

在 Oracle 环境中，开发者需要连接到不同的用户帐户和数据库。当维护系统时，用户通常需要以不同的用户帐户连接到数据库中。本书将解释如何在 SQL * Plus 中连接到数据库中以及在网络上如何连到不同的数据库上。

问题的提出

在 SQL * Plus 中工作时，用户要使用不同的数据库帐户，也要在自己的网络中连到不同的数据库。如果用户离开 SQL * Plus 后再返回时系统会提示用户帐户。不幸的是，这需要很长时间。怎样才能在 SQL * Plus 中改变用户帐户或数据库呢？

实现技术

CONNECT 命令的缩写为 CON，它可以用于连接到某个数据库上。每个用户要访问的每个数据库都有一个用户帐户和口令。在访问数据库之前，SQL * Plus 要求用户通过 CONNECT 命令指明用户帐户和口令。CONNECT 命令的语法是：CONN[ECT][USERNAME][/PASSWORD][@database]

username 和 password 可以在 CONNECT 命令的同一行中提供，否则 SQL * Plus 将会提示指明它们。如果在 CONNECT 命令中没有提明数据库名字，每台计算机都有一个缺省数据库被设置为要进行连接。在仅有一个数据库的环境中，所有用户都连接到同一个缺省数据库。

实现步骤

1）运行 SQL * Plus 并作为某个用户帐户进行连接。这里，执行 CONNECT 命令作为 WAITE 用户进行连接。WAITE 用户帐户是由与本书配套的 CD 所提供的安装脚本命令创建的。如果还没有创建 WAITE 用户帐户，请运行 CD-ROM 上位于\SAMPLES 目录下的 INSTALL. sql 文件来创建此用户帐户。

```
SQL>CONNECT WAITE/PRESS
Connected.
```

如果对于缺省数据库指定的用户帐户或口令无效，会发生一个错误。图 1-2 说明了当进行无效登录操作时所发生的错误消息。

如果向另一个用户帐户或数据库连接的操作失败，那么当前的 SQL * Plus 会话过程会断开连接。为了能够执行数据库访问语句，就必须重新与数据库进行连接。

2）如果是在家中工作或在周围没有人时，在命令行上输入一个口令是个不错的选择，但是

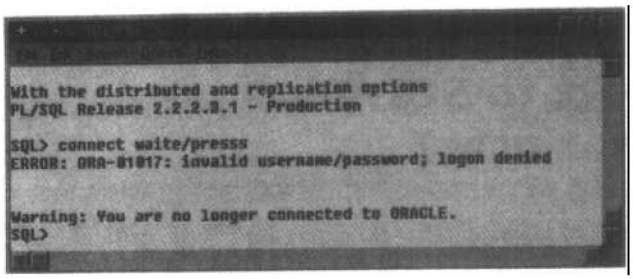


图 1-2　连接失败错误消息

要尽量避免别人读到自己的口令。如果在命令行上没有输入口令，SQL * Plus 会提示输入。

```
SQL> CONNECT WAITE
Enter password: * * * * *
```

当未指定口令时，SQL * Plus 会提示输入。在输入口令时，输入的字符是不能在屏幕上被认出的。

3)在分布式系统中，可能有多个数据库。每个数据库由唯一的名字所标识。如果要连接到除缺省数据库以外其它的数据库，请在 CONNECT 命令上指定数据库名字。下面的语句可以连接到远程数据库。

```
SQL> CONNECT WAITE/PRESS @ other_database
SQL> CONNECT WAITE @ other_database
```

对于远程数据库，用户帐户和口令必须都是有效的。

4)DISCONNECT 命令的缩写为 DISC，可以用来断开与数据库的连接。执行 DISCONNECT 命令以便从数据库中退出。

```
SQL> disconnect
Disconnected from personal oracle7 Release 7.2.2.3.1-90 day trial license
To purchase a production License,call 1-800-633-0586(U.S.only)

With the distributed and replication options
PL/SQL Release 2.2.2.3.1-Production
SQL>
```

一旦执行了 DISCONNECT 命令，就不能对数据库使用 SQL 或 PL/SQL 语句了。如果要重新建立连接，请执行 CONNECT 命令。

工作原理

CONNECT 语句，缩写为 CON，用于连接到某个数据库；而 DISCONNECT 语句，缩写为 DISC，用于断开当前与数据库的连接。在@符号后面可以指定某个数据库。如果未指定数据库，SQL * Plus 会假定连接到缺省数据库。步骤 1)使用指定用户名和口令的命令行进行连接。步骤 2)使用了不提供口令的 CONNECT 命令。当没有指定口令时，SQL * Plus 会提示输入口令。步骤 3)在连接数据库的选项中指明不是缺省数据库，而步骤 4)用 DISCONNECT 命令断开与数据库的连接。退出 SQL * Plus 会自动断开与数据库的会话连接。

