Practical No: 03

-**1.** Write a query to create range portioned table:

 Creates a table named- Sales consisting of four partitions, one for each quarter of sales. The columns sale_year, sale_month, and sale_day are the partitioning columns, while their values constitute the partitioning key of a specific row.

 Each partition is given a name (sales_q1, sales_q2, ...), and each partition is contained in a separate tablespace (tsa, tsb, ...)
 The columns for table must be prod_id, cust_id, promo_id, quantify sold, amount_sold – all in number format and time_id.

```
create TABLESPACE tsa DATAFILE 'E:\DW/tsa.dbf' SIZE 10M;
create TABLESPACE tsb DATAFILE 'E:\DW/tsb.dbf' SIZE 10M;
create TABLESPACE tsc DATAFILE 'E:\DW/tsc.dbf' SIZE 10M;
create TABLESPACE tsd DATAFILE 'E:\DW/tsd.dbf' SIZE 10M;

SQL> CREATE TABLE sales
  2  ( prod_id number(6),
  3    cust_id number,
  4    time_id date,
  5    channel_id char(1),
  6    promo_id number(3),
  7    amount_sold number(10,2))
  8    partition by range(time_id)
  9    (partition sales_q1 values less than(to_date('01-APR-2006','DD-MON-YYYY'))
 10     TABLESPACE tsa,
 11    partition sales_q2 values less than(to_date('01-JUL-2006','DD-MON-YYYY'))
 12     TABLESPACE tsb,
 13    partition sales_q3 values less than(to_date('01-OCT-2006','DD-MON-YYYY'))
 14    TABLESPACE tsc,
 15    partition sales_q4 values less than(to_date('01-JAN-2007','DD-MON-YYYY'))
 16   TABLESPACE tsd  );
Table created.

SQL> insert into sales values(101,111,'01-JAN-2006','B',999,10);
1 row created.
SQL>  insert into sales values(102,222,'01-APR-2006','B',999,10);
1 row created.
SQL>  insert into sales values(103,333,'01-JUL-2006','B',999,10);
1 row created.
SQL>  insert into sales values(104,444,'01-DEC-2006','B',999,10);
1 row created.

SQL> select partition_name,tablespace_name,high_value,num_rows from user_tab_partitions where
table_name='SALES';



PARTITION_NAME                 TABLESPACE_NAME
------------------------------ ------------------------------
HIGH_VALUE
--------------------------------------------------------------------------------
  NUM_ROWS
```

```
----------
SALES_Q1                       TSA
TO_DATE(' 2006-04-01 00:00:00', 'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIA
         1
SALES_Q2                       TSB
TO_DATE(' 2006-07-01 00:00:00', 'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIA
         1

PARTITION_NAME                 TABLESPACE_NAME
------------------------------ ------------------------------
HIGH_VALUE
--------------------------------------------------------------------------------
  NUM_ROWS----------
SALES_Q3                       TSC
TO_DATE(' 2006-10-01 00:00:00', 'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIA
         1
SALES_Q4                       TSD
TO_DATE(' 2007-01-01 00:00:00', 'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIA

PARTITION_NAME                 TABLESPACE_NAME
------------------------------ ------------------------------
HIGH_VALUE
--------------------------------------------------------------------------------
  NUM_ROWS
----------
         1
```

