PRACTICAL NO 3

NAME →HARSH JAIN
ROLL NO → 55
BATCH → B2

/*Q1. 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 TSA1 DATAFILE 'E:\TEMP\TSA1.dbf' SIZE 10M;
CREATE TABLESPACE TSB2 DATAFILE 'E:\TEMP\TSB2.dbf' SIZE 10M;
CREATE TABLESPACE TSC3 DATAFILE 'E:\TEMP\TSC3.dbf' SIZE 10M;
CREATE TABLESPACE TSD4 DATAFILE 'E:\TEMP\TSD4.dbf' SIZE 10M;

SQL> CREATE TABLE SALES (
  2   PROD_ID NUMBER(6),
  3   CUST_ID NUMBER(10) ,
  4   TIME_ID DATE,
  5   CHANNEL_ID CHAR(1),
  6   PROMO_ID NUMBER(6),
  7   QUANTITY_SOLD NUMBER(3),
  8   AMOUNT_SOLD NUMBER(10)
  9   )
 10   PARTITION BY RANGE(TIME_ID)(
 11    PARTITION SALES_Q1 VALUES LESS THAN (TO_DATE('05-APR-2007','dd-MON-
yyyy'))TABLESPACE tsa1,
 12    PARTITION SALES_Q2 VALUES LESS THAN (TO_DATE('05-JUL-2007','dd-MON-
yyyy'))TABLESPACE tsb2,
 13    PARTITION SALES_Q3 VALUES LESS THAN (TO_DATE('05-OCT-2007','dd-MON-
yyyy'))TABLESPACE tsc3,
 14    PARTITION SALES_Q4 VALUES LESS THAN (TO_DATE('05-JAN-2008','dd-MON-
yyyy'))TABLESPACE tsd4
```

```
 15  );

Table created.

SQL> INSERT INTO SALES VALUES(1,1,'15-FEB-2007','A',101,10,4);

1 row created.

SQL> INSERT INTO SALES VALUES(2,2,'15-MAY-2007','B',200,11,5);

1 row created.

SQL> INSERT INTO SALES VALUES(3,3,'15-SEP-2007','C',10,12,6);

1 row created.

SQL> INSERT INTO SALES VALUES(4,4,'15-NOV-2007','C',102,13,6);

1 row created.

SQL> INSERT INTO SALES VALUES(5,5,'01-JAN-2008','A',10,101,5);

1 row created.

SQL> EXEC DBMS_STATS.GATHER_TABLE_STATS('CS55', 'SALES');

PL/SQL procedure successfully completed.

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                       TSA1
TO_DATE(' 2007-04-05 00:00:00', 'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIA
         1

