File Edit View History Bookmarks Tools Help

Input variables

www.pgrocer.net/Cis50/orainvar.html

```
SQL> select idno, name from donor where idno >= &donorid;
Enter value for donorid: 12121
old   1: select idno, name from donor where idno >= &donorid
new   1: select idno, name from donor where idno >= 12121

IDNO  NAME
----- ---------------
12121 Jennifer Ames
22222 Carl Hersey
23456 Susan Ash
33333 Nancy Taylor
34567 Robert Brooks
```

In the third and fouth examples, I want character/string data and so I need to enter quotes either arou

## SQL CODE:

```
SQL> SELECT idno, name, state
  2  FROM donor
  3  WHERE state ='&statein';
Enter value for statein: MA
old   3: WHERE state ='&statein'
new   3: WHERE state ='MA'

IDNO  NAME            ST
----- --------------- --
11111 Stephen Daniels MA
23456 Susan Ash       MA
33333 Nancy Taylor    MA
34567 Robert Brooks   MA

SQL> SELECT idno, name, state
  2  FROM donor
  3  WHERE state = &statein;
Enter value for statein: 'MA'
old   3: WHERE state = &statein
new   3: WHERE state = 'MA'

IDNO  NAME            ST
----- --------------- --
11111 Stephen Daniels MA
23456 Susan Ash       MA
33333 Nancy Taylor    MA
34567 Robert Brooks   MA
```

In the fifth example, I am showing you that functions can be used with this. In this case I am going to look for the initcap version of the city.

## SQL CODE:

```
SQL> SELECT idno, name, city, state
  2  FROM donor
  3  WHERE city = INITCAP('&cityin');
Enter value for cityin: seekonk
old   3: WHERE city = INITCAP('&cityin')
new   3: WHERE city = INITCAP('seekonk')
```

pgrocer@cisweb:~

SMART Ink

```
IDNO  NAME
----- ---------------
12121 Jennifer Ames

SQL> select idno, name, state from donor
  2  where state = &statein;
Enter value for statein: 'MA'
old   2: where state = &statein
new   2: where state = 'MA'

IDNO  NAME            ST
----- --------------- --
11111 Stephen Daniels MA
23456 Susan Ash       MA
33333 Nancy Taylor    MA
34567 Robert Brooks   MA

SQL> select idno, name, state from donor
  2  where state = '&statein';
Enter value for statein: MA
old   2: where state = '&statein'
new   2: where state = 'MA'

IDNO  NAME            ST
----- --------------- --
11111 Stephen Daniels MA
23456 Susan Ash       MA
33333 Nancy Taylor    MA
34567 Robert Brooks   MA

SQL>
```

Type here to search

9:42 AM
10/1/2019

```
IDNO  NAME            ST
----- --------------- --
11111 Stephen Daniels MA
23456 Susan Ash       MA
33333 Nancy Taylor    MA
34567 Robert Brooks   MA
```

In the fifth example, I am showing you that functions can be used with this. In this case I am going to look for the initcap version of the city.

### SQL CODE:

```
SQL> SELECT idno, name, city, state
  2  FROM donor
  3  WHERE city = INITCAP('&cityin');
Enter value for cityin: seekonk
old   3: WHERE city = INITCAP('&cityin')
new   3: WHERE city = INITCAP('seekonk')

IDNO  NAME            CITY       ST
----- --------------- ---------- --
11111 Stephen Daniels Seekonk    MA
```

In the sixth example, I again asked for a specific idno. Since I know that idno was defined as a varchar2, I decided to use the quotes to take in the variable idno. It worked this way as well.

### SQL CODE:

```
SQL> SELECT idno, name, state
  2  FROM donor
  3  WHERE idno = '&donorid';
Enter value for donorid: 22222
old   3: WHERE idno = '&donorid'
new   3: WHERE idno = '22222'

IDNO  NAME            ST
----- --------------- --
22222 Carl Hersey     RI
```

If you don't want to see the old, new information, you can use the SET VERIFY OFF command and if you want to see it again you can change with SET VERIFY ON.

### SQL CODE:

```
SQL> SET VERIFY OFF
SQL> SELECT idno, name, city, state
  2  FROM donor
  3  WHERE city = INITCAP('&cityin');
Enter value for cityin: providence

IDNO  NAME            CITY       ST
----- --------------- ---------- --
12121 Jennifer Ames   Providence RI
22222 Carl Hersey     Providence RI
```

This concept can be carried even further. For example I can substitute in column names, the table name and the entire condition. In the second example, I even substituted in the field to order by and in the third example I substituted in everything but the

```
new    3: WHERE state = 'MA'
Enter value for ordname: yrgoal
old    4: ORDER BY &ordname
new    4: ORDER BY yrgoal

IDNO  NAME               YRGOAL
----- --------------- ---------
33333 Nancy Taylor           50
34567 Robert Brooks          50
23456 Susan Ash             100
11111 Stephen Daniels       500

SQL> SELECT &selname;
Enter value for selname: idno, name, yrgoal from donor where yrgoal > 50
old    1: SELECT &selname
new    1: SELECT idno, name, yrgoal from donor where yrgoal > 50

IDNO  NAME               YRGOAL
----- --------------- ---------
11111 Stephen Daniels       500
12121 Jennifer Ames         400
23456 Susan Ash             100
```

If you want to define a field so you can reuse the variable name without prompting, use the &&.

**SQL CODE:**

```
SQL> SELECT idno, name, &&col
  2  FROM donor
  3  WHERE &&col = 'MA';
Enter value for col: state
old    1: SELECT idno, name, &&col
new    1: SELECT idno, name, state
old    3: WHERE &&col = 'MA'
new    3: WHERE state = 'MA'

IDNO  NAME            ST
----- --------------- --
11111 Stephen Daniels MA
23456 Susan Ash       MA
33333 Nancy Taylor    MA
34567 Robert Brooks   MA

SQL> SELECT idno, name, city, &&col
  2  FROM donor;
old    1: SELECT idno, name, city, &&col
new    1: SELECT idno, name, city, state

IDNO  NAME            CITY       ST
----- --------------- ---------- --
11111 Stephen Daniels Seekonk    MA
12121 Jennifer Ames   Providence RI
22222 Carl Hersey     Providence RI
23456 Susan Ash       Fall River MA
33333 Nancy Taylor    Fall River MA
34567 Robert Brooks   Fall River MA

6 rows selected.
```

pgrocer@cisweb:~

```
  2  from donor
  3  where &&col = 'MA';
Enter value for col: state
old    1: select idno, name, &&col
new    1: select idno, name, state
old    3: where &&col = 'MA'
new    3: where state = 'MA'

IDNO  NAME            ST
----- --------------- --
11111 Stephen Daniels MA
23456 Susan Ash       MA
33333 Nancy Taylor    MA
34567 Robert Brooks   MA

SQL> select idno, &&col from donor;
old    1: select idno, &&col from donor
new    1: select idno, state from donor

IDNO  ST
----- --
11111 MA
12121 RI
22222 RI
23456 MA
33333 MA
34567 MA

6 rows selected.

SQL>
```

Title: Oct 1-9:52 AM (3 of 25)

File Edit View History Bookmarks Tools Help

Input variables

www.pgrocer.net/Cis50/orainvar.html

```
----- --------------- ---------
33333 Nancy Taylor           50
34567 Robert Brooks          50
23456 Susan Ash             100
11111 Stephen Daniels       500

SQL> SELECT &selname;
Enter value for selname: idno, name, yrgoal from donor where yrgoal > 50
old   1: SELECT &selname
new   1: SELECT idno, name, yrgoal from donor where yrgoal > 50

IDNO  NAME                YRGOAL
----- --------------- ---------
11111 Stephen Daniels       500
12121 Jennifer Ames         400
23456 Susan Ash             100
```

If you want to define a field so you can reuse the variable name without prompting, use the &&.

