《ORACLE 关系数据库系统使用说明》之二

ORACLE Utilities
SQL* Report
SQL* Loader
用户指南

吕春莲　王　珙　胡　杉

国家统计局计算中心

《ORACLE 关系数据库系统使用说明》之二

ORACLE Utilities
SQL* Report
SQL* Loader
用户指南

吕春莲　王　珙　胡　杉

国家统计局计算中心

前　言

ORACLE RDBMS是美国Oracle公司的关系型数据库管理系统产品，它采用SQL（美国国家标准学会——ANSI于1986年10月16日正式颁布SQL语言为美国国家标准）作为数据库查询语言，拥有60多种大中型机、小型机、微机上的运行版本，包含有报表生成、全屏幕表格、高级语言接口等多种实用工具软件产品，是一种较好的高性能关系型数据库管理系统，其1986年8月宣布排出的版本5.1产品又具有了初步的分布式关系数据库管理系统的先进功能。

本套《ORACLE关系数据库系统使用说明》资料系我们对绝大多数ORACLE RDBMS产品说明书进行编译后形成，它包括了ORACLE产品的以下内容：

1. ORACLE RDBMS Kernel
   - Basic Utilities
   - SQL*Report
   - SQL*Loader
2. ORACLE SQL*Plus
3. ORACLE SQL*Forms
4. ORACLE SQL*Calc
5. ORACLE Pro*C
6. ORACLE Pro*Fortran
7. UNIX系统的ORACLE安装与用户指南
8. MS-DOS的ORACLE安装与用户指南
9. ORACLE Report*Writer
10. ORACLE SQL*Net

本套资料由国家统计局计算中心组织编译，参加工作的有：吕春莲、王延华、孔祥清、胡杉、王珙、谢坤等同志。

本套资料供国家统计局系统内部使用。

本册为ORACLE Utilities、SQL*Report、SQL*Loader用户指南。

A5107/02

Utilities 序言

本指南的宗旨

本指南介绍了ORACLE关系型数据库管理系统的两个实用程序：

1．ODL程序：该实用程序为ORACLE用于将外部的操作系统数据文件中的数据加载到ORACLE表中，称为ORACLE数据加载程序，简称ODL。

2．EXPORT/IMPORT程序：EXPORT实用程序（卸出程序）用于将ORACLE表中数据转贮成操作系统文件，IMPORT实用程序（装入程序）用于将通过Export（卸出程序）输出的文件中之数据再次加载到ORACLE数据库中。

全书论述了对被加载文件的要求，讲述了被卸出文件以及数据加载、卸出和装入的控制。每一个被生成的或被使用的文件的例子，装入、卸出的例子以及ODL操作在本书中均有详细说明。

本书的对象

本书是为有以下要求的读者编写的：

1．从外部操作系统文件向ORACLE数据库加载数据（使用ODL）

2．从ORACLE数据库中卸出数据和向ORACLE数据库装入数据（使用装入、卸出实用程序）

读者需具有运行SQL语言的知识以及ORACLE数据库的基础知识。另外，使用ODL要求读者能建立和编辑ODL需要使用的操作系统文件和数据文件。

全书的结构

本书包括二章

第一章　ORACLE数据加载器（ODL）

第二章　卸出和装入（EXPORT和IMPORT）

附录中罗列了运行ODL可能出现的错误信息。

相关出版物：

阅读本书的过程中，读者需要参考以下书籍

1．ORACLE维护发行版说明书

2．《ORACLE数据库管理员手册》

3．《错误信息和代码》

目 录

第一章 ORACLE数据加载器（ODL）

1.1 ODL的特性 …… (1)
1.2 ODL处理的概况 …… (1)
1.3 ODL所不能实现的操作 …… (2)
1.4 ODL的先行条件 …… (3)
1.5 ODL命令 …… (3)
1.6 控制文件 …… (4)
1.6.1 定义记录语句（DEFINE RECORD语句） …… (5)
1.6.2 DEFINE SOURCE语句 …… (6)
1.6.3 FOR EACH语句 …… (7)
1.7 原始数据文件 …… (8)
1.8 登记文件 …… (9)
1.9 坏文件 …… (10)
1.10 ODL信息 …… (10)
1.11 ODL提示 …… (11)
1.11.1 加载DATE（日期型）数据 …… (11)
1.11.2 范围检验或值检验 …… (11)