SALES_Q2                       TSB2
TO_DATE(' 2007-07-05 00:00:00', 'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIA
         1
```

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

SALES_Q3                       TSC3
TO_DATE(' 2007-10-05 00:00:00', 'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIA
       1

SALES_Q4                       TSD4
TO_DATE(' 2008-01-05 00:00:00', 'SYYYY-MM-DD HH24:MI:SS', 'NLS_CALENDAR=GREGORIA

PARTITION_NAME                 TABLESPACE_NAME
------------------------------ ------------------------------
HIGH_VALUE
--------------------------------------------------------------------------------
 NUM_ROWS
----------
       2


/* Q2. 2. Create the same table as in Q1. With a different name with ENABLE ROW MOVEMENT */

SQL> CREATE TABLE SALES_EN (
  2   PROD_ID NUMBER(6),
  3   CUST_ID NUMBER(10) ,
  4   TIME_ID DATE,
  5   CHANNEL_ID CHAR(1),
  6   PROMO_ID NUMBER(6),
  7   QUANTITY_SOLD NUMBER(3),
  8   AMOUNT_SOLD NUMBER(10)
  9   )
 10    PARTITION BY RANGE(TIME_ID)(
 11    PARTITION SALES_Q1 VALUES LESS THAN (TO_DATE('05-APR-2007','dd-MON-yyyy'))
TABLESPACE tsa1,
 12    PARTITION SALES_Q2 VALUES LESS THAN (TO_DATE('05-JUL-2007','dd-MON-
yyyy'))TABLESPACE tsb2,
 13    PARTITION SALES_Q3 VALUES LESS THAN (TO_DATE('05-OCT-2007','dd-MON-
yyyy'))TABLESPACE tsc3,
 14    PARTITION SALES_Q4 VALUES LESS THAN (TO_DATE('05-JAN-2008','dd-MON-
yyyy'))TABLESPACE tsd4
 15   )
 16  ENABLE ROW MOVEMENT;
```

```
Table created.

SQL> INSERT INTO SALES_EN VALUES(1,1,'15-FEB-2007','A',101,10,4);

1 row created.

SQL> INSERT INTO SALES_EN VALUES(2,2,'15-MAY-2007','B',200,11,5);

1 row created.

SQL> INSERT INTO SALES_EN VALUES(3,3,'15-SEP-2007','C',10,12,6);

1 row created.

SQL> INSERT INTO SALES_EN VALUES(4,4,'15-NOV-2007','C',102,13,6);

1 row created.

SQL> INSERT INTO SALES_EN VALUES(5,5,'01-JAN-2008','A',10,101,5);

1 row created.

SQL> EXEC DBMS_STATS.GATHER_TABLE_STATS('CS55', 'SALES_EN');

PL/SQL procedure successfully completed.

SQL> SELECT PARTITION_NAME,TABLESPACE_NAME,NUM_ROWS FROM USER_TAB_PARTITIONS
  2  WHERE UPPER(TABLE_NAME)='SALES_EN';

PARTITION_NAME                 TABLESPACE_NAME                NUM_ROWS
------------------------------ ------------------------------ ----------
SALES_Q1                       TSA1                                    1
SALES_Q2                       TSB2                                    1
SALES_Q3                       TSC3                                    1
SALES_Q4                       TSD4                                    2



SQL> UPDATE SALES_EN
  2  SET TIME_ID='01-OCT-2007'
  3  WHERE PROD_ID=5;

1 row updated.
```

```
SQL> SELECT PARTITION_NAME,TABLESPACE_NAME,NUM_ROWS FROM USER_TAB_PARTITIONS
  2  WHERE UPPER(TABLE_NAME)='SALES_EN';

PARTITION_NAME                 TABLESPACE_NAME                NUM_ROWS
------------------------------ ------------------------------ ----------
SALES_Q1                       TSA1                                    1
SALES_Q2                       TSB2                                    1
SALES_Q3                       TSC3                                    2
SALES_Q4                       TSD4                                    1

SQL>
```

```
/*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')
*/

CREATE TABLE DEPT
(
DEPT_NO NUMBER(2),
DEPT_NAME VARCHAR(30),
QUAT_SALES NUMBER(10,2),
STATE VARCHAR(2))
PARTITION BY LIST (STATE)
(
PARTITION NORTHWEST VALUES ('OR','WA'),
PARTITION SOUTHWEST VALUES ('AZ','UT','NM'),
PARTITION NORTHEAST VALUES ('NY','VM','NJ'),
PARTITION SOUTHEAST VALUES ('FL','GA'),
PARTITION NORTHCENTRAL VALUES ('SD','WI'),
PARTITION SOUTHCENTRAL VALUES ('OK','TX'));

INSERT INTO DEPT VALUES (10, 'accounting', 100, 'WA');
```

```
1 row created.

INSERT INTO DEPT VALUES (20, 'R&D', 150, 'OR') ;
1 row created.

INSERT INTO DEPT VALUES (30, 'sales', 100, 'FL')   ;
1 row created.

INSERT INTO DEPT VALUES (40, 'HR', 10, 'TX');
1 row created.

INSERT INTO DEPT VALUES (50, 'systems engineering', 10, 'CA') ;
ERROR at line 1:
ORA-14400: inserted partition key does not map to any partition

SQL> SELECT PARTITION_NAME, TABLESPACE_NAME, HIGH_VALUE, NUM_ROWS FROM
USER_TAB_PARTITIONS WHERE TABLE_NAME =  'DEPT';

PARTITION_NAME              TABLESPACE_NAME             HIGH_VALUE
NUM_ROWS
------------------------------ ------------------------------ ----------------------------------------------------------------------------
-------- ----------
NORTHWEST                 USERS                  'OR', 'WA'                                                    1
SOUTHWEST                 USERS                  'AZ', 'UT', 'NM'                                               0
NORTHEAST                 USERS                 'NY', 'VM', 'NJ'                                               0
SOUTHEAST                 USERS                 'FL', 'GA'                                                   0
NORTHCENTRAL                USERS                   'SD', 'WI'                                                0
SOUTHCENTRAL                USERS                   'OK', 'TX'                                                1

6 rows selected.


/* 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.
*/
--------------------------------------------------------------------
CREATE TABLE EMP
(
                EMP_NO NUMBER,
                JOB VARCHAR(2),
                SAL NUMBER(10,2),
```

```
DEPTNO NUMBER(4),
PARTITION  BY HASH (EMP_NO)
PARTITIONS 5);

 SELECT PARTITION_NAME, TABLESPACE_NAME, HIGH_VALUE, NUM_ROWS FROM
USER_TAB_PARTITIONS WHERE TABLE_NAME =  'EMP';


                        PARTITION_NAME              TABLESPACE_NAME
------------------------------ ------------------------------
HIGH_VALUE
--------------------------------------------------------------------------------
 NUM_ROWS
----------
SYS_P21                 USERS



SYS_P22                 USERS



SYS_P23                 USERS



SYS_P24                 USERS



SYS_P25                 USERS
```