### SQL CODE:

```
SQL> SELECT idno, name, &&col
  2  FROM donor
  3  WHERE &&col = 'MA';
Enter value for col: state
old   1: SELECT idno, name, &&col
new   1: SELECT idno, name, state
old   3: WHERE &&col = 'MA'
new   3: WHERE state = 'MA'

IDNO  NAME            ST
----- --------------- --
11111 Stephen Daniels MA
23456 Susan Ash       MA
33333 Nancy Taylor    MA
34567 Robert Brooks   MA

SQL> SELECT idno, name, city, &&col
  2  FROM donor;
old   1: SELECT idno, name, city, &&col
new   1: SELECT idno, name, city, state

IDNO  NAME            CITY       ST
----- --------------- ---------- --
11111 Stephen Daniels Seekonk    MA
12121 Jennifer Ames   Providence RI
22222 Carl Hersey     Providence RI
23456 Susan Ash       Fall River MA
33333 Nancy Taylor    Fall River MA
34567 Robert Brooks   Fall River MA

6 rows selected.
```

To clear a reusable variable you can use the UNDEFINE statement.

### SQL CODE:

```
SQL> undefine col;
```

Type here to search

9:52 AM
10/1/2019

Title: Oct 1-9:52 AM (4 of 25)

File Edit View History Bookmarks Tools Help

More variable information...

www.pgrocer.net/Cis50/oradefac.html

```
SQL> SELECT *
  2  FROM inven
  3  ORDER BY &ordercol;
Enter value for ordercol: itemname
old   3: ORDER BY &ordercol
new   3: ORDER BY itemname

ITEM ITEMNAME           ONHAND   ONORDER   REORDPT      COST     PRICE DE IT LOCA
---- --------------- --------- --------- --------- --------- --------- -- -- ----
1234 Adven Reddy Fox         5         0        10         9     14.75 BK CH X100
3333 Basketball             24        25        50        14     17.99 SP BK Y200
2222 Building Blocks         4         0        15        23     27.98 TY CH Z200
2345 Doll House              2         5        10        45     55.98 TY CH Z212
1111 Good Night Moon        24        30        40         8     12.99 BK BY X100
1212 Heidi                  12        25        25        10     14.99 BK CH X112
3456 Net/Hoop               12         0        25        25     27.95 SP BK Y200
2121 Teddy Bear              5        20        40        15     19.95 TY CH X115

8 rows selected.

SQL> SELECT *
  2  FROM inven
  3  ORDER BY &&orderhold;
Enter value for orderhold: price
old   3: ORDER BY &&orderhold
new   3: ORDER BY price


ITEM ITEMNAME           ONHAND   ONORDER   REORDPT      COST     PRICE DE IT LOCA
---- --------------- --------- --------- --------- --------- --------- -- -- ----
1111 Good Night Moon        24        30        40         8     12.99 BK BY X100
1234 Adven Reddy Fox         5         0        10         9     14.75 BK CH X100
1212 Heidi                  12        25        25        10     14.99 BK CH X112
3333 Basketball             24        25        50        14     17.99 SP BK Y200
2121 Teddy Bear              5        20        40        15     19.95 TY CH X115
3456 Net/Hoop               12         0        25        25     27.95 SP BK Y200
2222 Building Blocks         4         0        15        23     27.98 TY CH Z200
2345 Doll House              2         5        10        45     55.98 TY CH Z212

8 rows selected.

SQL> SELECT itemno, itemname, cost, price
  2  FROM inven
  3  ORDER BY &&orderhold;
old   3: ORDER BY &&orderhold
new   3: ORDER BY price

ITEM ITEMNAME             COST     PRICE
---- --------------- --------- ---------
1111 Good Night Moon         8     12.99
1234 Adven Reddy Fox         9     14.75
1212 Heidi                  10     14.99

3333 Basketball             14     17.99
2121 Teddy Bear             15     19.95
3456 Net/Hoop               25     27.95
2222 Building Blocks        23     27.98
2345 Doll House             45     55.98
```

Type here to search

10:02 AM 10/1/2019

Title: Oct 1-10:02 AM (5 of 25)

File Edit View History Bookmarks Tools Help

More variable information...

www.pgrocer.net/Cis50/oradefac.html

```
  2  FROM inven
  3  ORDER BY &&orderhold;
old   3: ORDER BY &&orderhold
new   3: ORDER BY price

ITEM ITEMNAME              COST     PRICE
---- --------------- --------- ---------
1111 Good Night Moon          8     12.99
1234 Adven Reddy Fox          9     14.75
1212 Heidi                   10     14.99

3333 Basketball              14     17.99
2121 Teddy Bear              15     19.95
3456 Net/Hoop                25     27.95
2222 Building Blocks         23     27.98
2345 Doll House              45     55.98

8 rows selected.
```

You can also use the DEFINE and ACCEPT commands in SQL*Plus to predefine variables. The DE ... AR datatype field while the DEFINE variable with no value assigned to it simply displays the value and its datatype. The ACCEPT takes user input and ... ]). Notice that everything is optional accept the variable. The datatype can be NUMBER, CHAR or DATE. The HIDE suppresses the user entry as i ... Here are some examples.

In this first example, I used &&orderhold to define the variable. The second SQL statement is this e ... st and it is a CHAR field.

pgrocer@cisweb:~

```
Enter value for col: state
old   1: select idno, name, &&col
new   1: select idno, name, state
old   3: where &&col = 'MA'
new   3: where state = 'MA'

IDNO  NAME            ST
----- --------------- --
11111 Stephen Daniels MA
23456 Susan Ash       MA
33333 Nancy Taylor    MA
34567 Robert Brooks   MA

SQL> select idno, &&col from donor;
old   1: select idno, &&col from donor
new   1: select idno, state from donor

IDNO  ST
----- --
11111 MA
12121 RI
22222 RI
23456 MA
33333 MA
34567 MA

6 rows selected.

SQL> define col;
DEFINE COL             = "state" (CHAR)
SQL>
```

### SQL CODE:

```
SQL> SELECT *
  2  FROM inven
  3  ORDER BY &&orderhold;
Enter value for orderhold: cost
old   3: ORDER BY &&orderhold
new   3: ORDER BY cost

ITEM ITEMNAME           ONHAND   ONORDER   REORDPT      COST     PRICE DE IT LOCA
---- --------------- --------- --------- --------- --------- --------- -- -- ----
1111 Good Night Moon        24        30        40         8     12.99 BK BY X100
1234 Adven Reddy Fox         5         0        10         9     14.75 BK CH X100
1212 Heidi                  12        25        25        10     14.99 BK CH X112
3333 Basketball             24        25        50        14     17.99 SP BK Y200
2121 Teddy Bear              5        20        40        15     19.95 TY CH X115
2222 Building Blocks         4         0        15        23     27.98 TY CH Z200
3456 Net/Hoop               12         0        25        25     27.95 SP BK Y200
2345 Doll House              2         5        10        45     55.98 TY CH Z212

8 rows selected.

SQL> DEFINE orderhold;
DEFINE ORDERHOLD       = "cost" (CHAR)
```

In this example, I am putting a value into the variable deptname. The value I am putting in is TY. I then ask that it show me the contents of the deptname just to double check. I then use deptname in a select.