第二章 卸出与装入（EXPORT/IMPORT）

2.1 EXPORT/IMPORT的特点 …… (12)
2.2 EXPORT/IMPORT不能实现的功能 …… (12)
2.3 错误处理 …… (12)
2.4 卸出（EXPORT）程序 …… (13)
2.4.1 卸出模式 …… (13)
2.4.2 谁能操作卸出 …… (13)
2.5 卸出命令 …… (13)
2.6 以全数据库模式卸出 …… (15)
2.6.1 全数据库模式卸出举例 …… (15)
2.7 USER（用户）模式卸出 …… (15)
2.7.1 用户模式卸出举例 …… (15)
2.8 TABLES（表）模式卸出 …… (16)
2.8.1 表模式卸出举例 …… (16)

此为试读，需要完整PDF请访问：www.ertongbook.com

2.9 装入（IMPORT）程序……（17）
2.9.1 谁可实施装入 ……（18）
2.10 装入命令……（18）
2.10.1 装入操作举例……（19）
2.11 装入由第五版以前ORACLE生成的卸出文件 ……（20）
2.12 装入与卸出提示……（21）
2.12.1 手工生成表……（21）
2.12.2 压缩选择……（21）
2.12.3 装入或卸出LONG（长字符）型数据……（21）
2.12.4 装入或卸出大表……（21）
附录A：ODL错误信息 ……（22）

第一章 ORACLE 数据加载器(ODL)

ORACLE数据加载器（简称ODL）是ORACLE用于将操作系统文件中的原始数据加载到ORACLE数据库表中的实用程序。在执行ODL时，数据库中必须有相应的表存在，ODL程序本身并不生成表。

ORACLE还有另外一个数据加载工具SQL*Loader，请参看相应用户指南。

1.1 ODL的特性

使用ODL可完成以下工作：

1．将一个文件中的所有数据加载到ORACLE表中

2．从一个数据文件中有选择地加载一部分数据

3．选择一张表中的一些列加载（这些列均为空）

4．利用一个函数为每一条记录生成唯一的整数标识

5．加载指定条数的记录

6．开始加载前跳过一定数量的记录

7．指明在产生了一定数量的错误后是继续还是停止加载

1.2 ODL处理的概况

为了运行ODL程序，用户只需简单地键入“ODL”以及一些参数。所有ODL运行需要的信息均在命令行或命令行上的文件中给出。图1表示ODL的执行过程。

在输入ODL命令时，用户需要给出以下两个文件：

1．控制文件，这一文件中包含了加载时所需的一般目录，它识别原始数据文件和接受加载的ORACLE中的表及表中的列。同时，它将文件中的数字段映象到ORACLE表中的列。

2．原始数据文件：该文件包含了要加载的数据。一个控制文件可加载一个或多个数据文件，数据文件各称由控制文件给出。

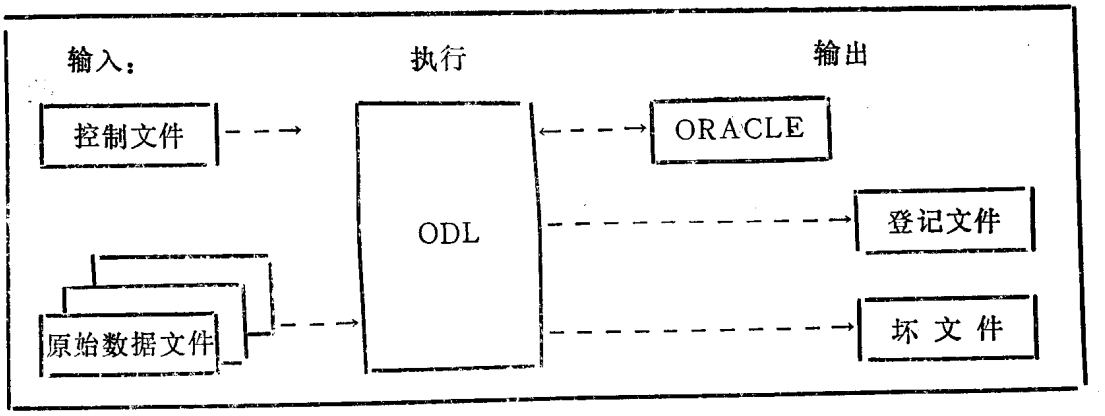
当ODL命令执行时，ODL按照控制文件的结构，读原始数据文件，利用SQL的INSERT语句加载数据到已被指定的ORACLE表中。为跟踪ODL的处理过程，ODL运行时会产生一个或二个文件：

