

```

1 .....prac3.....
2
3 QUERY 1 :
4     Write a query to create range portioned table:
5     Creates a table named- Sales consisting of four partitions, one for each quarter of
6     sales. The columns sale_year, sale_month, and sale_day are the partitioning columns,
7     while their values constitute the partitioning key of a specific row.
8     Each partition is given a name (sales_q1, sales_q2, ...), and each partition is
9     contained in a separate tablespace (tsa, tsb, ...)
10    The columns for table must be prod_id, cust_id, promo_id, quantify sold, amount_sold -
11    all in number format and time_id.
12
13 CREATE TABLESPACE ABC1 DATAFILE 'F:/APP/DELL/ORADATA/ORCL/abc1.dbf' SIZE 10M;
14 Tablespace created.
15 CREATE TABLESPACE ABC2 DATAFILE 'F:/APP/DELL/ORADATA/ORCL/abc2.dbf' SIZE 10M;
16 Tablespace created.
17 CREATE TABLESPACE ABC3 DATAFILE 'F:/APP/DELL/ORADATA/ORCL/abc3.dbf' SIZE 10M;
18 Tablespace created.
19 CREATE TABLESPACE ABC4 DATAFILE 'F:/APP/DELL/ORADATA/ORCL/abc4.dbf' SIZE 10M;
20 Tablespace created.
21
22 CREATE TABLE SALES(
23     PROD_ID NUMBER(5) NOT NULL,
24     CUST_ID NUMBER NOT NULL,
25     PROMO_ID NUMBER(5) NOT NULL,
26     QUANTITY_SOLD NUMBER(5) NOT NULL,
27     AMOUNT_SOLD NUMBER(5) NOT NULL,
28     TIME_ID DATE NOT NULL)
29     PARTITION BY RANGE(TIME_ID)
30     (PARTITION SALES_Q1 VALUES LESS THAN('01-APR-2017') TABLESPACE ABC1,
31     PARTITION SALES_Q2 VALUES LESS THAN('01-JUL-2017') TABLESPACE ABC2,
32     PARTITION SALES_Q3 VALUES LESS THAN('01-OCT-2017') TABLESPACE ABC3,
33     PARTITION SALES_Q4 VALUES LESS THAN('01-JAN-2018') TABLESPACE ABC4
34     );
35
36 Table created.
37
38 INSERT INTO SALES VALUES(10001,10001,10001,25,30000,'20-JAN-2017');
39 1 row created.
40 INSERT INTO SALES VALUES(10002,10002,10002,55,40000,'20-MAY-2017');
41 1 row created.
42 INSERT INTO SALES VALUES(10003,10003,10003,50,80000,'20-JUL-2017');
43 1 row created.
44 INSERT INTO SALES VALUES(10004,10004,10004,100,90000,'20-DEC-2017');
45 1 row created.
46
47 EXEC dbms_stats.gather_table_stats('RCOEM','SALES');
48
49 PL/SQL procedure successfully completed.
50
51 SELECT TABLESPACE_NAME,PARTITION_NAME,NUM_ROWS FROM USER_TAB_PARTITIONS WHERE
52 TABLE_NAME='SALES';
53
54 TABLESPACE_NAME          PARTITION_NAME          NUM_ROWS
55 -----
56 ABC1                      SALES_Q1                1
57 ABC2                      SALES_Q2                1
58 ABC3                      SALES_Q3                1
59 ABC4                      SALES_Q4                1
60
61 QUERY 2:
62     Create the same table as in Q1. With a different name with ENABLE ROW MOVEMENT.
63
64 CREATE TABLE SALES_2(
65     PROD_ID NUMBER(5) NOT NULL,
66     CUST_ID NUMBER NOT NULL,
67     PROMO_ID NUMBER(5) NOT NULL,
68     QUANTITY_SOLD NUMBER(5) NOT NULL,

```

```

65  AMOUNT_SOLD NUMBER(5) NOT NULL,
66  TIME_ID DATE NOT NULL)
67  PARTITION BY RANGE(TIME_ID)
68  (PARTITION SALES_Q1 VALUES LESS THAN('01-APR-2017') TABLESPACE ABC1,
69  PARTITION SALES_Q2 VALUES LESS THAN('01-JUL-2017') TABLESPACE ABC2,
70  PARTITION SALES_Q3 VALUES LESS THAN('01-OCT-2017') TABLESPACE ABC3,
71  PARTITION SALES_Q4 VALUES LESS THAN('01-JAN-2018') TABLESPACE ABC4
72  )
73  ENABLE ROW MOVEMENT;
74
75  Table created.
76
77  INSERT INTO SALES_2 VALUES(10001,10001,10001,25,30000,'20-JAN-2017');
78  1 row created.
79  INSERT INTO SALES_2 VALUES(10002,10002,10002,55,40000,'20-MAY-2017');
80  1 row created.