---

2. Create the same table as in Q1. With a different name with ENABLE ROW MOVEMENT

```
SQL> CREATE TABLE sales_row_movement
  2  ( prod_id number(6),
  3    cust_id number,
  4    time_id date,
  5    channel_id char(1),
  6    promo_id number(3),
  7    amount_sold number(10,2))
  8    partition by range(time_id)
  9    (partition sales_q1 values less than(to_date('01-APR-2006','DD-MON-YYYY'))
 10     TABLESPACE tsa,
 11    partition sales_q2 values less than(to_date('01-JUL-2006','DD-MON-YYYY'))
 12     TABLESPACE tsb,
 13    partition sales_q3 values less than(to_date('01-OCT-2006','DD-MON-YYYY'))
 14    TABLESPACE tsc,
 15    partition sales_q4 values less than(to_date('01-JAN-2007','DD-MON-YYYY'))
 16   TABLESPACE tsd  )
 17   ENABLE ROW MOVEMENT;

Table created.


SQL>  insert into sales values(101,111,'01-JAN-2006','B',999,10);
1 row created.
SQL> insert into sales values(102,222,'01-APR-2006','B',999,10);
1 row created.
SQL> insert into sales values(103,333,'01-JUL-2006','B',999,10);
1 row created.
```

```
SQL> insert into sales values(104,444,'01-DEC-2006','B',999,10);
1 row created.


SQL> update sales_row_movement set time_id='03-JUL-2006' WHERE time_id='01-JAN-2006';
SQL> select partition_name,tablespace_name,high_value,num_rows from user_tab_partitions where
table_name='SALES_ROW_MOVEMENT';

PARTITION_NAME                 TABLESPACE_NAME
------------------------------ ------------------------------
HIGH_VALUE
--------------------------------------------------------------------------------
  NUM_ROWS
----------
SALES_Q1                       TSA
TO_DATE(' 2006-04-01 00:00:00', 'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIA


SALES_Q2                       TSB
TO_DATE(' 2006-07-01 00:00:00', 'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIA


PARTITION_NAME                 TABLESPACE_NAME
------------------------------ ------------------------------
HIGH_VALUE
--------------------------------------------------------------------------------
  NUM_ROWS
----------

SALES_Q3                       TSC
TO_DATE(' 2006-10-01 00:00:00', 'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIA


SALES_Q4                       TSD
TO_DATE(' 2007-01-01 00:00:00', 'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIA

PARTITION_NAME                 TABLESPACE_NAME
------------------------------ ------------------------------
HIGH_VALUE
--------------------------------------------------------------------------------
  NUM_ROWS
----------
```

---

3. Create a table with list partition as follows:
 Table having columns deptno, deptname, quarterly_sales and state.
 Create partition on state:
 Northwest on OR and WA
 Southwest on AZ, UT and NM
 northeast on NY, VM and NJ
 southeast on FL and GA
 northcentral on SD and WI
 southcentral on OK and TX
 Add the following entries into the table and make conclusion to which partition the entry maps:
 (10, 'accounting', 100, 'WA')
 (20, 'R&D', 150, 'OR')
 (30, 'sales', 100, 'FL')
 (40, 'HR', 10, 'TX')

```
 (50, 'systems engineering', 10, 'CA')
-------------------------------------------------------------------------------------------

SQL> CREATE TABLE sales2
  2       (deptno number,
  3        deptname varchar2(20),
  4        quarterly_sales number(10, 2),
  5        state varchar2(2))
  6    PARTITION BY LIST (state)
  7       (PARTITION northwest VALUES ('OR', 'WA'),
  8        PARTITION southwest VALUES ('AZ', 'UT', 'NM'),
  9        PARTITION northeast VALUES  ('NY', 'VM', 'NJ'),
 10       PARTITION southeast VALUES ('FL', 'GA'),
 11      PARTITION nc VALUES ('SD','WI'),
 12      PARTITION sc VALUES ('OK','TX'));

Table created.

SQL> alter table sales2 add partition def values(default);
Table altered.


SQL> INSERT INTO sales2 values(10, 'accounts', 110, 'WA');
1 row created.
SQL> INSERT INTO sales2 values(20, 'Developer', 150, 'OR');
1 row created.
SQL> INSERT INTO sales2 values(30, 'sales', 110, 'FL');
1 row created.
SQL> INSERT INTO sales2 values(40, 'HR', 10, 'TX');
1 row created.
SQL> INSERT INTO sales2 values(50, 'Marketing', 10, 'CA');
1 row created.


DEPTNO DEPTNAME          QUARTERLY_SALES ST

---------- -------------------- --------------- --

   10 accounts                 110 WA
   20 Developer                150 OR
   30 sales                    110 FL
   40 HR                        10 TX
   50 Marketing                 10 CA
```