### SQL CODE:

Type here to search

10:04 AM
10/1/2019

File Edit View History Bookmarks Tools Help

More variable information...

www.pgrocer.net/Cis50/oradefac.html

```
  2  FROM inven
  3  ORDER BY &&orderhold;
old   3: ORDER BY &&orderhold
new   3: ORDER BY price

ITEM ITEMNAME             COST     PRICE
---- --------------- --------- ---------
1111 Good Night Moon         8     12.99
1234 Adven Reddy Fox         9     14.75
1212 Heidi                  10     14.99

3333 Basketball             14     17.99
2121 Teddy Bear             15     19.95
3456 Net/Hoop               25     27.95
2222 Building Blocks        23     27.98
2345 Doll House             45     55.98

8 rows selected.
```

You can also use the DEFINE and ACCEPT commands in SQL*Plus to predefine variables. The DE … AR datatype field while the DEFINE variable with no value assigned to it simply displays the value and its datatype. The ACCEPT takes user input and … ]). Notice that everything is optional accept the variable. The datatype can be NUMBER, CHAR or DATE. The HIDE suppresses the user entry as i… . Here are some examples.

In this first example, I used &&orderhold to define the variable. The second SQL statement is this ex… st and it is a CHAR field.

```
pgrocer@cisweb:~
BK Y200


ITEM ITEMNAME            ONHAND    ONORDER    REORDPT       COST      PRICE DE
---- --------------- ---------- ---------- ---------- ---------- ---------- --
IT LOCA
-- ----
2345 Doll House               2          5         10         45      55.98 TY
CH Z212

2222 Building Blocks          4          0         15         48      51.99 TY
CH Z200


8 rows selected.

SQL> define;
DEFINE _DATE           = "01-OCT-19" (CHAR)
DEFINE _CONNECT_IDENTIFIER = "orcl12c" (CHAR)
DEFINE _USER           = "PGROCER" (CHAR)
DEFINE _PRIVILEGE      = "" (CHAR)
DEFINE _SQLPLUS_RELEASE = "1201000100" (CHAR)
DEFINE _EDITOR         = "/usr/bin/nano" (CHAR)
DEFINE _O_VERSION      = "Oracle Database 12c Enterprise Edition Release 12.1.0.
1.0 - 64bit Production
With the Partitioning, OLAP, Advanced Analytics and Real Application Testing opt
ions" (CHAR)
DEFINE _O_RELEASE      = "1201000100" (CHAR)
DEFINE COL             = "state" (CHAR)
DEFINE ORDERHOLD       = "cost" (CHAR)
SQL>
```

## SQL CODE:

```
SQL> SELECT *
  2  FROM inven
  3  ORDER BY &&orderhold;
Enter value for orderhold: cost
old   3: ORDER BY &&orderhold
new   3: ORDER BY cost

ITEM ITEMNAME           ONHAND   ONORDER   REORDPT      COST     PRICE DE IT LOCA
---- --------------- --------- --------- --------- --------- --------- -- -- ----
1111 Good Night Moon        24        30        40         8     12.99 BK BY X100
1234 Adven Reddy Fox         5         0        10         9     14.75 BK CH X100
1212 Heidi                  12        25        25        10     14.99 BK CH X112
3333 Basketball             24        25        50        14     17.99 SP BK Y200
2121 Teddy Bear              5        20        40        15     19.95 TY CH X115
2222 Building Blocks         4         0        15        23     27.98 TY CH Z200
3456 Net/Hoop               12         0        25        25     27.95 SP BK Y200
2345 Doll House              2         5        10        45     55.98 TY CH Z212

8 rows selected.

SQL> DEFINE orderhold;
DEFINE ORDERHOLD       = "cost" (CHAR)
```

In this example, I am putting a value into the variable deptname. The value I am putting in is TY. I then ask that it show me the contents of the deptname just to double check. I then use deptname in a select.

## SQL CODE:

Type here to search

10:07 AM
10/1/2019

Title: Oct 1-10:07 AM (7 of 25)

File Edit View History Bookmarks Tools Help

More variable information...

www.pgrocer.net/Cis50/oradefac.html

```
  2  FROM inven
  3  ORDER BY &&orderhold;
old   3: ORDER BY &&orderhold
new   3: ORDER BY price

ITEM ITEMNAME             COST     PRICE
---- --------------- --------- ---------
1111 Good Night Moon         8     12.99
1234 Adven Reddy Fox         9     14.75
1212 Heidi                  10     14.99

3333 Basketball             14     17.99
2121 Teddy Bear             15     19.95
3456 Net/Hoop               25     27.95
2222 Building Blocks        23     27.98
2345 Doll House             45     55.98

8 rows selected.
```

You can also use the DEFINE and ACCEPT commands in SQL*Plus to predefine variables. The DEFINE variable with a value assigned to it (Syntax: DEFINE variable = value) puts user input into a CHAR datatype field while the DEFINE variable with no value assigned to it simply displays the value and its datatype. The ACCEPT takes user input and stores it in the variable (Syntax: ACCEPT variable [datatype][FORMAT format][PROMPT text][Hide]). Notice that everything is optional accept the variable. The datatype can be NUMBER, CHAR or DATE. The HIDE suppresses the user entry as in the entry of a password - HIDE is only available in later versions, not in the test version you are using. Here are some examples.

In this first example, I used &&orderhold to define the variable. The second SQL statement is this example simply does a DEFINE on orderhold. It comes back with the fact that orderhold is defined as cost and it is a CHAR field.

### SQL CODE:

```
SQL> SELECT *
  2  FROM inven
  3  ORDER BY &&orderhold;
Enter value for orderhold: cost
old   3: ORDER BY &&orderhold
new   3: ORDER BY cost

ITEM ITEMNAME           ONHAND   ONORDER   REORDPT      COST     PRICE DE IT LOCA
---- --------------- --------- --------- --------- --------- --------- -- -- ----
1111 Good Night Moon        24        30        40         8     12.99 BK BY X100
1234 Adven Reddy Fox         5         0        10         9     14.75 BK CH X100
1212 Heidi                  12        25        25        10     14.99 BK CH X112
3333 Basketball             24        25        50        14     17.99 SP BK Y200
2121 Teddy Bear              5        20        40        15     19.95 TY CH X115
2222 Building Blocks         4         0        15        23     27.98 TY CH Z200
3456 Net/Hoop               12         0        25        25     27.95 SP BK Y200
2345 Doll House              2         5        10        45     55.98 TY CH Z212

8 rows selected.

SQL> DEFINE orderhold;
DEFINE ORDERHOLD       = "cost" (CHAR)
```

In this example, I am putting a value into the variable deptname. The value I am putting in is TY. I then ask that it show me the contents of the deptname just to double check. I then use deptname in a select.

### SQL CODE:

Type here to search

10:08 AM
10/1/2019

File Edit View History Bookmarks Tools Help

More variable information...

www.pgrocer.net/Cis50/oradefac.html

```
  2  FROM inven
  3  ORDER BY &&orderhold;
old   3: ORDER BY &&orderhold
new   3: ORDER BY price

ITEM ITEMNAME             COST     PRICE
---- --------------- --------- ---------
1111 Good Night Moon         8     12.99
1234 Adven Reddy Fox         9     14.75
1212 Heidi                  10     14.99

3333 Basketball             14     17.99
2121 Teddy Bear             15     19.95
3456 Net/Hoop               25     27.95
2222 Building Blocks        23     27.98
2345 Doll House             45     55.98

8 rows selected.
```

You can also use the DEFINE and ACCEPT commands in SQL*Plus to predefine variables. The DE AR datatype field while the DEFINE variable with no value assigned to it simply displays the value and its datatype. The ACCEPT takes user input and ]). Notice that everything is optional accept the variable. The datatype can be NUMBER, CHAR or DATE. The HIDE suppresses the user entry as i . Here are some examples.