1．登记文件（Log FiLE）：ODL运行时会生成一个登记文件，该文件中包含了被读到的加载成功的、跳过的记录数以及错误信息。

2．坏文件(Bad File)：ODL运行时，如遇到坏的（即不正确的）记录时，就会自动

地生成一个坏文件。例如：ODL 运行时遇到了相对于被加载列来说为非法数据）的如企图将字符型数据加载到数字型的列上），ODL就将该记录写入坏文件，然后继续加载下一条记录。坏文件可在以后加以编辑以更正错误的记录，并重新加载。

图 1　ODL操作图解



1.3　ODL所不能实现的操作

ODL不可实现以下操作

1．在运行一个控制文件的过程中向一张以上的表加载数据

2．对于在数据文件中占有多于一个物理记录上的记录进行加载

例如按下表格式所写的原始数据文件可被ODL加载

记录 1
记录 2
记录 3
……

但下列格式的原始文件不可加载

记录 1
……记录 1 的继续
记录 2
……记录 2 的继续
记录 3
……记录 3 的继续
……

1.4 ODL的先行条件

1．ODL不象IMPORT命令，它不能为用户产生表，因此在运行ODL 前被加载的表必须已存在于数据库中。

2．运行ODL的用户仅需要对数据库具有CONNECT的存取数限（不需要RESOURCE数限），用户不必是被加载表的生成者，但必须具有对该表进行插入操作的权力。

3．用户必须提供控制文件和数据文件。控制文件名在ODL命令中引用。两个文件必须在ODL要运行时是可读的。

4．ODL要求读到的第一个被加载记录必须是有效记录，否则ODL 将中止运行。

1.5 ODL命令

当用户具有合适的输入文件，并已在ORACLE数据库中建立了要被加载的表以后，即可运行ODL，利用ODL命令的选择，用户可指明：

1．跳过x条记录开始加载（S选择）

2．每x条记录作一次提交（C选择）

3．仅加载x条记录（L选择）

4．忽略错误（即在迁到错误后继续处理）（E选择）

5．在迁到x个错误后停止操作（E选择）

这些选择在运行时指定，而不是在控制文件中指定。

因此，控制文件指定关于数据加载的一部分情况，另一些情况则在输入运行命令时指定。

ODL命令的句法由图 2 给出

图 2：ODL命令句法

```
ODL  控制文件名  登记文件名  用户标识  〔—选择项〕
```

用户必须在命令行上给出控制文件名和登记文件名。“控制文件名”虽为控制文件的名称，但它包含了原始数据文件名，因此用户不必在命令行上给出原始数据文件名。登记文件名指出这样的文件，它记录错误信息，并对加载过程进行统计。用户标识是接受加载表的拥有者的用户名和用户口令（或者该表用义词的拥有者），如在输入命令时未输入用户标识，ODL会提示用户给出。

ODL的选择项可以任意组合给出，每个选择项用正整数作为其参数。

S：跳过n条记录开始加载，缺省值为零，如加载的数据文件多于一个，则仅能在第一个文件中实现跳跃

L：加载n条记录（在跳过选择S给出的n条记录后），缺省表示加载到文件结束

n＞0 表示加载n条记录

n＝0 表示加载到文件的结束（与缺省情况同）

C：每加载n条记录后进行一次提交，缺省为每加载100条记录后进行一次提交

n＞0 表示成功地插入n条记录后进行一次提交

n＝0 表示全部加载完成后进行提交

E：在中止执行ODL之前允许出现n个错误，缺省为50，

n＞0 允许产生n个错误后停止执行

n＝0 加载过程中不考虑错误的出现。

B：用于一批数组和输入缓冲区所需的字节数，缺省为16K，一般来说，缺省值已足够了，即使用户希望加载一些特别宽的表。该值未设最大值，最大值依赖于可用内存。

下面介绍两个运行ODL的实例。第一个实例中的控制文件为JUNECONT，登记文件为JUNELOG。登记文件将包含实际加载的信息。加载时，ODL将先跳过1000条记录；如果产生了20个错误，ODL将停止执行；每加载成功100条记录后进行一次提交。这个命令输入如下：

ODL junecont. ctl junelog. log shirley/bear－S1000－E20

第二个例子要求每加载一条记录后即进行一次提交，并在整个加载过程中不考虑出现错误。

ODL Sample. ctl logsamp. log scott/tiger－C1－EQ

1.6 控制文件

ODL运行必须要求至少两个文件业已存在，一个为控制文件，另一为原始数据文件（原始数据文件可多于一个），控制文件定义了在一特定装载工作中ODL将作什么。用户在ODL命令中给出控制文件名，在控制文件中给出数据文件名。