-**4.** Create a table with hash partition as follows:

 Create table Emp with attributes empno, job, sal, deptno and perform hash partitioning on empno. Number of Partitions should be 5. Demonstarte using system defined and user defined partition concepts.

```
SQL> Create table Employee_hash
  2  (emp_no number(2),
  3   job varchar2(5),
  4  sal number,
  5  deptno number )
  6  partition by hash (emp_no)
  7  partitions 5;

Table created.

SQL> select partition_name from user_tab_partitions where table_name='EMPLOYEE_HASH';
```

```
PARTITION_NAME
------------------------------
SYS_P181
SYS_P182
SYS_P183
SYS_P184
SYS_P185

SQL> Create table emp2
  2  (emp_no number(2),
  3   job varchar2(5),
  4  sal number,
  5  deptno number )
  6  partition by hash (emp_no)
  7  (partition h1,
  8   partition h2,
  9    partition h3,
 10    partition h4,
 11    partition h5);

Table created.

SQL> select partition_name from user_tab_partitions where table_name='EMP2';

PARTITION_NAME
------------------------------
H1
H2
H3
H4
H5
```

---

5. Create a multi-column range partitioned table as directed:
 Create a table with the actual DATE information in three separate columns: year, month, and day. Also amount_ sold.
 Create following partitions:
o Before 2001: Less than jan 2001
o Less than april 2001
o Less than july 2001
o Les than oct 2001
o Less than jan 2002
o Future with max incoming value
 Insert values into table and show to which partition does the value belong.
o (2001,3,17, 2000);
o (2001,11,1, 5000);
o (2002,1,1, 4000); Make conclusion for each result.

```
SQL> create table tab5 (
  2   year number,
  3   month number,
  4   day number,
  5   amount_sold number)
  6   partition by range(year,month)
  7   ( partition p1 values less than (2001,1),
  8     partition p2 values less than (2001,4),
  9     partition p3 values less than (2001,7),
 10     partition p4 values less than (2001,10),
 11     partition p5 values less than (2002,1),
 12     partition p6 values less than (MAXVALUE,MAXVALUE));

Table created.

SQL> INSERT INTO tab5 values(2001,3,17, 4000);
1 row created.
SQL> INSERT INTO tab5 values(2001,11,1, 5500);
1 row created.
SQL> INSERT INTO tab5 values(2002,1,1, 4400);
1 row created.

SQL>  SELECT * FROM tab5 partition(p5);

      YEAR      MONTH        DAY AMOUNT_SOLD
---------- ---------- ---------- -----------
      2001         11          1        5500

SQL> SELECT * FROM tab5 partition(p2);

      YEAR      MONTH        DAY AMOUNT_SOLD
---------- ---------- ---------- -----------
      2001          3         17        4000

SQL> SELECT * FROM tab5 partition(p6);

      YEAR      MONTH        DAY AMOUNT_SOLD
---------- ---------- ---------- -----------
      2002          1          1        4400
```

----

6. Create a multicolumn partitioned table as directed: Table supplier_parts, storing the information about which suppliers deliver which parts. To distribute the data in equal-sized partitions, it is not sufficient to partition the table based on the supplier_id, because some suppliers might provide hundreds of thousands of parts, while others provide only a few specialty parts. Instead, you partition the table on (supplier_id, partnum) to manually enforce equal-sized partitions. Insert the following values
(5,5, 1000);
(5,150, 1000);
(10,100, 1000);

```
SQL> create table supplier_parts(
  2  sid number,
  3  pnum number,
  4  sold number)
  5  partition by range(sid,pnum)
  6  (partition p1 values less than(10,100),
  7   partition p2 values less than(20,200),
  8   partition future values less than(MAXVALUE,MAXVALUE));

Table created.
SQL> INSERT INTO supplier_parts values(5,5,1000);
1 row created.
SQL> INSERT INTO supplier_parts values(5,150,1000);
1 row created.
SQL> INSERT INTO supplier_parts values(10,100,1000);
1 row created.
SQL> INSERT INTO supplier_parts values(22,255,500);
```

```
1 row created.


SQL> select * from supplier_parts partition(p1);

       SID       PNUM       SOLD
---------- ---------- ----------
         5          5       1000
         5        150       1000

SQL> select * from supplier_parts partition(p2);

       SID       PNUM       SOLD
---------- ---------- ----------
        10        100       1000

SQL> select * from supplier_parts partition(future);

       SID       PNUM       SOLD
---------- ---------- ----------
        22        255        500
```
----