In this first example, I used &&orderhold to define the variable. The second SQL statement is this ex st and it is a CHAR field.

pgrocer@cisweb:~

```
--------------------------------------- -------- ----------------------------
ITEMNO                                           VARCHAR2(4)
ITEMNAME                                         VARCHAR2(15)
ONHAND                                           NUMBER(5)
ONORDER                                          NUMBER(5)
REORDPT                                          NUMBER(5)
COST                                             NUMBER(6,2)
PRICE                                            NUMBER(6,2)
DEPT                                             CHAR(2)
ITEMCLASS                                        CHAR(2)
LOCATION                                         VARCHAR2(4)

SQL> select &first from inven;
Enter value for first: onorder
old   1: select &first from inven
new   1: select onorder from inven

   ONORDER
----------
        30
        25
         0
        20
         0
         5
        25
         0

8 rows selected.

SQL>
```

## SQL CODE:

```
SQL> SELECT *
  2  FROM inven
  3  ORDER BY &&orderhold;
Enter value for orderhold: cost
old   3: ORDER BY &&orderhold
new   3: ORDER BY cost

ITEM ITEMNAME           ONHAND   ONORDER   REORDPT      COST     PRICE DE IT LOCA
---- --------------- --------- --------- --------- --------- --------- -- -- ----
1111 Good Night Moon        24        30        40         8     12.99 BK BY X100
1234 Adven Reddy Fox         5         0        10         9     14.75 BK CH X100
1212 Heidi                  12        25        25        10     14.99 BK CH X112
3333 Basketball             24        25        50        14     17.99 SP BK Y200
2121 Teddy Bear              5        20        40        15     19.95 TY CH X115
2222 Building Blocks         4         0        15        23     27.98 TY CH Z200
3456 Net/Hoop               12         0        25        25     27.95 SP BK Y200
2345 Doll House              2         5        10        45     55.98 TY CH Z212

8 rows selected.

SQL> DEFINE orderhold;
DEFINE ORDERHOLD       = "cost" (CHAR)
```

In this example, I am putting a value into the variable deptname. The value I am putting in is TY. I then ask that it show me the contents of the deptname just to double check. I then use deptname in a select.

## SQL CODE:

Type here to search

10:11 AM
10/1/2019

File Edit View History Bookmarks Tools Help

More variable information...

www.pgrocer.net/Cis50/oradefac.html

```
  2  FROM inven
  3  ORDER BY &&orderhold;
old   3: ORDER BY &&orderhold
new   3: ORDER BY price

ITEM ITEMNAME              COST     PRICE
---- --------------- --------- ---------
1111 Good Night Moon          8     12.99
1234 Adven Reddy Fox          9     14.75
1212 Heidi                   10     14.99

3333 Basketball              14     17.99
2121 Teddy Bear              15     19.95
3456 Net/Hoop                25     27.95
2222 Building Blocks         23     27.98
2345 Doll House              45     55.98

8 rows selected.
```

You can also use the DEFINE and ACCEPT commands in SQL*Plus to predefine variables. The DE ... AR datatype field while the DEFINE variable with no value assigned to it simply displays the value and its datatype. The ACCEPT takes user input and ... ]). Notice that everything is optional accept the variable. The datatype can be NUMBER, CHAR or DATE. The HIDE suppresses the user entry as i... . Here are some examples.

In this first example, I used &&orderhold to define the variable. The second SQL statement is this ex... st and it is a CHAR field.

**SQL CODE:**

```
SQL> SELECT *
  2  FROM inven
  3  ORDER BY &&orderhold;
Enter value for orderhold: cost
old   3: ORDER BY &&orderhold
new   3: ORDER BY cost

ITEM ITEMNAME           ONHAND   ONORDER   REORDPT      COST     PRICE DE IT LOC
---- --------------- --------- --------- --------- --------- --------- -- -- ---
1111 Good Night Moon        24        30        40         8     12.99 BK BY X10
1234 Adven Reddy Fox         5         0        10         9     14.75 BK CH X10
1212 Heidi                  12        25        25        10     14.99 BK CH X11
3333 Basketball             24        25        50        14     17.99 SP BK Y200
2121 Teddy Bear              5        20        40        15     19.95 TY CH X115
2222 Building Blocks         4         0        15        23     27.98 TY CH Z200
3456 Net/Hoop               12         0        25        25     27.95 SP BK Y200
2345 Doll House              2         5        10        45     55.98 TY CH Z212

8 rows selected.

SQL> DEFINE orderhold;
DEFINE ORDERHOLD       = "cost" (CHAR)
```

pgrocer@cisweb:~

```
LOCATION                                                   VARCHAR2(4)

SQL> select &first from inven;
Enter value for first: onorder
old   1: select &first from inven
new   1: select onorder from inven

   ONORDER
----------
        30
        25
         0
        20
         0
         5
        25
         0

8 rows selected.

SQL> define;
DEFINE _DATE           = "01-OCT-19" (CHAR)
DEFINE _CONNECT_IDENTIFIER = "orcl12c" (CHAR)
DEFINE _USER           = "PGROCER" (CHAR)
DEFINE _PRIVILEGE      = "" (CHAR)
DEFINE _SQLPLUS_RELEASE = "1201000100" (CHAR)
DEFINE _EDITOR         = "/usr/bin/nano" (CHAR)
DEFINE _O_VERSION      = "Oracle Database 12c Enterprise Edition Release 12.1.0.
1.0 - 64bit Production
With the Partitioning, OLAP, Advanced Analytics and Real Application Testing opt
ions" (CHAR)
DEFINE _O_RELEASE      = "1201000100" (CHAR)
DEFINE COL             = "state" (CHAR)
DEFINE ORDERHOLD       = "cost" (CHAR)
DEFINE ANOTHER         = "price" (CHAR)
SQL>
```

In this example, I am putting a value into the variable deptname. The value I am putting in is TY. I then ask that it show me the contents of the deptname just to double check. I then use deptname in a select.

**SQL CODE:**

Type here to search

10:12 AM
10/1/2019

Title: Oct 1-10:12 AM (10 of 25)

File Edit View History Bookmarks Tools Help

More variable information...

www.pgrocer.net/Cis50/oradefac.html

```
  2  FROM inven
  3  ORDER BY &&orderhold;
old   3: ORDER BY &&orderhold
new   3: ORDER BY price

ITEM ITEMNAME             COST     PRICE
---- --------------- --------- ---------
1111 Good Night Moon         8     12.99
1234 Adven Reddy Fox         9     14.75
1212 Heidi                  10     14.99

3333 Basketball             14     17.99
2121 Teddy Bear             15     19.95
3456 Net/Hoop               25     27.95
2222 Building Blocks        23     27.98
2345 Doll House             45     55.98

8 rows selected.
```

You can also use the DEFINE and ACCEPT commands in SQL*Plus to predefine variables. The DEFINE variable with a value assigned to it (Syntax: DEFINE variable = value) puts user input into a CHAR datatype field while the DEFINE variable with no value assigned to it simply displays the value and its datatype. The ACCEPT takes user input and stores it in the variable (Syntax: ACCEPT variable [datatype][FORMAT format][PROMPT text][Hide]). Notice that everything is optional accept the variable. The datatype can be NUMBER, CHAR or DATE. The HIDE suppresses the user entry as in the entry of a password - HIDE is only available in later versions, not in the test version you are using. Here are some examples.

In this first example, I used &&orderhold to define the variable. The second SQL statement is this example simply does a DEFINE on orderhold. It comes back with the fact that orderhold is defined as cost and it is a CHAR field.