对于控制文件来说，其扩展名或文件类型不存在缺省情况，因此用户在输入ODL命令时必须完整地给出控制文件的整个名字。

一个控制文件包括三个部分：

1．DEFIME RECORD语句（定义记录语句）

2．DEFINE SOURCE语句（定义数据源语句）

3．FOR EACH语句

图3给出了一个ODL的控制文件的例子，该例中控制文件定义了称为EMPLOYEE（就业人员）的记录，该记录包含三个字段：就业人员姓名（ENAME），级别（EGRADE）和工资（ESALARY）

图3　ODL控制文件

```
DEFINE RECORD emptoyee AS
  ename (char (20) , )
  egrade (integer ( 2 ) , )
  esatary (float ( 4 ) , loc ( +10) ; )
DEFINE SOURCE TAPE
  FROM file1, file2
  LENGTH 80
  CONTALNLNG emptoyee;
FOR EACH RECORD
  INSERT INTO empomain
   (name, performance, grade, gross
  nulcol, salary, start) VALUES
   (ename, 87.6543, tgrade, 86e - 6,
  null, esalary, 356)
NEXT RECORD
```

ENAME字段处于记录的开始，占用了20个字节，紧接着的两个字节用于存放EGRADE字段，ESALARY字段居EGRADE右边10个字节之后。输入数据来源为TAPE（磁带），其中包含有2个文件，每个文件的记录长度均为80字节。数据库表EMPMAIN中七列接受了加载，载入值或来自于就业人员这一原始数据记录中的字段或者为给定的常数。

1.6.1定义记录语句（DEFINE RECORD语句）

DEFINE RECORD语句描述了原始数据文件中的数据

1．给欲加载的记录命名

2．给欲加载的记录中的字段命名

3．给出每个字段的数据类型

4．说明记录中每个字段的位置

DEFLNE RECORD语句一般的句法结构如图4所示：

图4　DEFINE RECORD语法结构

```
DEFLNE RECORD rec-name AS fld-name (fld-type
                [, loc (fld-loc) ] )
                        fld-name (fld-type
                [, loc [fld-loc) ] ) ;
```


此为试读，需要完整PDF请访问：www.ertongbook.com

DEFLNE RECORD语句中的变量介绍

1．rec-name是一个用于与原始数据记录相对应的名称

2．fld-name是一个用于与原始数据字段相对应的名称

3．fld-type定义了数据类型，为以下三种类型中之一

{FLOAT | INTEGER | CHAR}〔（长度）〕

分别为　浮点型、　整型、　字符型

小括号中的长度为所占用字节数，最好不用缺省值。有效长度和缺省值如下所列：

数据类型	长度	机器默认值			
		RSX 11-M	VAX VMS	IBM MVS, CMS	UNIX
浮点型（TLOAT）	8，4	8	8	8	可变
整型（INTEGER）	4，2，1	2	4	4	可变
字符型（CHAR）	1到240	1	1	1	1

4．fld-loc是原始数据纪录中字段的位置，可有如下三种表示方式：

$$\text{loc}\left(\begin{vmatrix}+\text{number}\\-\text{number}\\\text{number}\end{vmatrix}\right)$$

注意，每个记录的第一个位置是第0列而不是第一列。未给出符号的整数表示从记录开始起的绝对位置，数字前的符号表示对于上一字段结尾处的相对位置（负号表示相对于字段开始的位置，正号表示相对于字段结束处的位置），缺省位置为前一字段的尾部，对一个字段来说则为纪录的开始。

如果数据库中的一列定义为NUMBER（数值）型列，而原始数据字段是一个表示成数字量值的ASCII串，则数据类型应为CHAR（字符）型，ORACLE将其转换成数值格式。

以下为一个DEFINE RECORD语句的例子

```
DEFINE RECORD REC1 AS
FLD1 (CHAR (6));
FLD2 (CHAR (25)),
FLD3 (FLOAT, LOC (-3));
```

1.6.2 DEFINE SOURCE 语句

DEFINE SOURCE 语句标明了输入媒介（诸如文件或磁带）以及在这一媒介中包含的记录，句法如图5所示：

图5　DEFINE SOURCE语句的语法

```
DEFINE SOURCE       Src-name
       FROM         filename [, filename……, filename]
       LENGTH       length
       CONTAINING   rec-name;
```