/*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:
  - Before 2001: Less than jan 2001
  - Less than april 2001
  - Less than july 2001
  - Les than oct 2001
  - Less than jan 2002
  - 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.

*/

```
CREATE TABLE DATE_TEST (
        YEAR1 NUMBER(4),
        MONTH1 NUMBER(2),
        DAY1 NUMBER(2),
        AMOUNT_SOLD NUMBER(10,2))
        PARTITION BY RANGE(YEAR1, MONTH1)(
                PARTITION PARTA VALUES LESS THAN (2001, 1),
                PARTITION PARTB VALUES LESS THAN (2001, 4),
                PARTITION PARTC VALUES LESS THAN (2001, 10),
                PARTITION PARTD VALUES LESS THAN (2002, 1),
                PARTITION PARTE VALUES LESS THAN (MAXVALUE, MAXVALUE));

        INSERT INTO DATE_TEST VALUES (2001,3,17, 2000);
        INSERT INTO DATE_TEST VALUES (2001,11,1, 5000);
        INSERT INTO DATE_TEST VALUES (2002,1,1, 4000);

        SQL> EXEC DBMS_STATS.GATHER_TABLE_STATS('cs55', 'DATE_TEST');
PL/SQL procedure successfully completed.

SQL> SELECT PARTITION_NAME, TABLESPACE_NAME, HIGH_VALUE, NUM_ROWS FROM
USER_TAB_PARTITIONS WHERE TABLE_NAME =  'DATE_TEST';

PARTITION_NAME              TABLESPACE_NAME            HIGH_VALUE
NUM_ROWS
------------------------------ ------------------------------ ----------------------------------------------------------------------------
-------- ----------
PARTA                   USERS                   2001, 1                                                                      0
PARTB                   USERS                   2001, 4                                                                      1
PARTC                   USERS                   2001, 10                                                                     0
PARTD                   USERS                   2002, 1                                                                      1
PARTE                   USERS                   MAXVALUE, MAXVALUE
```

/*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);
*/

```
CREATE TABLE SUPPLIER_PARTS (
SUPPLIER_ID NUMBER(2),
PART_NUM NUMBER(4),
QUANTITY NUMBER)
PARTITION BY RANGE(SUPPLIER_ID, PART_NUM)(
PARTITION PARTA VALUES LESS THAN (5,50),
PARTITION PARTB VALUES LESS THAN (10,100),
PARTITION PARTC VALUES LESS THAN (20,200));

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> exec dbms_stats.gather_table_stats('ASHISH','supplier_parts');

PL/SQL procedure successfully completed.
```

select partition_name,TABLESPACE_NAME,NUM_ROWS from user_tab_partitions where table_name='SUPPLIER_PARTS';

PARTITION_NAME	TABLESPACE_NAME	NUM_ROWS
PARTA	USERS	1
PARTB	USERS	1
PARTC	USERS	2

/*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.