### SQL CODE:

```
SQL> SELECT *
  2  FROM inven
  3  ORDER BY &&orderhold;
Enter value for orderhold: cost
old   3: ORDER BY &&orderhold
new   3: ORDER BY cost

ITEM ITEMNAME           ONHAND   ONORDER   REORDPT      COST     PRICE DE IT LOCA
---- --------------- --------- --------- --------- --------- --------- -- -- ----
1111 Good Night Moon        24        30        40         8     12.99 BK BY X100
1234 Adven Reddy Fox         5         0        10         9     14.75 BK CH X100
1212 Heidi                  12        25        25        10     14.99 BK CH X112
3333 Basketball             24        25        50        14     17.99 SP BK Y200
2121 Teddy Bear              5        20        40        15     19.95 TY CH X115
2222 Building Blocks         4         0        15        23     27.98 TY CH Z200
3456 Net/Hoop               12         0        25        25     27.95 SP BK Y200
2345 Doll House              2         5        10        45     55.98 TY CH Z212

8 rows selected.

SQL> DEFINE orderhold;
DEFINE ORDERHOLD       = "cost" (CHAR)
```

In this example, I am putting a value into the variable deptname. The value I am putting in is TY. I then ask that it show me the contents of the deptname just to double check. I then use deptname in a select.

### SQL CODE:

Type here to search

10:13 AM
10/1/2019

Title: Oct 1-10:13 AM (11 of 25)

File Edit View History Bookmarks Tools Help

More variable information...

www.pgrocer.net/Cis50/oradefac.html

```
2222 Building Blocks          4         0        15        23     27.98 TY CH Z200
3456 Net/Hoop                12         0        25        25     27.95 SP BK Y200
2345 Doll House               2         5        10        45     55.98 TY CH Z212

8 rows selected.

SQL> DEFINE orderhold;
DEFINE ORDERHOLD       = "cost" (CHAR)
```

In this example, I am putting a value into the variable deptname. The value I am putting in is TY. I then ask that it show me the contents of the deptname just to double check. I then use deptname in a select.

## SQL CODE:

```
SQL> DEFINE deptname = TY;
SQL> DEFINE deptname;
DEFINE DEPTNAME        = "TY" (CHAR)

SQL> SELECT *
  2  FROM inven
  3  WHERE dept = '&deptname';
old   3: WHERE dept = '&deptname'
new   3: WHERE dept = 'TY'

ITEM ITEMNAME           ONHAND   ONORDER   REORDPT      COST     PRICE DE IT LOCA
---- --------------- --------- --------- --------- --------- --------- -- -- ----
2121 Teddy Bear               5        20        40        15     19.95 TY CH X115
2222 Building Blocks          4         0        15        23     27.98 TY CH Z200
2345 Doll House               2         5        10        45     55.98 TY CH Z212
```

In the example above I was storing the actual value that I wanted to compare against in deptname so when I compared I put the &deptname in single quotes. In the example below, I am storing the name of a field in sortord. When I used it in the ORDER BY clause, I use it as &sortord without the single quotes surrounding it.

## SQL CODE:

```
SQL> DEFINE sortord = dept;
SQL> DEFINE sortord;
DEFINE SORTORD         = "dept" (CHAR)
SQL> SELECT *
  2  FROM inven
  3  ORDER BY &sortord;
old   3: ORDER BY &sortord
new   3: ORDER BY dept

ITEM ITEMNAME           ONHAND   ONORDER   REORDPT      COST     PRICE DE IT LOCA
---- --------------- --------- --------- --------- --------- --------- -- -- ----
1111 Good Night Moon         24        30        40         8     12.99 BK BY X100
1212 Heidi                   12        25        25        10     14.99 BK CH X112
1234 Adven Reddy Fox          5         0        10         9     14.75 BK CH X100
3333 Basketball              24        25        50        14     17.99 SP BK Y200
3456 Net/Hoop                12         0        25        25     27.95 SP BK Y200
2121 Teddy Bear               5        20        40        15     19.95 TY CH X115
2222 Building Blocks          4         0        15        23     27.98 TY CH Z200
2345 Doll House               2         5        10        45     55.98 TY CH Z212
```

Type here to search

10:14 AM
10/1/2019

```
SQL> DEFINE sortord = dept;
SQL> DEFINE sortord;
DEFINE SORTORD         = "dept" (CHAR)
SQL> SELECT *
  2  FROM inven
  3  ORDER BY &sortord;
old   3: ORDER BY &sortord
new   3: ORDER BY dept

ITEM ITEMNAME           ONHAND   ONORDER   REORDPT      COST     PRICE DE IT LOCA
---- --------------- --------- --------- --------- --------- --------- -- -- ----
1111 Good Night Moon        24        30        40         8     12.99 BK BY X100
1212 Heidi                  12        25        25        10     14.99 BK CH X112
1234 Adven Reddy Fox         5         0        10         9     14.75 BK CH X100
3333 Basketball             24        25        50        14     17.99 SP BK Y200
3456 Net/Hoop               12         0        25        25     27.95 SP BK Y200
2121 Teddy Bear              5        20        40        15     19.95 TY CH X115
2222 Building Blocks         4         0        15        23     27.98 TY CH Z200
2345 Doll House              2         5        10        45     55.98 TY CH Z212

8 rows selected.
```

Even though the define sets up a char field, Oracle can put a number in that field and compare it correctly with the number field cost.

## SQL CODE:

```
SQL> DEFINE costhold = 10;
SQL> DEFINE costhold;
DEFINE COSTHOLD        = "10" (CHAR)
SQL> SELECT *
  2  FROM inven
  3  WHERE cost > '&costhold';
old   3: WHERE cost > '&costhold'
new   3: WHERE cost > '10'

ITEM ITEMNAME           ONHAND   ONORDER   REORDPT      COST     PRICE DE IT LOCA
---- --------------- --------- --------- --------- --------- --------- -- -- ----
2121 Teddy Bear              5        20        40        15     19.95 TY CH X115
2222 Building Blocks         4         0        15        23     27.98 TY CH Z200
2345 Doll House              2         5        10        45     55.98 TY CH Z212
3333 Basketball             24        25        50        14     17.99 SP BK Y200
3456 Net/Hoop               12         0        25        25     27.95 SP BK Y200
```

Variables can be deleted or erased by using the UNDEFINE command on the variable or by leaving SQL*Plus. In other words, they will not be available in subsequent sessions. After undefining costhold, I checked to make sure it worked by using DEFINE costhold.

## SQL CODE:

```
SQL> UNDEFINE costhold;
SQL> DEFINE costhold;
symbol costhold is UNDEFINED
```

In the examples below, I first asked to see what was in deptname. In contained TY as a result of previous work I had done. I then asked it to ACCEPT deptname and it moved to the next line and waited. I entered BK. Then the SQL> prompt came up and I

File Edit View History Bookmarks Tools Help

More variable information...

www.pgrocer.net/Cis50/oradefac.html

**SQL CODE:**

```
SQL> UNDEFINE costhold;
SQL> DEFINE costhold;
symbol costhold is UNDEFINED
```

In the examples below, I first asked to see what was in deptname. In contained TY as a result of previous work I had done. I then asked it to ACCEPT deptname and it moved to the next line and waited. I entered BK. Then the SQL> prompt came up and I said DEFINE deptname to see what it contained. It contained the BK. This example convinced me of the value of the prompt, so on my next attempt I used the prompt. When the prompt appeared I entered SP which the DEFINE shows is now in deptname.

**SQL CODE:**

```
SQL> DEFINE deptname;
DEFINE DEPTNAME        = "TY" (CHAR)
SQL> ACCEPT deptname;
BK
SQL> DEFINE deptname;
DEFINE DEPTNAME        = "BK" (CHAR)
SQL> ACCEPT deptname PROMPT 'Please enter the name of the department you are using: ';
Please enter the name of the department you are using: SP
SQL> DEFINE deptname;
DEFINE DEPTNAME        = "SP" (CHAR)
```

In the example below I am using the ACCEPT to set up or define a variable called newreordpt as a NUMBER field. The PROMPT is also used. I then use this new field to compare against in the select. Notice that no quotes are needed since this is a numeric field.