81  INSERT INTO SALES_2 VALUES(10003,10003,10003,50,80000,'20-JUL-2017');
82  1 row created.
83  INSERT INTO SALES_2 VALUES(10004,10004,10004,100,90000,'20-DEC-2017');
84  1 row created.
85
86  UPDATE SALES_2 SET TIME_ID='01-FEB-2017' WHERE PROD_ID=10003;
87  1 row updated.
88
89  EXEC dbms_stats.gather_table_stats('RCOEM','SALES_2');
90  PL/SQL procedure successfully completed.
91  SELECT TABLESPACE_NAME,PARTITION_NAME,NUM_ROWS FROM USER_TAB_PARTITIONS WHERE
92  TABLE_NAME='SALES_2';

```

TABLESPACE_NAME	PARTITION_NAME	NUM_ROWS
ABC1	SALES_Q1	2
ABC2	SALES_Q2	1
ABC3	SALES_Q3	0
ABC4	SALES_Q4	1

```

100
101  QUERY 3:
102      Create a table with list partition as follows:
103  Table having columns deptno, deptname, quarterly_sales and state.
104  Create partition on state:
105  •Northwest on OR and WA
106  •Southwest on AZ, UT and NM
107  •northeast on NY, VM and NJ
108  •southeast on FL and GA
109  •northcentral on SD and WI
110  •southcentral on OK and TX
111  Add the following entries into the table and make conclusion to which partition the
112  entry maps:
113  •(10, 'accounting', 100, 'WA')
114  •(20, 'R&D', 150, 'OR')
115  •(30, 'sales', 100, 'FL')
116  •(40, 'HR', 10, 'TX')
117  •(50, 'systems engineering', 10, 'CA')
118
119  CREATE TABLE DEPT(
120  DEPTNO NUMBER(3) NOT NULL,
121  DEPTNAME VARCHAR2(20) NOT NULL,
122  QUATERLY_SALES NUMBER(5) NOT NULL,
123  STATE VARCHAR2(2) NOT NULL)
124  PARTITION BY LIST(STATE)
125  (
126  PARTITION NORTHWEST VALUES('OR','WA'),
127  PARTITION SOUTHWEST VALUES('AZ','UT','NM'),
128  PARTITION NORTHEAST VALUES('NY','VM','NJ'),
129  PARTITION SOUTHEAST VALUES('FL','GA'),
130  PARTITION NORTHCENTRAL VALUES('SD','WI'),
131  PARTITION SOUTHCENTRAL VALUES('OK','TX')

```

```

132 );
133
134 Table created.
135
136 INSERT INTO DEPT VALUES(10,'ACCOUNTING',100,'WA');
137 1 row created.
138 INSERT INTO DEPT VALUES(20,'R&D',150,'OR');
139 1 row created.
140 INSERT INTO DEPT VALUES(30,'SALES',100,'FL');
141 1 row created.
142 INSERT INTO DEPT VALUES(40,'HR',10,'TX');
143 1 row created.
144 INSERT INTO DEPT VALUES(50,'SYSTEM ENGINEERING',10,'CA');
145 INSERT INTO DEPT VALUES(50,'SYSTEM ENGINEERING',10,'CA')
146 *
147 ERROR at line 1:
148 ORA-14400: inserted partition key does not map to any partition
149
150 ALTER TABLE DEPT ADD PARTITION DEF_STATE VALUES(DEFAULT);
151 Table altered.
152
153 INSERT INTO DEPT VALUES(50,'SYSTEM ENGINEERING',10,'CA');
154 1 row created.
155
156 SELECT * FROM DEPT;
157
158      DEPTNO DEPTNAME      QUARTERLY_SALES ST
159 -----
160          10 ACCOUNTING          100 WA
161          20 R              150 OR
162          30 SALES          100 FL
163          40 HR              10 TX
164          50 SYSTEM ENGINEERING  10 CA
165
166
167 EXEC dbms_stats.gather_table_stats('RCOEM','DEPT');
168 PL/SQL procedure successfully completed.
169
170 SELECT TABLESPACE_NAME,PARTITION_NAME,NUM_ROWS FROM USER_TAB_PARTITIONS WHERE
TABLE_NAME='DEPT';
171 TABLESPACE_NAME      PARTITION_NAME      NUM_ROWS
172 -----
173 USERS                NORTHWEST                2
174 USERS                SOUTHWEST                0
175 USERS                NORTHEAST                0
176 USERS                SOUTHEAST                1
177 USERS                NORTHCENTRAL            0
178 USERS                SOUTHCENTRAL            1
179 USERS                DEF_STATE                1
180
181 7 rows selected.
182
183 QUERY 4:
184 Create a table with hash partition as follows:
185 •Create table Emp with attributes empno, job, sal, deptno and perform hash
partitioning on empno. Number of Partitions should be 5. Demonstrate using system
defined and user defined partition concepts.