语句中的参数说明如下：

src-name是源文件的一个象征性名称，它可以是任何标识符。一般使用FILE（文件）或TAPE（磁带）作为此参数的选择

filename是含有原始数据的操作系统文件的文件名，可以用逗号将多个串联的文件分隔开来。

langth是原始数据的记录长度。该记录长度表示的是原始数据记录的最大长度。如果记录超出给定的记录长度，ODL运行时会提示如下出错信息：

Read error, Skipping to next file

（读到错误，跳到下一文件）

同时ODL继续执行。如果最后一列是变长的或变长数据项后所有列均为NULL（空），那么该变长记录可以被正常地加载。逻辑记录的长度并不依赖于文件属性，但与物理记录的长度一致。例如：RSX-11M和VAX/VMS允许文件具有隐含的回车。如原文件是一个记录用定长格式写成的变长记录文件，那么用户就不必加LOC去跳过该记录之前的空白字节。ODL可以通过计算字段位置来识别字节数，并使用实际上的逻辑记录长度。

rec-name是用在DEFINE RECORD语句中的记录名。

以下为一个DEFINE SOURCE语句的例子：

```
DEFINE SOURCE    SRC 1
          FROM FILE 1, FILE 2, FILE 3
          CENGTH  80
          CONTAINING  REC 1;
```

1.6.3 FOR EACH语句

FOR EACH 语句将原始数据的每一记录读出，并插入到指定的表中。该语句的句法如图6所示。

图6：FOR EACE语句的句法

```
FOR EACH RECORD INSERT INTO table-name
                    (col-name, col-name…)
VALUES (fld-name, fld-name…)
NEXT RECORD
```

标准的SQL INSERT语句是对每一个记录执行的，
FOR EACH语句其实就是一个如下形式的标准SQL扦入语句。

```
INSERT INTO tab-name
 (Col-name, ……, Col-name) VALUES (src-dafa, …)
```

其中，tab-name是用户所拥有的表名或是该用户为某些其他表所建立的同义词。

表名为SCOTT.DERT的表不可加载。为加载数据到其他用户的表中，你可利用自己对他人表已拥有INSERT（插入）权力的表的同义词来实现。你还必须说明要被插入的列名并使用关键词 VALUES，然后说明用逗号分隔的原始数据字段名或数据的值。以下列出的任何一种数均可作为数据被扦入到表中：

1．原始数据文件中的字段名
2．字符串
3．数值（可为数字串或以科学计数法表示）
4．NULL（空格）
5．GENSEQ函数（GENSEQ函数用于对每个记录生成新的整数值，格式为：
GENSEQ (initial_value, increment_value))

（初值）　　（增量）

例如在下一例中

```
FOR EACH RECORD
INSERT INTO CODE-TABLE (GENKEY, DESCRIPTION) VALUES
                (GENSEQ (100, 5 ), PURPOSE)
NEXT RECORD
```

其中，CODE-TABLE表中的列DESCRIPTION用PURPOSE 字段中的值加载；而对GENKEY则由加载时GENSEQ 函数产生的唯一值给定，第一个值是100，以后每次增加5，因此第二个为105，第三个为110，如此一直增加到加载结束为止。

NEXT RECORD短语用于结束扦入循环。

以下给出一个FOR EACH语句的例子

```
FOR EACH RECORD
INSERT INTO EMP (EMPNO, EMPNAME, SALARY, COMM) VA-
LUES
                FLD1, FLD2, FLD3, NULL)
NEXT RECORD
```