*/

```
CREATE TABLE SALESTAB (
                        PROD_ID NUMBER(4),
                        CUST_ID NUMBER(4),
                        PROMO_ID NUMBER(4),
                        TIME_ID DATE,
                        QUANTITY_SOLD NUMBER(4),
                        AMOUNT_SOLD NUMBER(5),
                        MONTHS NUMBER )
                        PARTITION BY RANGE(TIME_ID)
                        INTERVAL(NUMTOYMINTERVAL(1, 'MONTH'))
                        ( PARTITION SALES_Q1 VALUES LESS THAN (TO_DATE('1-4-2005','DD-MM-
YYYY')),
                          PARTITION SALES_Q2 VALUES LESS THAN (TO_DATE('1-7-2005','DD-MM-
YYYY')),
                          PARTITION SALES_Q3 VALUES LESS THAN (TO_DATE('1-10-2005','DD-MM-
YYYY')),
                          PARTITION SALES_Q4 VALUES LESS THAN (TO_DATE('1-1-2006','DD-MM-
YYYY')));

INSERT INTO SALESTAB VALUES (1001, 1001, 1001, (TO_DATE('2-3-2005', 'DD-MM-YYYY')), 10,
10000, 3);
SQL> INSERT INTO SALESTAB VALUES (1006, 1006, 1006, (TO_DATE('5-6-2005', 'DD-MM-YYYY')), 10,
10000, 7);
1 row created.
SQL> INSERT INTO SALESTAB VALUES (1007, 1007, 1007, (TO_DATE('5-6-2005', 'DD-MM-YYYY')), 10,
10000, 10);
1 row created.
SQL> INSERT INTO SALESTAB VALUES (1008, 1008, 1008, (TO_DATE('5-6-2005', 'DD-MM-YYYY')), 10,
10000, 11);
1 row created.
SQL> EXEC DBMS_STATS.GATHER_TABLE_STATS('ASHISH', 'SALESTAB');
```

PL/SQL procedure successfully completed.

SQL> select partition_name,TABLESPACE_NAME,NUM_ROWS from user_tab_partitions where table_name='SALESTAB';

PARTITION_NAME	TABLESPACE_NAME	NUM_ROWS
SALES_Q1	USERS	1
SALES_Q2	USERS	4
SALES_Q3	USERS	0
SALES_Q4	USERS	3

1

/*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.

*/

```
CREATE TABLE ORDERS (
                ORDERID NUMBER(5) PRIMARY KEY,
                ORDERDATE DATE,
                CUSTID NUMBER,
                SHIPPERID NUMBER(5))
                PARTITION BY RANGE(ORDERDATE)
                ( PARTITION Q1 VALUES LESS THAN (TO_DATE('1-4-2005','DD-MM-YYYY')),
                  PARTITION Q2 VALUES LESS THAN (TO_DATE('1-7-2005','DD-MM-YYYY')),
                  PARTITION Q3 VALUES LESS THAN (TO_DATE('1-10-2005','DD-MM-YYYY')),
                  PARTITION Q4 VALUES LESS THAN (TO_DATE('1-1-2006','DD-MM-YYYY')));

                CREATE TABLE ORDER_ITEMS(
                        ORDERID NUMBER(5) PRIMARY KEY,
                        PRODUCT_ID NUMBER(5),
                        PRICE NUMBER(5),
                        QUANTITY NUMBER(3),
                        CONSTRAINT ORDER_ITEM_FK FOREIGN KEY (ORDERID)
REFERENCES ORDERS
                )
                PARTITION BY REFERENCE (ORDER_ITEM_FK);
```

INSERT INTO ORDERS VALUES (111, TO_DATE('2-3-2005','DD-MM-YYYY'),500,20);

INSERT INTO ORDER_ITEMS VALUES (111,300,5000,20);

select partition_name,TABLESPACE_NAME,NUM_ROWS from user_tab_partitions where table_name='LIST_HASH';

ALTER TABLE ORDERS DROP PARTITION Q1;

SQL> exec dbms_stats.gather_table_stats('cs55','ORDER_ITEMS');

PL/SQL procedure successfully completed.

SQL> exec dbms_stats.gather_table_stats('cs55','ORDERS');

PL/SQL procedure successfully completed.

SQL> select partition_name,TABLESPACE_NAME,NUM_ROWS from user_tab_partitions where table_name='ORDER_ITEMS';

PARTITION_NAME	TABLESPACE_NAME	NUM_ROWS
Q1	USERS	1
Q2	USERS	0
Q3	USERS	0
Q4	USERS	0

SQL> select partition_name,TABLESPACE_NAME,NUM_ROWS from user_tab_partitions where table_name='ORDERS';

PARTITION_NAME	TABLESPACE_NAME	NUM_ROWS
Q1	USERS	1
Q2	USERS	0
Q3	USERS	0
Q4	USERS	0

SQL> ALTER TABLE ORDERS DROP PARTITION Q1;
Table altered.