186
187
188 CREATE TABLE EMP(
189 EMPNO NUMBER(5) NOT NULL,
190 JOB VARCHAR2(10) NOT NULL,
191 SAL NUMBER(5) NOT NULL,
192 DEPTNO NUMBER(5) NOT NULL)
193 PARTITION BY HASH(EMPNO)
194 PARTITIONS 5;
195 Table created.
196
197 INSERT INTO EMP VALUES(110,'QA',35000,3);

```

```

198 1 row created.
199 INSERT INTO EMP VALUES (210, 'DEV', 35000, 4);
200 1 row created.
201 INSERT INTO EMP VALUES (250, 'QA', 45000, 3);
202 1 row created.
203 INSERT INTO EMP VALUES (310, 'CEO', 90000, 1);
204 1 row created.
205
206 EXEC dbms_stats.gather_table_stats('RCOEM', 'EMP');
207 PL/SQL procedure successfully completed.
208
209 SELECT TABLESPACE_NAME, PARTITION_NAME, NUM_ROWS FROM USER_TAB_PARTITIONS WHERE
    TABLE_NAME='EMP';
210
211 TABLESPACE_NAME          PARTITION_NAME          NUM_ROWS
212 -----
213 USERS                     SYS_P26                  1
214 USERS                     SYS_P27                  2
215 USERS                     SYS_P28                  1
216 USERS                     SYS_P29                  0
217 USERS                     SYS_P30                  0
218
219
220 CREATE TABLE EMP_2 (
221 EMPNO NUMBER(5) NOT NULL,
222 JOB VARCHAR2(10) NOT NULL,
223 SAL NUMBER(5) NOT NULL,
224 DEPTNO NUMBER(5) NOT NULL)
225 PARTITION BY HASH(EMPNO)
226 (PARTITION H1,
227 PARTITION H2,
228 PARTITION H3,
229 PARTITION H4,
230 PARTITION H5);
231 Table created.
232
233 INSERT INTO EMP_2 VALUES (110, 'QA', 35000, 3);
234 1 row created.
235 INSERT INTO EMP_2 VALUES (210, 'DEV', 35000, 4);
236 1 row created.
237 INSERT INTO EMP_2 VALUES (250, 'QA', 45000, 3);
238 1 row created.
239 INSERT INTO EMP_2 VALUES (310, 'CEO', 90000, 1);
240 1 row created.
241
242 EXEC dbms_stats.gather_table_stats('RCOEM', 'EMP_2');
243 PL/SQL procedure successfully completed.
244
245 SELECT TABLESPACE_NAME, PARTITION_NAME, NUM_ROWS FROM USER_TAB_PARTITIONS WHERE
    TABLE_NAME='EMP_2';
246
247 TABLESPACE_NAME          PARTITION_NAME          NUM_ROWS
248 -----
249 USERS                     H1                      1
250 USERS                     H2                      2
251 USERS                     H3                      1
252 USERS                     H4                      0
253 USERS                     H5                      0
254
255 QUERY 5:
256 Create a multi-column range partitioned table as directed:
257 Create a table with the actual DATE information in three separate columns: year, month,
    and day. Also amount_sold.
258 Create following partitions:
259 oBefore 2001: Less than jan 2001
260 oLess than april 2001
261 oLess than july 2001
262 oLess than oct 2001
263 oLess than jan 2002

```

```

264 oFuture with max incoming value
265 Insert values into table and show to which partition does the value belong.
266 o(2001,3,17, 2000);
267 o(2001,11,1, 5000);
268 o(2002,1,1, 4000);
269 Make conclusion for each result.
270
271
272
273 QUERY 6 :
274     Create a multicolumn partitioned table as directed:
275 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.
276 Insert the following values
277 -(5,5, 1000);
278 -(5,150, 1000);
279 -(10,100, 1000);
280
281 CREATE TABLE SUPPLIER_PARTS
282 (
283 SUPPLIER_ID NUMBER(5) ,
284 PARTNUM NUMBER(5) ,
285 PRICE NUMBER(5) )
286 PARTITION BY RANGE(SUPPLIER_ID,PARTNUM) (
287 PARTITION P1 VALUES LESS THAN(5,100),
288 PARTITION P2 VALUES LESS THAN(5,150),
289 PARTITION P3 VALUES LESS THAN(5,200),
290 PARTITION P4 VALUES LESS THAN(10,100),
291 PARTITION P5 VALUES LESS THAN(10,150),
292 PARTITION P6 VALUES LESS THAN(MAXVALUE,MAXVALUE)
293 );
294 Table created.
295 INSERT INTO SUPPLIER_PARTS VALUES(5,5,1000);
296 1 row created.
297 INSERT INTO SUPPLIER_PARTS VALUES(5,150,1000);
298 1 row created.
299 INSERT INTO SUPPLIER_PARTS VALUES(10,100,1000);
300 1 row created.