**SQL CODE:**

```
SQL> ACCEPT newreordpt NUMBER PROMPT 'Enter reorderpt to compare against: ';
Enter reorderpt to compare against: 25
SQL> DEFINE newreordpt;
DEFINE NEWREORDPT      =        25 (NUMBER)
SQL> SELECT *
  2  FROM inven
  3  WHERE reordpt >= &newreordpt;
old   3: WHERE reordpt >= &newreordpt
new   3: WHERE reordpt >=        25

ITEM ITEMNAME           ONHAND   ONORDER   REORDPT      COST     PRICE DE IT LOCA
---- --------------- --------- --------- --------- --------- --------- -- -- ----
1111 Good Night Moon        24        30        40         8     12.99 BK BY X100
1212 Heidi                  12        25        25        10     14.99 BK CH X112
2121 Teddy Bear              5        20        40        15     19.95 TY CH X115
3333 Basketball             24        25        50        14     17.99 SP BK Y200
3456 Net/Hoop               12         0        25        25     27.95 SP BK Y200
```

This example takes in new price as a number formatted with two decimal places. I then used this in the WHERE statement to find matching prices.

**SQL CODE:**

```
SQL> ACCEPT newprice NUMBER FORMAT 9999.99 PROMPT 'Enter price to compare against: ';
Enter price to compare against: 12.99
```

Type here to search

10:16 AM
10/1/2019

File Edit View History Bookmarks Tools Help

More variable information...

www.pgrocer.net/Cis50/oradefac.html

```
SQL> DEFINE deptname;
DEFINE DEPTNAME        = "TY" (CHAR)
SQL> ACCEPT deptname;
BK
SQL> DEFINE deptname;
DEFINE DEPTNAME        = "BK" (CHAR)
SQL> ACCEPT deptname PROMPT 'Please enter the name of the department you are using: ';
Please enter the name of the department you are using: SP
SQL> DEFINE deptname;
DEFINE DEPTNAME        = "SP" (CHAR)
```

In the example below I am using the ACCEPT to set up or define a variable called newreordpt as a NUMBER field. The PROMPT is also used. I then use this new field to compare against in the select. Notice that no quotes are needed since this is a numeric field.

### SQL CODE:

```
SQL> ACCEPT newreordpt NUMBER PROMPT 'Enter reorderpt to compare against: ';
Enter reorderpt to compare against: 25
SQL> DEFINE newreordpt;
DEFINE NEWREORDPT      =        25 (NUMBER)
SQL> SELECT *
  2  FROM inven
  3  WHERE reordpt >= &newreordpt;
old   3: WHERE reordpt >= &newreordpt
new   3: WHERE reordpt >=        25

ITEM ITEMNAME            ONHAND   ONORDER   REORDPT      COST     PRICE DE IT LOCA
---- --------------- --------- --------- --------- --------- --------- -- -- ----
1111 Good Night Moon        24        30        40         8     12.99 BK BY X100
1212 Heidi                  12        25        25        10     14.99 BK CH X112
2121 Teddy Bear              5        20        40        15     19.95 TY CH X115
3333 Basketball             24        25        50        14     17.99 SP BK Y200
3456 Net/Hoop               12         0        25        25     27.95 SP BK Y200
```

This example takes in new price as a number formatted with two decimal places. I then used this in the WHERE statement to find matching prices.

### SQL CODE:

```
SQL> ACCEPT newprice NUMBER FORMAT 9999.99 PROMPT 'Enter price to compare against: ';
Enter price to compare against: 12.99
SQL> DEFINE newprice;
DEFINE NEWPRICE        =     12.99 (NUMBER)

SQL> SELECT *
  2  FROM inven
  3  WHERE price = &newprice;
old   3: WHERE price = &newprice
new   3: WHERE price =     12.99

ITEM ITEMNAME            ONHAND   ONORDER   REORDPT      COST     PRICE DE IT LOCA
---- --------------- --------- --------- --------- --------- --------- -- -- ----
1111 Good Night Moon        24        30        40         8     12.99 BK BY X100
```

Type here to search

10:17 AM
10/1/2019

Title: Oct 1-10:17 AM (15 of 25)

File Edit View History Bookmarks Tools Help

More maintenance...

www.pgrocer.net/Cis50/oramain3.html

```
SQL>
Enter value for userid: 456
Enter value for username: Blocks
Enter value for userprice: 10
Enter value for userpru: 12-JUN-99
Enter value for userdept: TY
Enter value for usercost: 8
old   2: VALUES('&userid', '&username', &userprice, '&userpru', '&userdept', &usercost)
new   2: VALUES('456', 'Blocks', 10, '12-JUN-99', 'TY', 8)

1 row created.

SQL> SELECT * FROM maintain;

IDN ITEMNAME         PRICE PURCHASED DE      COST
--- ------------ --------- --------- -- ---------
123 Teddy Bear          20           TY        18
234 Dump Truck       15.95           TY     14.36
345 Baby Doll           12 23-JUN-99
456 Blocks              10 12-JUN-99 TY         8
```

## Taking rows from another table

The statement below creates a new table called movemain. It has four fields that match the description of the table maintain.

### SQL CODE:

```
SQL> CREATE TABLE movemain
  2  ( itemnum VARCHAR2(3), itemname VARCHAR2(12), itemcost NUMBER(6,2), itemprice NUMBER(6,2));

Table created.

SQL> DESC maintain;
 Name                            Null?    Type
 ------------------------------- -------- ----
 IDNO                                     VARCHAR2(3)
 ITEMNAME                                 VARCHAR2(12)
 PRICE                                    NUMBER(6,2)
 PURCHASED                                DATE
 DEPT                                     CHAR(2)
 COST                                     NUMBER(6,2)

SQL> DESC movemain;

 Name                            Null?    Type
 ------------------------------- -------- ----
 ITEMNUM                                  VARCHAR2(3)
 ITEMNAME                                 VARCHAR2(12)
 ITEMCOST                                 NUMBER(6,2)
 ITEMPRICE                                NUMBER(6,2)
```

In the example below, I want to take some of the information from the table maintain and move it to the table movemain. For this move, I am selecting the four columns/fields that are the structure for movemain from all rows/records where the price is > 15. The four columns are moved for the two records that meet the criteria. In the second example, I only want to move over the item number, the item name and the cost. I do not want to move the price. To accomplish this, I need to specify the receiving fields in the INSERT.

Type here to search

10:19 AM
10/1/2019

File Edit View History Bookmarks Tools Help

More maintenance...

www.pgrocer.net/Cis50/oramain3.html

```
SQL> DESC maintain;
 Name                            Null?    Type
 ------------------------------- -------- ----
 IDNO                                     VARCHAR2(3)
 ITEMNAME                                 VARCHAR2(12)
 PRICE                                    NUMBER(6,2)
 PURCHASED                                DATE
 DEPT                                     CHAR(2)
 COST                                     NUMBER(6,2)

SQL> DESC movemain;

 Name                            Null?    Type
 ------------------------------- -------- ----
 ITEMNUM                                  VARCHAR2(3)
 ITEMNAME                                 VARCHAR2(12)
 ITEMCOST                                 NUMBER(6,2)
 ITEMPRICE                                NUMBER(6,2)
```

In the example below, I want to take some of the information from the table maintain and move it to the table movemain. For this move, I am selecting the four columns/fields that are the structure for movemain from all rows/records where the price is > 15. The four columns are moved for the two records that meet the criteria. In the second example, I only want to move over the item number, the item name and the cost. I do not want to move the price. To accomplish this, I need to specify the receiving fields in the INSERT.