/*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
*/
---------------------------------------------------------------------

CREATE TABLE EMPLOYEE (
        EMP_ID NUMBER(5),
        EMP_NAME VARCHAR(50),
        FIXED_SALARY NUMBER(5),
        VARIABLE_SALARY NUMBER(5),
        TOTAL_SALARY NUMBER(5) GENERATED ALWAYS AS (FIXED_SALARY + VARIABLE_SALARY )
VIRTUAL )
        PARTITION BY RANGE(TOTAL_SALARY)
        ( PARTITION P1 VALUES LESS THAN (25000),
          PARTITION P2 VALUES LESS THAN (50000),
          PARTITION P3 VALUES LESS THAN (75000),
          PARTITION P4 VALUES LESS THAN (MAXVALUE));

          INSERT INTO EMPLOYEE (EMP_ID, EMP_NAME, FIXED_SALARY, VARIABLE_SALARY) VALUES
(100, 'hornet', 24000, 10000);
          INSERT INTO EMPLOYEE (EMP_ID, EMP_NAME, FIXED_SALARY, VARIABLE_SALARY) VALUES
(101, 'SMITH', 34000, 10000);


EXEC DBMS_STATS.GATHER_TABLE_STATS('CS55', 'EMPLOYEE');


SELECT PARTITION_NAME,TABLESPACE_NAME,NUM_ROWS FROM USER_TAB_PARTITIONS
WHERE UPPER(TABLE_NAME)='EMPLOYEE';

SQL> EXEC DBMS_STATS.GATHER_TABLE_STATS('CS55', 'EMPLOYEE');

PL/SQL procedure successfully completed.

SQL> SELECT * FROM EMPLOYEE;