301 EXEC dbms_stats.gather_table_stats('MANJARI','SUPPLIER_PARTS');
302 PL/SQL procedure successfully completed.
303
304 SELECT TABLESPACE_NAME,PARTITION_NAME,NUM_ROWS FROM USER_TAB_PARTITIONS WHERE
TABLE_NAME='SUPPLIER_PARTS';
305
306 TABLESPACE_NAME          PARTITION_NAME          NUM_ROWS
307 -----
308 USERS                     P1                       2
309 USERS                     P2                       0
310 USERS                     P3                       2
311 USERS                     P4                       0
312 USERS                     P5                       2
313 USERS                     P6                       0
314
315 6 rows selected.
316
317
318 QUERY 7 :
319     Create interval partitioned table as directed:
320 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, ...)
321 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
322 Perform interval partitioning on month and take interval of 01 months.
323
324 CREATE TABLE SALES(

```

```

325     PROD_ID NUMBER(3) NOT NULL,
326     CUST_ID NUMBER(3) NOT NULL,
327     PROMO_ID NUMBER(3) NOT NULL,
328     QUANTITY_SOLD NUMBER(3) NOT NULL,
329     AMOUNT_SOLD NUMBER(3) NOT NULL,
330     MONTH NUMBER(2) NOT NULL)
331     PARTITION BY RANGE(MONTH)
332     INTERVAL(1) (
333     PARTITION SALES_Q1 VALUES LESS THAN(4),
334     PARTITION SALES_Q2 VALUES LESS THAN(7),
335     PARTITION SALES_Q3 VALUES LESS THAN(10)
336 );
337 Table created.
338     INSERT INTO SALES VALUES(101,101,101,3,300,3);
339 1 row created.
340     INSERT INTO SALES VALUES(102,102,102,4,400,4);
341 1 row created.
342     INSERT INTO SALES VALUES(103,103,103,7,700,6);
343 1 row created.
344     INSERT INTO SALES VALUES(104,104,104,1,100,11);
345 1 row created.
346
347 EXEC dbms_stats.gather_table_stats('MANJARI','SALES');
348 PL/SQL procedure successfully completed.
349 SELECT TABLESPACE_NAME,PARTITION_NAME,NUM_ROWS FROM USER_TAB_PARTITIONS WHERE
TABLE_NAME='SALES';
350 TABLESPACE_NAME      PARTITION_NAME      NUM_ROWS
351 -----
352 USERS                  SALES_Q1                  1
353 USERS                  SALES_Q2                  2
354 USERS                  SALES_Q3                  0
355 USERS                  SYS_P41                   1
356
357
358 QUERY 8 :
359     Demonstrate reference partitioning as directed:
360 Create parent table Orders with the attributes order_id, order_date, customer_id,
shipper_id.
361 Perform Range partitioning on Order Date. Take Range of 03 Months i.e. 01 quarter
362 Create child table order_items with attributes order_id, product_id, price and quantity.
363 Perform Reference partitioning on child table.
364 Delete the created partitions.
365
366 CREATE TABLE ORDERS
367 ( ORDER_ID NUMBER(3) PRIMARY KEY,
368 ORDER_DATE DATE NOT NULL,
369 CUSTOMER_ID NUMBER(3) NOT NULL,
370 SHIPPER_ID NUMBER(3) NOT NULL)
371 PARTITION BY RANGE(ORDER_DATE)
372 (PARTITION ORDERS_Q1 VALUES LESS THAN('01-APR-2017'),
373 PARTITION ORDERS_Q2 VALUES LESS THAN('01-JUL-2017') ,
374 PARTITION ORDERS_Q3 VALUES LESS THAN('01-OCT-2017'),
375 PARTITION ORDERS_Q4 VALUES LESS THAN('01-JAN-2018')
376 );
377 Table created.
378
379 CREATE TABLE ORDER_ITEMS
380 ( ORDER_ID NUMBER(3) NOT NULL,
381 PRODUCT_ID NUMBER(3) NOT NULL,
382 PRICE NUMBER(5) NOT NULL,
383 QUANTITY NUMBER(3) NOT NULL,
384 CONSTRAINT ORDERS_ORDER_ID_FK FOREIGN KEY(ORDER_ID) REFERENCES ORDERS)
385 PARTITION BY REFERENCE(ORDERS_ORDER_ID_FK);
386 Table created.
387
388 INSERT INTO ORDERS VALUES(111,'01-FEB-2017',101,101);
389 1 row created.
390 INSERT INTO ORDERS VALUES(222,'01-JUN-2017',102,102);
391 1 row created.

```

```

392 INSERT INTO ORDERS VALUES (333, '01-AUG-2017', 103, 103);
393 1 row created.
394 INSERT INTO ORDERS VALUES (444, '01-NOV-2017', 104, 104);
395 1 row created.
396
397 INSERT INTO ORDER_ITEMS VALUES (111, 202, 10000, 3);
398 1 row created.
399 INSERT INTO ORDER_ITEMS VALUES (222, 204, 20000, 6);
400 1 row created.