7. Create interval partitioned table as directed: Creates a table named- Sales consisting of four partitions, one for each quarter of sales. Each partition is given a name (sales_q1, sales_q2, ...) The columns for table must be prod_id, cust_id, promo_id, quantify sold, amount_sold all in number format and month in number format Perform interval partitioning on month and take interval of 01 months.

```
SQL> CREATE TABLE sales3
  2  ( prod_id number(6),
  3    cust_id number,
  4    promo_id number(3),
  5    amount_sold number(10,2),
  6    q_sold number,
  7    month number)
  8    partition by range(month)
  9    interval(1)
 10    (partition sales_q1 values less than(04),
 11    partition sales_q2 values less than(07),
 12    partition sales_q3 values less than(10));

Table created.

SQL>  insert into sales3 values (9,8,7,6,5,4);
1 row created.
SQL>   insert into sales3 values (1,8,3,4,5,5);
1 row created.
SQL>   insert into sales3 values (1,2,3,4,5,9);
1 row created.
SQL>   insert into sales3 values (11,56,3,8,5,1);
1 row created.
```

```
SQL>   select partition_name,tablespace_name,high_value,num_rows from user_tab_partitions where
table_name='SALES3';

PARTITION_NAME                 TABLESPACE_NAME
------------------------------ ------------------------------
HIGH_VALUE
--------------------------------------------------------------------------------
  NUM_ROWS
----------
SALES_Q1                       SYSTEM
04


SALES_Q2                       SYSTEM
07


PARTITION_NAME                 TABLESPACE_NAME
------------------------------ ------------------------------
HIGH_VALUE
--------------------------------------------------------------------------------
  NUM_ROWS
----------

SALES_Q3                       SYSTEM
10
```
----

8. Demonstrate reference partitioning as directed: Create parent table Orders with the attributes order_id, order_date, customer_id, shipper_id. Perform Range partitioning on Order Date. Take Range of 03 Months i.e. 01 quarter Create child table order_items with attributes order_id, product_id, price and quantity. Perform Reference partitioning on child table. Delete the created partitions.

```
SQL> create table orders
  2  ( order_id number primary key,
  3    order_date date ,
  4    customer_id number,
  5    shipper_id number)
  6    partition by range(order_date)
  7    ( PARTITION p1 VALUES LESS THAN (TO_DATE('01-apr-2011', 'DD-MON-YYYY')),
  8       PARTITION p2  VALUES LESS THAN (TO_DATE('01-jul-2011', 'DD-MON-YYYY')),
  9       PARTITION p3 VALUES LESS THAN (TO_DATE('01-oct-2011', 'DD-MON-YYYY')),
 10      PARTITION p4 VALUES LESS THAN (TO_DATE('01-jan-2012', 'DD-MON-YYYY')));

Table created.

SQL> create table order_items
  2  ( order_id number not null,
  3  product_id number primary key,
  4  price number,
  5  quantity number,
  6  constraint fo foreign key (order_id) references orders)
  7  partition by reference(fo);

Table created.

SQL> select table_name, partition_name
  2          from user_tab_partitions where table_name in ('ORDERS', 'ORDER_ITEMS');

TABLE_NAME                     PARTITION_NAME
------------------------------ ------------------------------
ORDERS                         P1
ORDERS                         P2
ORDERS                         P3
ORDERS                         P4
ORDER_ITEMS                    P1
ORDER_ITEMS                    P2
ORDER_ITEMS                    P3
ORDER_ITEMS                    P4

8 rows selected.

SQL> insert into orders values(11,'12-feb-2011',07,5);

1 row created.

SQL> insert into order_items values(11,1,160,150);

1 row created.

SQL> alter table orders drop partition p2;

Table altered.

SQL> select table_name, partition_name
  2          from user_tab_partitions where table_name in ('ORDERS', 'ORDER_ITEMS');

TABLE_NAME                     PARTITION_NAME
------------------------------ ------------------------------
ORDERS                         P1
ORDERS                         P3
ORDERS                         P4
ORDER_ITEMS                    P1
ORDER_ITEMS                    P3
ORDER_ITEMS                    P4

6 rows selected.
```