1.7 原始数据文件

一个DDL控制文件可调用一个或多个数据文件。数据文件可以是由操作系统定义的任何记录格式，但有如下要求：

1．记录可为定长或变长。

2．文件中的每一个物理记录对应于一个新的逻辑记录。

3．ODL在数据文件中读到的第一条记录必须是有效的，否则ODL将中止运行。如用户要求ODL跳过前100条记录后开始加载，则第101条记录必须是有效的。

4．记录中的每个字段必须是定长的（最后一个字段除外）。

5．最后一个字段可为变长。如为变长，则在插入到指定表中之前，数据将被补充空格。

6．并不是原始记录的所有字段均须被加载，控制文件中可以规定跳过一些列。

7．原始记录中的数据列并不一定要与表中的列相对应，但必须与控制文件中的DEFINE RECORD以及FOR EACH…INSERT语句中的列对应。

8．支持的数据类型有：①字符串　　②二进制整数　　③浮点数

ODL现在还不能接受压缩十进制数。

用户可以从你所用操作系统的《ORALLE安装和用户指南》中找到其他一些关于原始数据文件的信息。

1.8 登记文件

当ODL进行一次加载时，它将生成一个登记文件(log file)，该登记文件记录了加载时所统计的记录数目及加载时产生的错误信息。登记文件名在调用ODL时给出，对于登记文件的扩展名及其文件类型无缺省情况，所有这些均需由用户给出（不能使用BAD作为扩展名或文件类型，因为ODL在运行时将自动使用BAD作为坏文件的扩展名或文件类型）

登记文件中有关于多少条记录从缓冲区中被加载的信息，一旦运行ODL后，用户可利用这个反馈信息去调整缓冲区的大小，以增加或减少每次扦入的行数。通过调整缓冲区增加其所存入的行数，可加快ODL加载速度。图7给出了一个登记文件的例子（见下页）。

如ODL在原始数据文件的第一条记录读到错误，运行将立即中止，无记录被加载，登记文件中出现如下的错误信息：

```
EXEC ERROR: ORA-1438: value larger than specified
                    precision allows for this column
RECORD 1 REJECTED
LOGGED OUT FROM ORACLE
```

图7　登记文件例子

```
ORACLE DATA LOADER: Version 5.0.12 on Wed Sep 25 11:35:03
1985
Copyright (c) 1985, oracle corporafion, califormia, USA. ALL rights re-
served.
LOGGED INTO ORACLE V5.0.12-Beta
         16384  TOTAL BIND SPACE
           654  MAXIMUM BIND ARRAY DIMENSION
         16368  MAXIMUM BIND ARRAY SIZE IN BYTES
           654  ACTUAL BIND ARRAY DIMENSION LIMITED
                                        BY MEMORY
         16368  ACTUAL BIND ARRAY
                          SIZE IN BYTES
  LOGGED OUT FROM ORACLE
  STATISTICS
               574  BYTES ALLOCATED
                 0  RECORDS SKIPPED
               254  RECORDS READ
                 0  RECORDS REJECTED
               254  ROWS LOADED
                 0  ERRORS
  END ORACLE DATA LOADER Wed Sep 25 11:35:13 1985
```

1.9　坏文件

如果遇到原始数据文件的记录不能被成功地扦入，ODL将自动产生一个坏文件（bad file）以保存丢掉的记录。

一条输入记录如未满足控制文件中的定义，它就要被丢掉，用户可在以后编辑这个坏文件，更正错误，然后利用同样的ODL控制文件重新加载这些记录。

坏文件的文件名由ODL自动地根据用户所给的登记文件产生，其文件名为登记文件名，后缀为BAD。在下面的一个ODL操作中，如有坏的记录产生，则将生成一个名为IM.BAD的坏文件。

```
ODL SURVEY.CTL IM.LOG RODNEY/HERRLNG
```

1.10　ODL信息

ODL的处理信息都写在登记文件中，其中包括下列统计信息：

1．所读的记录数 (number of records read)

2．跳过的记录数 (number of records skipped)

3．加载的记录数 (number of records loaded)

4．丢掉的记录数 (number of records rejected)

当处理记录过程中出现错误时，ODL就产生如下的一个信息（其中#是正被读的记录号）

RECORD ＜#＞ REJECTED （第#条记录被丢掉）

1.11 ODL提示

1.11.1 加载DATE（日期型）数据

为了加载DATE型数据，首先必须将它加载到一张临时表中的定义为CHAR字符型的列中，然后利用一条含有嵌套选择的INSERT语句，将这些行杆从临时表中选出、插入到永久性表中（或者利用一条含有嵌套选择的CREAT TABLE命令），再利用TO-DATE函数将字符型日期转换成标准的ORACLE DATE型数据格式

1.11.2 范围检验或值检验

在数据加载时，为了增强数据的有效性，可以做一些检验工作。例如进行数据范围检验，首先将数据加载到一张临时表中，然后再利用一条含有嵌套选择的INSERT语句（或者用包含嵌套选择的CREATE TABLE命令），将该数据放入永久性表中。此时，嵌套选择的WHERE子句必须指定数据检验，例如：

```
… WHERE COL 1 in ('A', 'B', 'C') and
         COL 2 BETWEEN 1 AND 10…
```


此为试读，需要完整PDF请访问：www.ertongbook.com