401 INSERT INTO ORDER_ITEMS VALUES (333, 206, 15000, 2);
402 1 row created.
403 INSERT INTO ORDER_ITEMS VALUES (444, 208, 45000, 1);
404 1 row created.
405
406 SELECT * FROM ORDERS PARTITION(ORDERS_Q1);
407
408 ORDER_ID ORDER_DAT CUSTOMER_ID SHIPPER_ID
409 -----
410          111 01-FEB-17          101          101
411
412 SELECT * FROM ORDER_ITEMS PARTITION(ORDERS_Q1);
413
414 ORDER_ID PRODUCT_ID PRICE QUANTITY
415 -----
416          111          202          10000          3
417
418 ALTER TABLE ORDERS DROP PARTITION(ORDERS_Q1);
419 Table altered.
420
421 QUERY 9:
422 Implement virtual column based partitioning as below:
423 Create table employee with attributes Emp_id, emp_name, fixed_salary, variable_salary.
424 Generate Total salary as virtual colum.
425 Perform range partitioning on Total Salary with four partitions as below:
426 Partition P1 stores salary less than 25000
427 Partition P2 stores salary less than 50000
428 Partition P3 stores salary less than 75000
429 Partition P4 stores any salary above and equal to than 75000
430
431 CREATE TABLE EMPLOYEE(
432 EMP_ID NUMBER(3) NOT NULL,
433 EMP_NAME VARCHAR2(20) NOT NULL,
434 VARIABLE_SALARY NUMBER(5) NOT NULL,
435 FIXED_SALARY NUMBER(5) NOT NULL,
436 TOTAL NUMBER(7)
437 GENERATED ALWAYS AS
438 ( FIXED_SALARY+VARIABLE_SALARY) VIRTUAL
439 ) PARTITION BY RANGE(TOTAL)(
440 PARTITION P1 VALUES LESS THAN (25000),
441 PARTITION P2 VALUES LESS THAN (50000),
442 PARTITION P3 VALUES LESS THAN (75000),
443 PARTITION P4 VALUES LESS THAN (MAXVALUE)
444 );
445
446 INSERT INTO EMPLOYEE(EMP_ID,EMP_NAME,VARIABLE_SALARY,FIXED_SALARY) VALUES(111,'Jiawei
447 Han',30000,25000);
448
449 INSERT INTO EMPLOYEE(EMP_ID,EMP_NAME,VARIABLE_SALARY,FIXED_SALARY) VALUES(222,'Will
450 Smith',40000,10000);
451
452 INSERT INTO EMPLOYEE(EMP_ID,EMP_NAME,VARIABLE_SALARY,FIXED_SALARY) VALUES(333,'Jiawei
453 Han',60000,45000);
454
455 SELECT * FROM EMPLOYEE;
456
457 EMP_ID EMP_NAME VARIABLE_SALARY FIXED_SALARY TOTAL
458 -----
459          111 Jiawei Han          30000          25000          55000
460          222 Will Smith          40000          10000          50000
461          333 Jiawei Han          60000          45000          105000

```

```

457 QUERY 10 :
458     Demonstrate Composite partitioning technique as directed
459 Implement range list partitioning for customer table having attributes cust_id,
cust_name, cust_state, and time_id
460 oPerform range partitioning on time-id and list partitioning on state attributes. Also
create maxvalue and default partition for range and list partition respectively.
461 oPartition definitions for range are as below:
462 Partition old should accept values less than 01-Jan-2005
463 Partition acquired should accept values less than 01-Jan-2010
464 Partition recent should accept values less than 01-Jan-2015
465 Partition unknown should accept values greater than 01-Jan-2015
466 Partition definitions for list are as below:
467 Partition west should accept values ('MH', 'GJ')
468 Partition south should accept values ('TN', 'AP')
469 Partition north should accept values ('UP', 'HP')
470 Partition unknown should accept any other state.
471
472 CREATE TABLE CUSTOMER_RANGE_LIST(
473 CUST_ID NUMBER(3) NOT NULL,
474 CUST_NAME VARCHAR2(20) NOT NULL,
475 CUST_STATE VARCHAR2(20) NOT NULL,
476 TIME_ID DATE NOT NULL)
477 PARTITION BY RANGE(TIME_ID)
478 SUBPARTITION BY LIST(CUST_STATE)
479 SUBPARTITION TEMPLATE(
480 SUBPARTITION WEST VALUES('MH','GJ'),
481 SUBPARTITION SOUTH VALUES('TN','AP'),
482 SUBPARTITION NORTH VALUES('UP','HP'),
483 SUBPARTITION UNKNOWN VALUES(DEFAULT))
484 (PARTITION OLD VALUES LESS THAN('01-JAN-2005'),
485 PARTITION ACQUIRED VALUES LESS THAN('01-JAN-2010'),
486 PARTITION RECENT VALUES LESS THAN('01-JAN-2015'),
487 PARTITION UNKOWN VALUES LESS THAN(MAXVALUE)
488 );
489 Table created.
490
491 INSERT INTO CUSTOMER_RANGE_LIST VALUES(111,'WILL SMITH','MH','01-AUG-2005');
492 1 row created.