----

9. Implement virtual column based partitioning as below: Create table employee with attributes Emp_id, emp_name, fixed_salary, variable_salary. Generate Total salary as virtual colum. Perform range partitioning on Total Salary with four partitions as below: o Partition P1 stores salary less than 25000 o Partition P2 stores salary less than 50000 o Partition P3 stores salary less than 75000 o Partition P4 stores any salary above and equal to than 75000

```
SQL> create table employee
  2   ( emp_id number,
  3    emp_name varchar2(10),
  4    fixed_salary number,
  5    variable_salary number,
  6    total_salary number generated always as (fixed_salary+variable_salary)virtual
  7   )
  8   partition by range(total_salary)
  9   ( partition p1 values less than (25000),
 10     partition p2 values less than (50000),
 11     partition p3 values less than (75000),
 12     partition p4 values less than (maxvalue)
 13   );

Table created.

SQL>  insert into employee(emp_id,emp_name,fixed_salary,variable_salary) values(11,'Jack',35000,85000);

1 row created.

SQL> select * from employee;

    EMP_ID EMP_NAME   FIXED_SALARY VARIABLE_SALARY TOTAL_SALARY
---------- ---------- ------------ --------------- ------------

        11 Jack              35000           85000       120000

SQL> exec dbms_stats.gather_table_stats('system','CUSTOMER');
PL/SQL procedure successfully completed.
SQL> select partition_name,tablespace_name,high_value,num_rows from user_tab_partitions where
table_name='EMPLOYEE

PARTITION_NAME                 TABLESPACE_NAME
------------------------------ ------------------------------
HIGH_VALUE
--------------------------------------------------------------------------------
  NUM_ROWS
----------
P1                             SYSTEM
25000


P2                             SYSTEM
50000


PARTITION_NAME                 TABLESPACE_NAME
------------------------------ ------------------------------
```

```
HIGH_VALUE
--------------------------------------------------------------------------------
  NUM_ROWS
----------

P3                             SYSTEM
75000


P4                             SYSTEM
MAXVALUE

PARTITION_NAME                 TABLESPACE_NAME
------------------------------ ------------------------------
HIGH_VALUE
--------------------------------------------------------------------------------
  NUM_ROWS
----------

--------------------------------------------------------------------------------------------------
```