   EMP_ID EMP_NAME                                    FIXED_SALARY
---------- -------------------------------------------------- ------------
VARIABLE_SALARY TOTAL_SALARY
--------------- ------------
       100 hornet                                         24000
         10000       34000
```

101 SMITH 34000
10000 44000

SQL> SELECT PARTITION_NAME,TABLESPACE_NAME,NUM_ROWS FROM USER_TAB_PARTITIONS
2 WHERE UPPER(TABLE_NAME)='EMPLOYEE';

PARTITION_NAME	TABLESPACE_NAME	NUM_ROWS
P4	USERS	0
P3	USERS	0
P2	USERS	2
P1	USERS	0

*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-Jan2010
- Partition recent should accept values less than 01-Jan-2015
- Partition unknown should accept values greater than 01-Jan2015

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(5),
 3  CUST_NAME VARCHAR(50),
 4  CUST_STATE VARCHAR(2),
 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 ('TN', 'AP'),
 11  SUBPARTITION north VALUES ('UP', 'HP'),
 12  SUBPARTITION unaccepted VALUES (default))
 13  (PARTITION OLD VALUES LESS THAN (TO_DATE('1/01/2005','DD/MM/YYYY')),
 14  PARTITION ACQUIRED VALUES LESS THAN (TO_DATE('1/01/2010','DD/MM/YYYY')),
 15  PARTITION RECENT VALUES LESS THAN (TO_DATE('1/01/2015','DD/MM/YYYY')),
 16  PARTITION UNKNOWN VALUES LESS THAN (MAXVALUE));

Table created.

SQL> INSERT INTO CUSTOMER VALUES (1001, 'JAMES', 'MH', TO_DATE('1/04/2005','DD/MM/YYYY'));

1 row created.

SQL> INSERT INTO CUSTOMER VALUES (1002, 'WILLIAM', 'GJ',
TO_DATE('1/04/2011','DD/MM/YYYY'));

1 row created.

SQL> INSERT INTO CUSTOMER VALUES (1001, 'JOE', 'UP', TO_DATE('1/04/2014','DD/MM/YYYY'));

1 row created.

SQL> INSERT INTO CUSTOMER VALUES (1003, 'SMITH', 'UP', TO_DATE('1/04/2014','DD/MM/YYYY'));

1 row created.

SQL> EXEC DBMS_STATS.GATHER_TABLE_STATS('CS55', 'CUSTOMER');

PL/SQL procedure successfully completed.

SQL> select subpartition_name,TABLESPACE_NAME,NUM_ROWS from user_tab_subpartitions
where table_name='CUSTOMER';

SUBPARTITION_NAME            TABLESPACE_NAME                NUM_ROWS
------------------------------ ------------------------------ ----------
OLD_WEST                     USERS                               0
OLD_SOUTH                    USERS                               0
OLD_NORTH                    USERS                               0
OLD_UNACCEPTED               USERS                               0
ACQUIRED_WEST                USERS                               1
ACQUIRED_SOUTH               USERS                               0
ACQUIRED_NORTH               USERS                               0
ACQUIRED_UNACCEPTED          USERS                               0
RECENT_WEST                  USERS                               1
```

```
RECENT_SOUTH               USERS                          0
RECENT_NORTH               USERS                          2

SUBPARTITION_NAME          TABLESPACE_NAME                NUM_ROWS
------------------------------ ------------------------------ ----------
RECENT_UNACCEPTED          USERS                           0
UNKNOWN_WEST               USERS                          0
UNKNOWN_SOUTH              USERS                           0
UNKNOWN_NORTH              USERS                           0
UNKNOWN_UNACCEPTED         USERS                          0

16 rows selected.

SQL> select * from customer subpartition(RECENT_NORTH);

  CUST_ID CUST_NAME                                          CU TIME_ID
---------- -------------------------------------------------- -- ---------
     1001 JOE                                              UP 01-APR-14
     1003 SMITH                                              UP 01-APR-14


SQL> select * from customer subpartition(RECENT_NORTH);

  CUST_ID CUST_NAME                                          CU TIME_ID
---------- -------------------------------------------------- -- ---------
     1001 JOE                                              UP 01-APR-14
     1003 SMITH                                              UP 01-APR-14

SQL> INSERT INTO CUSTOMER VALUES (1003, 'SMITH', 'UP', TO_DATE('1/04/2014','DD/MM/YYYY'));

1 row created.

SQL> select * from customer subpartition(RECENT_NORTH);

  CUST_ID CUST_NAME                                          CU TIME_ID
---------- -------------------------------------------------- -- ---------
     1001 JOE                                              UP 01-APR-14
     1003 SMITH                                              UP 01-APR-14
     1003 SMITH                                              UP 01-APR-14

/*QUERY 11-15

        CREATE FOUR TABLESPACES PART 1, PART 2 , PART 3 AND PART 4.
        CREATE PARTITIONING ON THE SCHEMA OF PROBLEM NO 10.
```

```
        PERFORM FOLLOWING PARTITIONING :
                1. RANGE HASH
                2. RANGE RANGE
                3. LIST LIST
                4. LIST RANGE
                5. LIST HASH
*/

CREATE TABLESPACE TSA1 DATAFILE 'E:\TEMP\TSA1.dbf' SIZE 10M;
CREATE TABLESPACE TSB2 DATAFILE 'E:\TEMP\TSB2.dbf' SIZE 10M;
CREATE TABLESPACE TSC3 DATAFILE 'E:\TEMP\TSC3.dbf' SIZE 10M;
CREATE TABLESPACE TSD4 DATAFILE 'E:\TEMP\TSD4.dbf' SIZE 10M;
-----------------------------------------------------------------
--RANGE -RANGE PARTITIONING :
-----------------------------------------------------------------
CREATE TABLE COMPOSITE_RANGE (
        CUST_ID NUMBER(10),
        CUST_NAME VARCHAR2(25),
        CUST_STATE VARCHAR2(2),
        TIME_ID DATE)
        PARTITION BY RANGE (TIME_ID)
        SUBPARTITION BY RANGE (CUST_ID)
        SUBPARTITION TEMPLATE (
                SUBPARTITION ORIGINAL VALUES LESS THAN(1001)
                TABLESPACE TSA1,
                SUBPARTITION ACQUIRED VALUES LESS THAN(2001)
                TABLESPACE TSB2,
                SUBPARTITION GAIN VALUES LESS THAN(5001)
                TABLESPACE TSC3,
                SUBPARTITION RECENT VALUES LESS THAN(MAXVALUE)
                TABLESPACE TSD4)
                (PARTITION PART1 VALUES LESS THAN (TO_DATE('1/01/2005','DD/MM/YYYY')),
                PARTITION PART2 VALUES LESS THAN (TO_DATE('1/01/2010','DD/MM/YYYY')),
                PARTITION PART3 VALUES LESS THAN (TO_DATE('1/01/2015','DD/MM/YYYY')),
                PARTITION FUTURE VALUES LESS THAN (MAXVALUE));

                INSERT INTO COMPOSITE_RANGE VALUES ( 1001, 'HARSH', 'MH',
TO_DATE('1/04/2014','DD/MM/YYYY'));
                INSERT INTO COMPOSITE_RANGE VALUES ( 1002, 'SUNSPOT', 'MH',
TO_DATE('1/08/2004','DD/MM/YYYY'));
                INSERT INTO COMPOSITE_RANGE VALUES ( 1003, 'SHINCHAN', 'AP',
TO_DATE('1/08/2010','DD/MM/YYYY'));

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