493 INSERT INTO CUSTOMER_RANGE_LIST VALUES(222,'SRK','TN','01-AUG-2010');
494 1 row created.
495 INSERT INTO CUSTOMER_RANGE_LIST VALUES(333,'SALMAN','HP','01-AUG-2015');
496 1 row created.
497 INSERT INTO CUSTOMER_RANGE_LIST VALUES(444,'AAMIR','MP','01-AUG-2018');
498 1 row created.
499
500 EXEC dbms_stats.gather_table_stats('MANJARI','CUSTOMER_RANGE_LIST');
501 PL/SQL procedure successfully completed.
502
503 SELECT PARTITION_NAME,SUBPARTITION_NAME,NUM_ROWS
504 FROM USER_TAB_SUBPARTITIONS WHERE TABLE_NAME='CUSTOMER_RANGE_LIST';
505
506 PARTITION_NAME          SUBPARTITION_NAME          NUM_ROWS
507 -----
508 OLD                      OLD_WEST                    0
509 OLD                      OLD_SOUTH                   0
510 OLD                      OLD_NORTH                   0
511 OLD                      OLD_UNKNOWN                 0
512 ACQUIRED                 ACQUIRED_WEST               1
513 ACQUIRED                 ACQUIRED_SOUTH              0
514 ACQUIRED                 ACQUIRED_NORTH              0
515 ACQUIRED                 ACQUIRED_UNKNOWN            0
516 RECENT                  RECENT_WEST                 0
517 RECENT                  RECENT_SOUTH                1
518 RECENT                  RECENT_NORTH                0
519
520 PARTITION_NAME          SUBPARTITION_NAME          NUM_ROWS
521 -----
522 RECENT                  RECENT_UNKNOWN              0
523 UNKOWN                  UNKOWN_WEST                 0

```



```

524 UNKNOWN UNKNOWN_SOUTH 0
525 UNKNOWN UNKNOWN_NORTH 1
526 UNKNOWN UNKNOWN_UNKNOWN 1
527
528 16 rows selected.
529
530 SELECT * FROM CUSTOMER_RANGE_LIST SUBPARTITION(ACQUIRED_WEST);
531 CUST_ID CUST_NAME CUST_STATE TIME_ID
532 -----
533 111 WILL SMITH MH 01-AUG-05
534
535
536 .....prac3_+5 problem
statement.....
537 11) RANGE-HASH
538 SQL> CREATE TABLE composite_rng_hash(
539 2 cust_id NUMBER(10),
540 3 cust_name VARCHAR2(25),
541 4 cust_state VARCHAR2(2),
542 5 amt_sold VARCHAR2(2),
543 6 time_id DATE)
544 7 PARTITION BY RANGE(time_id)
545 8 SUBPARTITION BY HASH(cust_id)
546 9 SUBPARTITION TEMPLATE(
547 10 SUBPARTITION h1,
548 11 SUBPARTITION h2,
549 12 SUBPARTITION h3)
550 13 (PARTITION YEAR_2006 VALUES LESS THAN (TO_DATE('01-APR-2006','DD-MON-YYYY'))),
551 14 PARTITION YEAR_2007 VALUES LESS THAN(TO_DATE('01-APR-2007','DD-MON-YYYY'))),
552 15 PARTITION YEAR_2008 VALUES LESS THAN(TO_DATE('01-APR-2008','DD-MON-YYYY'))
553 16 );
554
555 Table created.
556
557 SQL> DESC composite_rng_hash;
558 Name Null? Type
559 -----
560 CUST_ID NUMBER(10)
561 CUST_NAME VARCHAR2(25)
562 CUST_STATE VARCHAR2(2)
563 AMT_SOLD VARCHAR2(2)
564 TIME_ID DATE
565
566
567 SQL> insert into composite_rng_hash values(11,'cse','lp',21,'11-feb-2008');
568
569 1 row created.
570
571 SQL> SELECT partition_name, subpartition_name, num_rows
572 2 FROM user_tab_subpartitions where table_name='COMPOSITE_RNG_HASH';
573
574 PARTITION_NAME SUBPARTITION_NAME NUM_ROWS
575 -----
576 YEAR_2006 YEAR_2006_H1
577 YEAR_2006 YEAR_2006_H2
578 YEAR_2006 YEAR_2006_H3
579 YEAR_2007 YEAR_2007_H1
580 YEAR_2007 YEAR_2007_H2
581 YEAR_2007 YEAR_2007_H3
582 YEAR_2008 YEAR_2008_H1
583 YEAR_2008 YEAR_2008_H2
584 YEAR_2008 YEAR_2008_H3
585
586 9 rows selected.