10. Demonstrate Composite partitioning technique as directed Implement range list partitioning for customer table having attributes cust_id, cust_name, cust_state, and time_id o Perform range partitioning on time-id and list partitioning on state attributes. Also create maxvalue and default partition for range and list partition respectively. o Partition definitions for range are as below:
 Partition old should accept values less than 01-Jan-2005
 Partition acquired should accept values less than 01-Jan-2010
 Partition recent should accept values less than 01-Jan-2015
 Partition unknown should accept values greater than 01-Jan-2015
o Partition definitions for list are as below:
 Partition west should accept values (‘MH’, ‘GJ’)
 Partition south should accept values (‘TN’, ‘AP’)
 Partition north should accept values (‘UP’, ‘HP’)
 Partition unknown should accept any other state.

```
SQL> create table customer
  2  ( cust_id number,
  3    cust_name varchar2(10),
  4    cust_state varchar2(10),
  5    time_id date)
  6  partition by range(time_id)
  7  subpartition by list(cust_state)
  8  subpartition template(
  9   subpartition west values('mh','gj'),
 10   subpartition south values('ap','tn'),
 11   subpartition north values('hp','up'),
 12   subpartition other values(default))
 13  ( partition old values less than(TO_DATE('01-jan-2005', 'DD-MON-YYYY')),
 14   partition acquired values less than(TO_DATE('01-jan-2010', 'DD-MON-YYYY')),
 15  partition recent values less than(TO_DATE('01-jan-2015', 'DD-MON-YYYY')),
 16  partition p1 values less than(maxvalue));

Table created.

SQL> insert into customer values(1,'JAck','mh','04-feb-2009');

1 row created.
```

```
SQL> exec dbms_stats.gather_table_stats('system','CUSTOMER');

PL/SQL procedure successfully completed.

SQL> select partition_name,tablespace_name,high_value,num_rows from user_tab_partitions where
table_name='CUSTOMER';

PARTITION_NAME                 TABLESPACE_NAME
------------------------------ ------------------------------
HIGH_VALUE
--------------------------------------------------------------------------------
  NUM_ROWS
----------
ACQUIRED                       SYSTEM
TO_DATE(' 2010-01-01 00:00:00', 'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIA
         1

OLD                            SYSTEM
TO_DATE(' 2005-01-01 00:00:00', 'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIA
         0

PARTITION_NAME                 TABLESPACE_NAME
------------------------------ ------------------------------
HIGH_VALUE
--------------------------------------------------------------------------------
  NUM_ROWS
----------

P1                             SYSTEM
MAXVALUE
         0

RECENT                         SYSTEM
TO_DATE(' 2015-01-01 00:00:00', 'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIA

PARTITION_NAME                 TABLESPACE_NAME
------------------------------ ------------------------------
HIGH_VALUE
--------------------------------------------------------------------------------
  NUM_ROWS
----------
         0
-------------------------------------------------------------------------------------------------------------
```