```
PART1                   USERS                          1
PART2                   USERS                          0
PART3                   USERS                          2
FUTURE                   USERS                          0
-------------------------------------------------------------------------
--LIST HASH
-------------------------------------------------------------------------
CREATE TABLE LIST_HASH (
        CUST_ID NUMBER(10),
        CUST_NAME VARCHAR2(25),
        CUST_STATE VARCHAR2(2),
        TIME_ID DATE)
        PARTITION BY LIST (CUST_STATE)
        SUBPARTITION BY HASH (CUST_ID)
        SUBPARTITION TEMPLATE (
                SUBPARTITION SP1 TABLESPACE TSA1,
                SUBPARTITION SP2 TABLESPACE TSB2,
                SUBPARTITION SP3 TABLESPACE TSC3,
                SUBPARTITION SP4 TABLESPACE TSD4)
                (PARTITION WEST VALUES ('OR','WA'),
                 PARTITION EAST VALUES ('AZ','UT','NM'),
                 PARTITION NORTH VALUES ('NY','VM','NJ'),
                 PARTITION SOUTH VALUES ('FL','GA'));




                INSERT INTO LIST_HASH VALUES ( 1001, 'HARSH', 'OR',
TO_DATE('1/04/2014','DD/MM/YYYY'));
                INSERT INTO LIST_HASH VALUES ( 1002, 'SUNSPOT', 'NM',
TO_DATE('1/08/2004','DD/MM/YYYY'));
                INSERT INTO LIST_HASH VALUES ( 1003, 'SHINCHAN', 'NY',
TO_DATE('1/08/2010','DD/MM/YYYY'));
                INSERT INTO LIST_HASH VALUES ( 1003, 'JB', 'WA',
TO_DATE('1/08/2010','DD/MM/YYYY'));


SQL> select partition_name,TABLESPACE_NAME,NUM_ROWS from user_tab_partitions where
table_name='LIST_HASH';
PARTITION_NAME              TABLESPACE_NAME              NUM_ROWS
------------------------------ ------------------------------ ----------
WEST                    USERS                          2
EAST                   USERS                         1
NORTH                    USERS                          1
SOUTH                    USERS                          0
```

```
-------------------------------------------------------------------------
--LIST LIST
-------------------------------------------------------------------------
CREATE TABLE LIST_LIST (
        CUST_ID NUMBER(10),
        CUST_NAME VARCHAR2(25),
        CUST_STATE VARCHAR2(2),
        TIME_ID DATE)
        PARTITION BY LIST (CUST_STATE)
        SUBPARTITION BY LIST (CUST_ID)
        SUBPARTITION TEMPLATE (
                SUBPARTITION SP1 VALUES (1,3,5) TABLESPACE TSA1,
                SUBPARTITION SP2 VALUES (2,4,6) TABLESPACE TSB2,
                SUBPARTITION SP3 VALUES (7,8,9,0) TABLESPACE TSC3)
                (PARTITION WEST VALUES ('OR','WA'),
                 PARTITION EAST VALUES ('AZ','UT','NM'),
                 PARTITION NORTH VALUES ('NY','VM','NJ'),
                 PARTITION SOUTH VALUES ('FL','GA'));

                INSERT INTO LIST_LIST VALUES ( 1, 'HARSH', 'OR',
TO_DATE('1/04/2014','DD/MM/YYYY'));
                INSERT INTO LIST_LIST VALUES ( 2, 'SUNSPOT', 'NM',
TO_DATE('1/08/2004','DD/MM/YYYY'));
                INSERT INTO LIST_LIST VALUES ( 3, 'SHINCHAN', 'NY',
TO_DATE('1/08/2010','DD/MM/YYYY'));
                INSERT INTO LIST_LIST VALUES ( 3, 'JB', 'WA',
TO_DATE('1/08/2010','DD/MM/YYYY'));

                select partition_name,TABLESPACE_NAME,NUM_ROWS from user_tab_partitions
where table_name='LIST_LIST';

                SQL> select partition_name,TABLESPACE_NAME,NUM_ROWS from
user_tab_partitions where table_name='LIST_LIST';
PARTITION_NAME              TABLESPACE_NAME               NUM_ROWS
------------------------------ ------------------------------ ----------
WEST                    USERS                          2
EAST                   USERS                         1
NORTH                    USERS                          1
SOUTH                    USERS                          0
-------------------------------------------------------------------------
--RANGE HASH
-------------------------------------------------------------------------
CREATE TABLE RANGE_HASH (
        CUST_ID NUMBER(10),
        CUST_NAME VARCHAR2(25),
```