587
588 select * from composite_rng_hash subpartition(YEAR_2008_h1);
589 CUST_ID CUST_NAME CU AM TIME_ID
590 -----
591 11 cse lp 21 11-FEB-08

```

```

592
593 SQL> select * from composite_rng_hash subpartition(YEAR_2008_h2);
594
595 no rows selected
596
597 SQL> select * from composite_rng_hash subpartition(YEAR_2008_h3);
598
599 no rows selected
600
601
602 12) RANGE-RANGE
603 CREATE TABLE composite_rng_rng (
604     cust_id NUMBER(10),
605     cust_name VARCHAR2(25),
606     cust_state VARCHAR2(2),
607     amt_sold VARCHAR2(2),
608     time_id DATE)
609 PARTITION BY RANGE(time_id)
610 SUBPARTITION BY RANGE (cust_id)
611 SUBPARTITION TEMPLATE(
612     SUBPARTITION original VALUES LESS THAN (1001),
613     SUBPARTITION acquired VALUES LESS THAN (8001),
614     SUBPARTITION recent VALUES LESS THAN (MAXVALUE))
615 (PARTITION YEAR_2006 VALUES LESS THAN (TO_DATE('01-APR-2006','DD-MON-YYYY')),
616     PARTITION YEAR_2007 VALUES LESS THAN (TO_DATE('01-APR-2007','DD-MON-YYYY')),
617     PARTITION YEAR_2008 VALUES LESS THAN (TO_DATE('01-APR-2008','DD-MON-YYYY'))
618 );
619 SQL> desc composite_rng_rng;
620 Name Null? Type
621 -----
622 CUST_ID NUMBER(10)
623 CUST_NAME VARCHAR2(25)
624 CUST_STATE VARCHAR2(2)
625 AMT_SOLD VARCHAR2(2)
626 TIME_ID DATE
627
628 SQL> insert into composite_rng_rng values(11,'cse','OR',21,'11-feb-2007');
629
630 1 row created.
631
632 SQL> insert into composite_rng_rng values(11,'cse','OR',21,'11-feb-2008');
633
634 1 row created.
635
636 SQL> SELECT partition_name, subpartition_name, num_rows
637 2 FROM user_tab_subpartitions where table_name='COMPOSITE_RNG_RNG';
638
639 PARTITION_NAME SUBPARTITION_NAME NUM_ROWS
640 -----
641 YEAR_2006 YEAR_2006_ORIGINAL
642 YEAR_2006 YEAR_2006_ACQUIRED
643 YEAR_2006 YEAR_2006_RECENT
644 YEAR_2007 YEAR_2007_ORIGINAL
645 YEAR_2007 YEAR_2007_ACQUIRED
646 YEAR_2007 YEAR_2007_RECENT
647 YEAR_2008 YEAR_2008_ORIGINAL
648 YEAR_2008 YEAR_2008_ACQUIRED
649 YEAR_2008 YEAR_2008_RECENT
650
651 9 rows selected.
652
653 select * from composite_rng_rng subpartition(YEAR_2008_original);
654 CUST_ID CUST_NAME CU AM TIME_ID
655 -----
656 11 cse OR 21 11-FEB-08
657
658 13) LIST-HASH
659 SQL> CREATE TABLE composite_list_hash (
660 2 cust_id NUMBER(10),

```

```

661 3 cust_name VARCHAR2(25),
662 4 cust_state VARCHAR2(2),
663 5 amt_sold VARCHAR2(2),
664 6 time_id DATE)
665 7 PARTITION BY LIST(cust_state)
666 8 SUBPARTITION BY HASH (cust_id)
667 9 SUBPARTITION TEMPLATE(
668 10 SUBPARTITION h1,
669 11 SUBPARTITION h2,
670 12 SUBPARTITION h3)
671 13 (PARTITION west VALUES ('OR', 'WA'),
672 14 PARTITION east VALUES ('NY', 'CT'),
673 15 PARTITION cent VALUES ('IL', 'MN')
674 16 );
675
676 Table created.
677
678 SQL> desc composite_list_hash;
679 Name Null? Type
680 -----
681 CUST_ID NUMBER(10)
682 CUST_NAME VARCHAR2(25)
683 CUST_STATE VARCHAR2(2)
684 AMT_SOLD VARCHAR2(2)
685 TIME_ID DATE
686
687 SQL> SELECT partition_name, subpartition_name, num_rows
688 2 FROM user_tab_subpartitions where table_name='COMPOSITE_LIST_HASH';
689
690 PARTITION_NAME SUBPARTITION_NAME NUM_ROWS
691 -----
692 WEST WEST_H1
693 WEST WEST_H2
694 WEST WEST_H3
695 EAST EAST_H1
696 EAST EAST_H2
697 EAST EAST_H3
698 CENT CENT_H1
699 CENT CENT_H2
700 CENT CENT_H3
701
702 9 rows selected.