Rang-Range

```
-------------------------------------------------------------------------------------------------------------
SQL> create table cust2
  2  ( cust_id number,
  3    cust_name varchar2(10),
  4    cust_state varchar2(10),
  5    amount_sold number,
  6    time_id date)
  7  partition by range(time_id)
  8  subpartition by range(cust_id)
  9  subpartition template(
 10   subpartition s1 values less than(10),
 11  subpartition s2 values less than(20),
 12   subpartition other values less than(maxvalue))
 13  ( partition old values less than(TO_DATE('01-jan-2005', 'DD-MON-YYYY')),
 14   partition acquired values less than(TO_DATE('01-jan-2010', 'DD-MON-YYYY')),
 15  partition recent values less than(TO_DATE('01-jan-2015', 'DD-MON-YYYY')),
 16  partition p1 values less than(maxvalue));

Table created.

SQL>
SQL> insert into cust2 values(11,'jack','mh',8,'11-feb-2009');
1 row created.
SQL> exec dbms_stats.gather_table_stats('system','CUST2');

PL/SQL procedure successfully completed.

SQL>  select partition_name,tablespace_name,high_value,num_rows from user_tab_partitions where
table_name='CUST2';

PARTITION_NAME                 TABLESPACE_NAME
------------------------------ ------------------------------
HIGH_VALUE
--------------------------------------------------------------------------------
  NUM_ROWS
----------
ACQUIRED                       SYSTEM
TO_DATE(' 2010-01-01 00:00:00', 'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIA
         1

OLD                            SYSTEM
TO_DATE(' 2005-01-01 00:00:00', 'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIA
         0

PARTITION_NAME                 TABLESPACE_NAME
------------------------------ ------------------------------
HIGH_VALUE
--------------------------------------------------------------------------------
  NUM_ROWS
----------

P1                             SYSTEM
MAXVALUE
         0

RECENT                         SYSTEM
TO_DATE(' 2015-01-01 00:00:00', 'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIA

PARTITION_NAME                 TABLESPACE_NAME
------------------------------ ------------------------------
HIGH_VALUE
--------------------------------------------------------------------------------
  NUM_ROWS
----------
         0
select * from customer2 subpartition(acquired_s1);

   CUST_ID CUST_NAME  CUST_STATE AMOUNT_SOLD TIME_ID
---------- ---------- ---------- ----------- ---------
        11 jack       mh                   8 11-FEB-09
```

```
-------------------------------------------------------------------------------------------------------------
Range Hash

-------------------------------------------------------------------------------------------------------------

SQL> create table cust3
  2  ( cust_id number,
  3    cust_name varchar2(10),
  4    cust_state varchar2(10),
  5    amount_sold number,
  6    time_id date)
  7  partition by range(time_id)
  8  subpartition by hash(cust_id)
  9  subpartition template(
 10   subpartition h1,
 11    subpartition h2)
 12  ( partition old values less than(TO_DATE('01-jan-2005', 'DD-MON-YYYY')),
 13   partition acquired values less than(TO_DATE('01-jan-2010', 'DD-MON-YYYY')),
 14  partition recent values less than(TO_DATE('01-jan-2015', 'DD-MON-YYYY')),
 15  partition p1 values less than(maxvalue));

Table created.

SQL> insert into cust3 values(11,'jack','mh',5,'01-feb-2009');

1 row created.

SQL>
SQL> exec dbms_stats.gather_table_stats('system','CUST3');

PL/SQL procedure successfully completed.

SQL> select partition_name,tablespace_name,high_value,num_rows from user_tab_partitions where
table_name='CUST3';

PARTITION_NAME                 TABLESPACE_NAME
------------------------------ ------------------------------
HIGH_VALUE
--------------------------------------------------------------------------------
  NUM_ROWS
----------
ACQUIRED                       SYSTEM
TO_DATE(' 2010-01-01 00:00:00', 'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIA
         1

OLD                            SYSTEM
TO_DATE(' 2005-01-01 00:00:00', 'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIA
         0

PARTITION_NAME                 TABLESPACE_NAME
------------------------------ ------------------------------
HIGH_VALUE
--------------------------------------------------------------------------------
  NUM_ROWS
----------

P1                             SYSTEM
```

```
MAXVALUE
         0

RECENT                         SYSTEM
TO_DATE(' 2015-01-01 00:00:00', 'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIA

PARTITION_NAME                 TABLESPACE_NAME
------------------------------ ------------------------------
HIGH_VALUE
--------------------------------------------------------------------------------
  NUM_ROWS
----------
         0
select * from cust3 subpartition(acquired_h2);

 CUST_ID CUST_NAME  CUST_STATE AMOUNT_SOLD TIME_ID
---------- ---------- ---------- ----------- ---------
        11 jack     mh                   5 01-FEB-09
```

----

list hash

```
SQL> create table cust4
  2  ( cust_id number,
  3    cust_name varchar2(10),
  4    cust_state varchar2(10),
  5    amount_sold number,
  6    time_id date)
  7  partition by list(cust_state)
  8  subpartition by hash(cust_id)
  9  subpartition template(
 10   subpartition h1,
 11    subpartition h2)
 12  ( partition old values ('mh','tn'),
 13   partition acquired values ('cg','up'),
 14  partition recent values (default));

Table created.

SQL> insert into cust4 values(1,'jack','mh',5,'01-feb-2009');

1 row created.

SQL> exec dbms_stats.gather_table_stats('system','CUST4');

PL/SQL procedure successfully completed.

SQL> select partition_name,tablespace_name,high_value,num_rows from user_tab_partitions where
table_name='CUST4';

PARTITION_NAME                 TABLESPACE_NAME
------------------------------ ------------------------------
HIGH_VALUE
--------------------------------------------------------------------------------
```

```
  NUM_ROWS
----------
ACQUIRED                       SYSTEM
'cg', 'up'
         0

OLD                            SYSTEM
'mh', 'tn'
         1

PARTITION_NAME                 TABLESPACE_NAME
------------------------------ ------------------------------
HIGH_VALUE
--------------------------------------------------------------------------------
  NUM_ROWS
----------

RECENT                         SYSTEM
default
         0

SQL> select * from cust4 subpartition(old_h2);

   CUST_ID CUST_NAME  CUST_STATE AMOUNT_SOLD TIME_ID
---------- ---------- ---------- ----------- ---------
         1 jack       mh                   5 01-FEB-09
```