```
        CUST_STATE VARCHAR2(10),
        TIME_ID DATE)
        PARTITION BY RANGE (TIME_ID)
        SUBPARTITION BY HASH (CUST_ID)
        SUBPARTITION TEMPLATE (
                SUBPARTITION SP1  TABLESPACE TSA1,
                SUBPARTITION SP2  TABLESPACE TSB2,
                SUBPARTITION SP3  TABLESPACE TSC3,
                SUBPARTITION SP4  TABLESPACE TSD4)
                (PARTITION PART1 VALUES LESS THAN (TO_DATE('1/01/2005','DD/MM/YYYY')),
                PARTITION PART2 VALUES LESS THAN (TO_DATE('1/01/2010','DD/MM/YYYY')),
                PARTITION PART3 VALUES LESS THAN (TO_DATE('1/01/2015','DD/MM/YYYY')),
                PARTITION FUTURE VALUES LESS THAN (MAXVALUE));


                INSERT INTO RANGE_HASH
                SELECT C.CUST_ID , C.CUST_FIRST_NAME || ' ' ||
                C.CUST_LAST_NAME,
                S.AMOUNT_SOLD, S.TIME_ID
                FROM SH.SALES S, SH.CUSTOMERS C
                WHERE S.CUST_ID = C.CUST_ID
                AND ROWNUM < 250001;


SQL>  EXEC DBMS_STATS.GATHER_TABLE_STATS('cs55', 'RANGE_HASH');
PL/SQL procedure successfully completed.
SQL> select partition_name,TABLESPACE_NAME,NUM_ROWS from user_tab_partitions where
table_name='RANGE_HASH';
PARTITION_NAME              TABLESPACE_NAME              NUM_ROWS
--------------------------- ------------------------------ ----------
PART1                  USERS                          2280
PART2                  USERS                          1300
PART3                  USERS                         10980
FUTURE                  USERS                         12720
-----------------------------------------------------------------------
--LIST RANGE
-----------------------------------------------------------------------
CREATE TABLE LIST_RANGE (
        CUST_ID NUMBER(10),
        CUST_NAME VARCHAR2(25),
        CUST_STATE VARCHAR2(2),
        TIME_ID DATE)
        PARTITION BY RANGE (CUST_ID)
        SUBPARTITION BY LIST (CUST_STATE)
        SUBPARTITION TEMPLATE (
```

```
                SUBPARTITION SP1 VALUES ('OR','WA') TABLESPACE TSA1,
                SUBPARTITION SP2 VALUES ('MH','NY')TABLESPACE TSB2,
                SUBPARTITION SP3 VALUES ('FL','GA') TABLESPACE TSC3,
                SUBPARTITION SP4 VALUES ('VM','NJ') TABLESPACE TSD4)
                (PARTITION ORIGINAL VALUES LESS THAN(1001),
                 PARTITION OTHER VALUES LESS THAN(2001),
                 PARTITION RECENT VALUES LESS THAN(3001),
                 PARTITION NEWEST VALUES LESS THAN(4001));
                INSERT INTO LIST_RANGE VALUES ( 1001, 'HARSH', 'OR',
TO_DATE('1/04/2014','DD/MM/YYYY'));
                INSERT INTO LIST_RANGE VALUES ( 2002, 'SUNSPOT', 'FL',
TO_DATE('1/08/2004','DD/MM/YYYY'));
                INSERT INTO LIST_RANGE VALUES ( 3003, 'SHINCHAN', 'VM',
TO_DATE('1/08/2010','DD/MM/YYYY'));
                INSERT INTO LIST_RANGE VALUES ( 4000, 'JB', 'NY',
TO_DATE('1/08/2010','DD/MM/YYYY'));

SQL> EXEC DBMS_STATS.GATHER_TABLE_STATS('cs55', 'LIST_RANGE');
PL/SQL procedure successfully completed.
SQL> select partition_name,TABLESPACE_NAME,NUM_ROWS from user_tab_partitions where
table_name='LIST_RANGE';
PARTITION_NAME              TABLESPACE_NAME              NUM_ROWS
------------------------------ ------------------------------ ----------
ORIGINAL                    USERS                         0
OTHER                       USERS                         1
RECENT                      USERS                         1
NEWEST                      USERS                         2
```