## SQL CODE:

```
SQL> INSERT INTO movemain
  2      SELECT idno, itemname, price, cost
  3      FROM maintain
  4      WHERE price > 15;

2 rows created.

SQL> SELECT * FROM movemain;

ITE ITEMNAME      ITEMCOST ITEMPRICE
--- ------------ --------- ---------
123 Teddy Bear          20        18
234 Dump Truck       15.95     14.36
```

In the second example, I only want to move over the item number, the item name and the cost. I do not want to move the price. To accomplish this, I need to specify the receiving fields in the INSERT. Note that the order of fields in the INSERT corresponds to the order in the SELECT. The where clause in this example is selecting a record with a specific id number.

## SQL CODE:

```
SQL> INSERT INTO movemain (itemnum, itemname, itemprice)
  2      SELECT idno, itemname, price
  3      FROM maintain
  4      WHERE idno = '456';

1 row created.

SQL> SELECT * FROM movemain;
```

Type here to search

10:21 AM 10/1/2019

File Edit View History Bookmarks Tools Help

More maintenance...

www.pgrocer.net/Cis50/oramain3.html

```
234 Dump Truck       15.95     14.36
456 Blocks                        10
```

Now I am altering the table movemain to have the column ITEMDEPT.

## SQL CODE:

```
SQL> ALTER TABLE movemain
  2  ADD (itemdept CHAR(2));

Table altered.

SQL> DESC movemain;
 Name                            Null?    Type
 ------------------------------- -------- ----
 ITEMNUM                                  VARCHAR2(3)
 ITEMNAME                                 VARCHAR2(12)
 ITEMCOST                                 NUMBER(6,2)
 ITEMPRICE                                NUMBER(6,2)
 ITEMDEPT                                 CHAR(2)
```

Now I am going to put the department TY into the the record with id 234 on movemain. The way I do this is to UPDATE movemain and set the itemdept = to the dept from the maintain file in the row where idno = 123. I do this for the row/record in movemain where the itemnum = 234.

## SQL CODE:

```
SQL> UPDATE movemain
  2  SET itemdept = (SELECT dept
  3                  FROM maintain
  4                  WHERE idno = '123')
  5  WHERE itemnum = '234';

1 row updated.

SQL> SELECT * FROM movemain;

ITE ITEMNAME      ITEMCOST ITEMPRICE IT
--- ------------ --------- --------- --
123 Teddy Bear          20        18
234 Dump Truck       15.95     14.36 TY
456 Blocks                        10
```

In this example, I am setting the itemdept in the row/record where itemnum is 456 to be equal to the row/record in the same table where itemnum = 234 (that is the one I updated in the code above).

## SQL CODE:

```
SQL> UPDATE movemain
  2  SET itemdept = (SELECT itemdept
  3                  FROM movemain
  4                  WHERE itemnum = '234')
  5  WHERE itemnum = '456';

1 row updated.
```

Type here to search

10:23 AM 10/1/2019

File Edit View History Bookmarks Tools Help

More maintenance...

www.pgrocer.net/Cis50/oramain3.html

In this example, I am setting the itemdept in the row/record where itemnum is 456 to be equal to the row/record in the same table where itemnum = 234 (that is the one I updated in the code above).

### SQL CODE:

```
SQL> UPDATE movemain
  2  SET itemdept = (SELECT itemdept
  3                  FROM movemain
  4                  WHERE itemnum = '234')
  5  WHERE itemnum = '456';

1 row updated.

SQL> SELECT * FROM movemain;

ITE ITEMNAME      ITEMCOST ITEMPRICE IT
--- ------------ --------- --------- --
123 Teddy Bear          20        18
234 Dump Truck       15.95     14.36 TY
456 Blocks                        10 TY
```

## Deleting a row

Remember the DELETE FROM table statement allows you to delete rows from a table. The WHERE condition attached to the DELETE allows you to remove a row or rows that meet specific criteria. Omitting the where clause deletes all rows from the table but leaves the table structure. I added a couple of more rows to the maintain table so it now looks like this.

### SQL CODE:

```
SQL> SELECT * FROM maintain;

IDN ITEMNAME         PRICE PURCHASED DE      COST
--- ------------ --------- --------- -- ---------
123 Teddy Bear          20           TY        18
234 Dump Truck       15.95           TY     14.36
345 Baby Doll           12 23-JUN-99
456 Blocks              10 12-JUN-99 TY         8
567 Tea Set              9 09-JUN-99 TY       7.5
678 Warrior          15.99 09-JUN-99 TY        14

6 rows selected.
```

A delete statement to delete all records where cost is less than 8 and the results are shown below:

### SQL CODE:

```
SQL> DELETE FROM maintain
  2  WHERE cost < 8;

1 row deleted.
```

Type here to search

10:23 AM
10/1/2019

Title: Oct 1-10:24 AM (19 of 25)

but does not effect the update of the date so I do a rollback to the savepoint date_update. If you look at the table after the rollback the deleted record is restored and the updated record 123 still has a date. Finally I do another rollback with no savepoint and this rollback goes back and undoes the update. Record 123 now has no date.

### SQL CODE:

```
SQL> SELECT * FROM maintain;

IDN ITEMNAME          PRICE PURCHASED DE      COST
--- ------------ --------- --------- -- ---------
123 Teddy Bear           20           TY        18
234 Dump Truck        15.95           TY     14.36
345 Baby Doll            12 23-JUN-99
456 Blocks               10 12-JUN-99 TY         8
678 Warrior           15.99 09-JUN-99 TY        14

SQL> UPDATE maintain
  2  SET purchased = '12-MAY-99'
  3  WHERE idno = '123';

1 row updated.

SQL> SELECT * FROM maintain;

IDN ITEMNAME          PRICE PURCHASED DE      COST
--- ------------ --------- --------- -- ---------
123 Teddy Bear           20 12-MAY-99 TY        18
234 Dump Truck        15.95           TY     14.36
345 Baby Doll            12 23-JUN-99
456 Blocks               10 12-JUN-99 TY         8
678 Warrior           15.99 09-JUN-99 TY        14

SQL> SAVEPOINT date_update;

Savepoint created.

SQL> DELETE FROM maintain
  2  WHERE idno = '345';

1 row deleted.

SQL> SELECT * FROM maintain;

IDN ITEMNAME          PRICE PURCHASED DE      COST
--- ------------ --------- --------- -- ---------
123 Teddy Bear           20 12-MAY-99 TY        18
234 Dump Truck        15.95           TY     14.36
456 Blocks               10 12-JUN-99 TY         8
678 Warrior           15.99 09-JUN-99 TY        14

SQL> ROLLBACK to date_update;

Rollback complete.

SQL> SELECT * FROM maintain;

IDN ITEMNAME          PRICE PURCHASED DE      COST
--- ------------ --------- --------- -- ---------
```

File Edit View History Bookmarks Tools Help

Maintenance cont.

www.pgrocer.net/Cis50/oramain4.html

# Continuation of table maintenance revisited (again)...

Oracle has a table dictionary called user_tables which gives detailed information about the tables, an object table called user_objects which gives information about objects and the user_catalog. If you want to see specific characteristics you can do a desc of user_objects or sys.user.objects for example, look at the types and then do a select using that type as a column name. Distinct can be used to avoid repetition. The code below shows a partial list.

### SQL CODE:

```
SQL> Desc user_objects;
 Name                            Null?    Type
 ------------------------------- -------- ----
 OBJECT_NAME                              VARCHAR2(128)
 OBJECT_ID                                NUMBER
 OBJECT_TYPE                              VARCHAR2(13)
 CREATED                                  DATE
 LAST_DDL_TIME                            DATE
 TIMESTAMP                                VARCHAR2(75)
 STATUS                                   VARCHAR2(7)

SQL> SELECT object_name, object_type, created
  2  FROM user_objects;

OBJECT_NAME
--------------------------------
OBJECT_TYPE   CREATED
------------- ---------
BONUS
TABLE         16-SEP-96

DEPARTMENT
TABLE         17-JUN-99

DEPT
TABLE         16-SEP-96

DEPTARTMENT
TABLE         23-JUN-99

DEPTNO_PK
INDEX         23-JUN-99

DONATION
TABLE         23-JUN-99
```

If you want to look at the tables, views etc. that you have created, you can do a SELECT * FROM USER_CATALOG or just SELECT * FROM CAT. This will print out the list as shown below.