说明

与数据库的连接及断开连接操作是很常用的操作。计算机中的缺省数据库应该是最常用

的数据库。如果要离开正在使用的计算机，应该断开会话连接以防止其它人使用自己的帐户。

1.4 怎样使用 SQL * Plus 编辑语句

没有人是完美的打字员，SQL 语句越长，用户越可能犯错误。SQL * Plus 包含了许多编辑功能，允许用户编辑正在使用的 SQL 语句。如果读者熟悉 Microsoft Windows 上的编辑器，SQL * Plus 的编辑功能可能会显得有些过时。但是经过短时间的使用，用户编辑 SQL 语句的速度会有长足的进步。本节将带领读者熟悉 SQL * Plus 环境中所需的基本编辑命令。

问题的提出

用户可能并不善于在 SQL * Plus 中键入语句。因为输入错误而重新输入语句是十分浪费时间的，也是十分烦人的。这便需要掌握在 SQL * Plus 中编辑 SQL 语句。如何能做到这一点呢？

实现技术

对于简单的更改，例如单字拼写错误或要添加一些代码，SQL * Plus 提供了相应的编辑功能。尽管可以使用 SQL * Plus 产生复杂的脚本命令，一些象 Windows Notepad 的编辑器更适合于较长的 SQL 语句。SQL * Plus 中的 EDIT 命令执行操作系统中缺省的编辑器。在 Windows 中，在 Notepad 被执行时，当前的语句会进行显示。表 1-1 列出了 SQL * Plus 中使用的编辑命令。

表 1-1 SQL * Plus 编辑命令

SQL * Plus 编辑命令		定　义
APPEND	A	在行末增加文件
CHANGE/old /new	C	更改行
CLEAR BUFFER	CL BUFF	删除缓冲区中所有行
DEL		删除当前行
INPUT	I	添加一行或多行
LIST	L	列出 SQL 缓冲区中所有的行

实现步骤

1)运行 SQL * Plus 并作为 WAITE 用户帐户进行连接。用 GET 命令将 CHP1-2. sql 装入 SQL 缓冲区。该文件中包含了一个相当长的查询命令，可用之进行编辑操作练习。图 1-3 给出

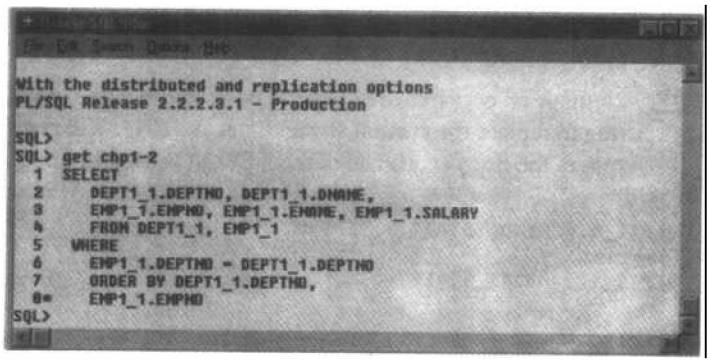


图 1-3 SQL * Plus 的 GET 命令

了在 SQL * Plus 中 GET 命令的执行情况。

在用 GET 命令进行装入时，文件的内容含自动列出。可以在文件名后面提供 NOLIST 子句来禁止对文件内容的列表显示。

2)输入 EDIT 命令启动操作系统中缺省的编辑器。如果对文件进行更复杂的变更操作，EDIT 命令是最容易的方式。图 1-4 说明了将 SQL 缓冲区中的内容装入 Windows Notepad 进行编辑的情形。