---

list list

```
SQL> create table cust5
  2  ( cust_id number,
  3    cust_name varchar2(10),
  4    cust_state varchar2(10),
  5    amount_sold number,
  6    time_id date)
  7  partition by list(cust_state)
  8  subpartition by list(cust_id)
  9  subpartition template(
 10   subpartition s1 values ('1','2'),
 11    subpartition s2 values(default))
 12  ( partition old values ('mh','tn'),
 13   partition acquired values ('cg','up'),
 14  partition recent values (default));

Table created.
SQL> insert into cust5 values(1,'jack','mh',5,'01-feb-2009');
1 row created.

SQL> exec dbms_stats.gather_table_stats('system','CUST5');
PL/SQL procedure successfully completed.
```

```
SQL> select partition_name,tablespace_name,high_value,num_rows from user_tab_partitions where
table_name='CUST5';

PARTITION_NAME                 TABLESPACE_NAME
------------------------------ ------------------------------
HIGH_VALUE
--------------------------------------------------------------------------------
  NUM_ROWS
----------
ACQUIRED                       SYSTEM
'cg', 'up'
         0

OLD                            SYSTEM
'mh', 'tn'
         0

PARTITION_NAME                 TABLESPACE_NAME
------------------------------ ------------------------------
HIGH_VALUE
--------------------------------------------------------------------------------
  NUM_ROWS
----------

RECENT                         SYSTEM
default
         0

SQL> select * from cust5 subpartition(old_s1);

   CUST_ID CUST_NAME  CUST_STATE AMOUNT_SOLD TIME_ID
---------- ---------- ---------- ----------- ---------
         1 jack       mh                   5 01-FEB-09

--------------------------------------------------------------------------------------------------

List Range
--------------------------------------------------------------------------------------------------



SQL>
SQL> create table cust6
  2  ( cust_id number,
  3    cust_name varchar2(10),
  4    cust_state varchar2(10),
  5    amount_sold number,
  6    time_id date)
  7  partition by list(cust_state)
  8  subpartition by range(cust_id)
  9  subpartition template(
 10   subpartition s1 values less than(5),
 11    subpartition s2 values less than(maxvalue))
 12  ( partition old values ('mh','tn'),
 13   partition acquired values ('cg','up'),
 14  partition recent values (default));

Table created.

SQL> insert into cust6 values(1,'Jack','mh',5,'01-feb-2009');

1 row created.

SQL> exec dbms_stats.gather_table_stats('system','CUST6');

PL/SQL procedure successfully completed.

SQL> select partition_name,tablespace_name,high_value,num_rows from user_tab_partitions where
table_name='CUST6';

PARTITION_NAME                 TABLESPACE_NAME
------------------------------ ------------------------------
HIGH_VALUE
--------------------------------------------------------------------------------
  NUM_ROWS
----------
ACQUIRED                       SYSTEM
'cg', 'up'
         0

OLD                            SYSTEM
'mh', 'tn'
         1

PARTITION_NAME                 TABLESPACE_NAME
------------------------------ ------------------------------
HIGH_VALUE
--------------------------------------------------------------------------------
  NUM_ROWS
----------

RECENT                         SYSTEM
default
         0


SQL> select * from cust6 subpartition(old_s1);

   CUST_ID CUST_NAME  CUST_STATE AMOUNT_SOLD TIME_ID
---------- ---------- ---------- ----------- ---------
         1 Jack       mh                   5 01-FEB-09
```