### SQL CODE:

```
SQL> select * from cat;

TABLE_NAME                     TABLE_TYPE
------------------------------ -----------
```

Type here to search

10:29 AM
10/1/2019

# CREATE

Here are a few more things about the create. When you create a table you can set a DEFAULT value to be entered if the INSERT does not put data into the column. When doing this, you cannot use another column name, it must be a literal, an expression or a function and the data type must match.

## SQL CODE:

```
SQL> CREATE TABLE test1
  2  (idnum NUMBER (3), name VARCHAR(2), dept CHAR(2) DEFAULT 'CI', dateapply DATE DEFAULT sysdate);


Table created.

SQL> DESC test1;
 Name                            Null?    Type
 ------------------------------- -------- ----
 IDNUM                                    NUMBER(3)
 NAME                                     VARCHAR2(2)
 DEPT                                     CHAR(2)
 DATEAPPLY                                DATE

SQL> INSERT INTO test1 (idnum, name)
  2  VALUES (111, 'AB');

1 row created.

SQL> SELECT * FROM test1;

    IDNUM NA DE DATEAPPLY
--------- -- -- ---------
      111 AB CI 24-JUN-99

SQL> INSERT INTO test1 (idnum, name)
  2  VALUES (112, 'BC');

1 row created.

SQL> INSERT INTO test1 (idnum, name, dateapply)
  2  VALUES (115,'CD', '12-JUN-99');

SQL> INSERT INTO test1 (idnum, dept)
  2  VALUES (116,'BU');

SQL> INSERT INTO test1 (idnum, dept, dateapply)
  2  VALUES (115, 'GS', last_day(sysdate));

SQL> SELECT * FROM test1;

    IDNUM NA DE DATEAPPLY
--------- -- -- ---------
      111 AB CI 24-JUN-99
      112 BC CI 24-JUN-99
      115 CD CI 12-JUN-99
      116    BU 24-JUN-99
      115    GS 30-JUN-99
```

File Edit View History Bookmarks Tools Help

Maintenance cont.

www.pgrocer.net/Cis50/oramain4.html

```
 Name                            Null?    Type
 ------------------------------- -------- ----
 IDNO                                     VARCHAR2(3)
 ITEMNAME                                 VARCHAR2(12)
 UPDTPRICE                                NUMBER
 DEPT                                     CHAR(2)

SQL> SELECT * FROM newmain;

IDN ITEMNAME     UPDTPRICE DE
--- ------------ --------- --
123 Teddy Bear          21 TY
234 Dump Truck       16.95 TY
678 Warrior          16.99 TY
```

# DROP TABLE

The DROP TABLE command is used to drop a table which removes the definition of the table and all data in the table. It is not reversible since it is a Data Definition Language (DDL) statement and is immediately committed when executed and there is now warning about are you sure etc., so be careful. In the example below I created a table called test1x from test1, showed the successful creation, dropped the table and showed that it was no longer available.

## SQL CODE:

```
SQL> CREATE TABLE test1x
  2  AS
  3    SELECT * FROM test1;

Table created.

SQL> SELECT * from test1x;

    IDNUM NA DE DATEAPPLY
--------- -- -- ---------
      111 AB CI 24-JUN-99
      112 BC CI 24-JUN-99
      115 CD CI 12-JUN-99
      116    BU 24-JUN-99
      115    GS 30-JUN-99

SQL> DROP TABLE test1x;

Table dropped.

SQL> SELECT * FROM test1x;
SELECT * FROM test1x
              *
ERROR at line 1:
ORA-00942: table or view does not exist
```

# RENAME

A table, view, sequence or synonym (coming) can be renamed using the rename. This is a DDL so it commits upon execution.

Type here to search

10:33 AM 10/1/2019

File Edit View History Bookmarks Tools Help

Maintenance cont.

www.pgrocer.net/Cis50/oramain4.html

```
      112 BC CI 24-JUN-99
      115 CD CI 12-JUN-99
      116    BU 24-JUN-99
      115    GS 30-JUN-99

SQL> RENAME test12 TO test2;

Table renamed.

SQL> SELECT * FROM test2;

    IDNUM NA DE DATEAPPLY
--------- -- -- ---------
      111 AB CI 24-JUN-99
      112 BC CI 24-JUN-99
      115 CD CI 12-JUN-99
      116    BU 24-JUN-99
      115    GS 30-JUN-99

SQL> DESC test2;
 Name                            Null?    Type
 ------------------------------- -------- ----
 IDNUM                                    NUMBER(3)
 NAME                                     VARCHAR2(2)
 DEPT                                     CHAR(2)
 DATEAPPLY                                DATE
```

## TRUNCATE

The truncate statement removes all of the rows in the table and releases the storage space. It is a DDL so it is committed upon execution. The DELETE can also be used to remove all rows however it does no release the storage space and it can be rolled back because it is not a DDL statement.

### SQL CODE:

```
SQL> TRUNCATE TABLE test2;

Table truncated.

SQL> SELECT * FROM test2;

no rows selected

SQL> DESC test2;
 Name                            Null?    Type
 ------------------------------- -------- ----
 IDNUM                                    NUMBER(3)
 NAME                                     VARCHAR2(2)
 DEPT                                     CHAR(2)
 DATEAPPLY                                DATE
```

## ALTER

Don't forget that you can alter a table by adding or modifying columns, this was covered in another handout. Here I am going to alter test2 to give the name more characters.

Type here to search

10:33 AM 10/1/2019

File Edit View History Bookmarks Tools Help

Maintenance cont.

www.pgrocer.net/Cis50/oramain4.html

```
Table truncated.

SQL> SELECT * FROM test2;

no rows selected

SQL> DESC test2;
 Name                            Null?    Type
 ------------------------------- -------- ----
 IDNUM                                    NUMBER(3)
 NAME                                     VARCHAR2(2)
 DEPT                                     CHAR(2)
 DATEAPPLY                                DATE
```

## ALTER

Don't forget that you can alter a table by adding or modifying columns, this was covered in another handout. Here I am going to alter test2 to give the name more characters.

### SQL CODE:

```
SQL> DESC test1;
 Name                            Null?    Type
 ------------------------------- -------- ----
 IDNUM                                    NUMBER(3)
 NAME                                     VARCHAR2(2)
 DEPT                                     CHAR(2)
 DATEAPPLY                                DATE

SQL> ALTER TABLE test1
  2  MODIFY (name VARCHAR2(15));

Table altered.

SQL> DESC test1;
 Name                            Null?    Type
 ------------------------------- -------- ----
 IDNUM                                    NUMBER(3)
 NAME                                     VARCHAR2(15)
 DEPT                                     CHAR(2)
 DATEAPPLY                                DATE
```

## COMMENTS

The comment command can be used to add comments to a table or to a column within the table. To view the comments you can query the data dictionary. The data dictionaries that can hold the comments are (ALL_COL_COMMENTS, USER_COL_COMMENTS, ALL_TAB_COMMENTS, USER_TAB_COMMENTS). To get rid of a comment you can set it to an empty string which is done with the IS ' ' clause.

### SQL CODE:

```
SQL> COMMENT ON TABLE TEST1
  2  IS 'This is a test table created and modified frequently';

Comment created.
```

Type here to search

10:34 AM
10/1/2019