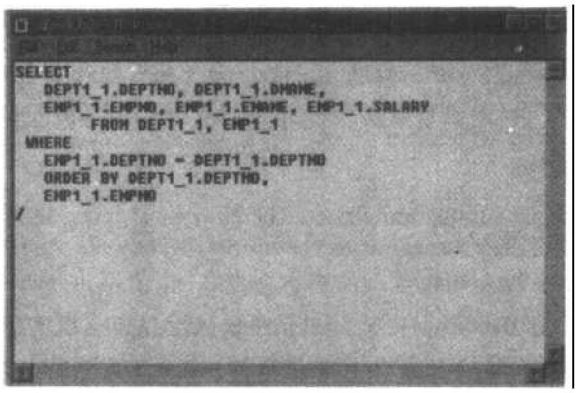


图 1-4　在 Windows Notepad 中正在编辑的 SQL 缓冲区内容

当对语句进行完必要的修改后，保存最后结果，退出编辑器并返回 SQL * Plus。在编辑器中进行的修改会自动装入 SQL * Plus。

3)列出 SQL 缓冲区中的各行内容。LIST 命令用于列出 SQL 缓冲区中各行内容。该命令可以缩写为字母 L。

```
SQL> L
1  SELECT
2    DEPT1_1.DEPTNO,DEPT1_1.DNAME,
3    EMP1_1.EMPNO,EMP1_1.ENAME,EMP1_1.SALARY
4    FROM DEPT1_1,EMP1_1
5  WHERE
6    EMP1_1.DEPTNO=DEPT1_1.DEPTNO
7    ORDER BY DEPT1_1.DEPTNO,
8*   EMP1_1.EMPNO
```

第 8 行的星号将它标识为当前行。如果执行 CHANGE 或 DEL 命令时未指定某一行，那么该命令将针对当前行。

4)使用 LIST 命令并指定一个行号以显示该行，同时将它标定为当前行。

```
SQL> L  1
1* SELECT
```

列出的这一行将成为缓冲区的当前行。如果想要编辑缓冲区中的某行，请使用这个技巧以将它标识为当前行。

5)使用 CHANGE 命令修改正文。CHANGE 命令缩写为字母 C，用来修改当前行中一个正文字符串的第一次出现内容。必须首先使用 LIST 命令和一个行号以将某一行指定为当前

行。

```
SQL> L  2
  2*   DEPT1_1.DEPTNO,DEPT1_1.DNAME,
SQL> C/DNAME/DEPTNO
  2*   DEPT1_1.DEPTNO,DEPT1_1.DEPTNO,
```

如果在该行中包含了要修改的字串的两次出现，那么必须要键入两次 CHANGE

6)如果要在一行中删除某个字符串的一次出现，请不要指定要替换当前串的内容。输入 CHANGE 命令，它会删除第 2 行中 DEPTNO 字符串的第一次出现。

```
SQL>  L  2
  2*   DEPT1_1.DEPTNO,DEPT1_1.DEPTNO,
SQL> C/deptno/
  2* DEPT1_1. ,DEPT1_1.DEPTNO,
```

7)使用 DEL 命令删除 SQL 缓冲区中的行。只输入 DEL 命令会删除当前行，这会使前一行成为当前行。

```
SQL>  DEL
SQL>
SQL>  LIST
  1  SELECT
  2    DEPT1_1.DEPTNO,DEPT1_1.DNAME,
  3    EMP1_1.EMPNO,EMP1_1.ENAME,EMP1_1.SALARY
  4    FROM DEPT1_1,EMP1_1
  5  WHERE
  6    EMP1_1.DEPTNO = DEPT1_1.DEPTNO
  7*   ORDER BY DEPT1_1.DEPTNO,
```

8)可以在 DEL 命令中指定一个行号以删除某一特定行。当删除某一行时，其它的行号会在必要时重新编号以保证行号连续。下面在 DEL 命令中包含行号以删除第 5 行。

```
SQL>  DEL 5
SQL>
SQL> LIST
  1  SELECT
  2    DEPT1_1.DEPTNO,DEPT1_1.DNAME,
  3    EMP1_1.EMPNO,EMP1_1.ENAME,EMP1_1.SALARY
  4    FROM DEPT1_1,EMP1_1
  5    EMP1_1.DEPTNO = DEPT1_1.DEPTNO
  6*   ORDER BY DEPT1_1.DEPTNO,
```

9)使用 CLEAR BUFFER 命令删除 SQL 缓冲区中所有的行。

```
SQL>  CLEAR BUFFER
buffer cleared

SQL>  LIST
No lines in SQL buffer.
```

工作原理

SQL * Plus 包含了各种编辑命令以便对 SQL 缓冲区中的语句进行修改。步骤 1)使用 GET 命令将一个查询命令装入 SQL 缓冲区。使用 SQL * Plus 编辑命令只能编辑 SQL 缓冲区