703 SQL> insert into composite_list_hash values(2,'MEC','NY',22,'10-feb-2018');
704
705 1 row created.
706 SQL> insert into composite_list_hash values(2,'CSE','IL',22,'10-JAN-2018');
707
708 1 row created.
709
710 select * from composite_list_hash subpartition(west_h1);
711
712 SQL> select * from composite_list_hash subpartition(east_h1);
713
714 no rows selected
715
716 SQL> select * from composite_list_hash subpartition(east_h2);
717
718 no rows selected
719
720 SQL> select * from composite_list_hash subpartition(east_h3);
721
722 CUST_ID CUST_NAME CU AM TIME_ID
723 -----
724 2 MEC NY 22 10-FEB-18
725
726 14)LIST-LIST
727 SQL> CREATE TABLE composite_list_list(
728 2 cust_id NUMBER(10),
729 3 cust_name VARCHAR2(25),

```

```

730 4 cust_state VARCHAR2(2),
731 5 amt_sold VARCHAR2(2),
732 6 time_id DATE)
733 7 PARTITION BY LIST(cust_state)
734 8 SUBPARTITION BY LIST(cust_id)
735 9 SUBPARTITION TEMPLATE
736 10 (SUBPARTITION original VALUES(1001),
737 11 SUBPARTITION acquired VALUES (8001),
738 12 SUBPARTITION recent VALUES (default))
739 13 (PARTITION west VALUES ('OR', 'WA'),
740 14 PARTITION east VALUES ('NY', 'CT'),
741 15 PARTITION cent VALUES ('IL', 'MN')
742 16 );
743
744 Table created.
745
746 SQL> desc composite_list_list;
747 Name Null? Type
748 -----
749 CUST_ID NUMBER(10)
750 CUST_NAME VARCHAR2(25)
751 CUST_STATE VARCHAR2(2)
752 AMT_SOLD VARCHAR2(2)
753 TIME_ID DATE
754 SQL> SELECT partition_name, subpartition_name, num_rows
755 2 FROM user_tab_subpartitions where table_name='COMPOSITE_LIST_LIST';
756
757 PARTITION_NAME SUBPARTITION_NAME NUM_ROWS
758 -----
759 WEST WEST_ORIGINAL
760 WEST WEST_ACQUIRED
761 WEST WEST_RECENT
762 EAST EAST_ORIGINAL
763 EAST EAST_ACQUIRED
764 EAST EAST_RECENT
765 CENT CENT_ORIGINAL
766 CENT CENT_ACQUIRED
767 CENT CENT_RECENT
768
769 9 rows selected.
770 SQL> insert into composite_list_list values (21,'IND','IL',22,'10-feb-2019');
771
772 1 row created.
773
774 SQL> insert into composite_list_list values(21,'IND','IL',32,'10-APR-2020');
775
776 1 row created.
777
778 SQL> select * from composite_list_list subpartition(cent_recent);
779
780 CUST_ID CUST_NAME CU AM TIME_ID
781 -----
782 21 IND IL 22 10-FEB-19
783 21 IND IL 32 10-APR-20
784
785 15)LIST-RANGE
786 SQL> CREATE TABLE composite_list_rng (
787 2 cust_id NUMBER(10),
788 3 cust_name VARCHAR2(25),
789 4 cust_state VARCHAR2(2),
790 5 amt_sold VARCHAR2(2),
791 6 time_id DATE)
792 7 PARTITION BY LIST(cust_state)
793 8 SUBPARTITION BY RANGE (cust_id)
794 9 SUBPARTITION TEMPLATE(
795 10 SUBPARTITION original VALUES LESS THAN (1001),
796 11 SUBPARTITION acquired VALUES LESS THAN (8001),
797 12 SUBPARTITION recent VALUES LESS THAN (MAXVALUE))
798 13 (PARTITION west VALUES ('OR', 'WA'),

```

```

799      14 PARTITION east VALUES ('NY', 'CT'),
800      15 PARTITION cent VALUES ('IL', 'MN')
801      16 );
802
803      Table created.
804
805      SQL> desc composite_list_rng;
806      Name Null? Type
807      -----
808      CUST_ID NUMBER(10)
809      CUST_NAME VARCHAR2(25)
810      CUST_STATE VARCHAR2(2)
811      AMT_SOLD VARCHAR2(2)
812      TIME_ID DATE
813
814      SQL> SELECT partition_name, subpartition_name, num_rows
815      FROM user_tab_subpartitions where table_name='COMPOSITE_LIST_RNG';
816
817      PARTITION_NAME SUBPARTITION_NAME NUM_ROWS
818      -----
819      CENT CENT_ORIGINAL
820      CENT CENT_ACQUIRED
821      CENT CENT_RECENT
822      EAST EAST_ORIGINAL
823      EAST EAST_ACQUIRED
824      EAST EAST_RECENT
825      WEST WEST_ORIGINAL
826      WEST WEST_ACQUIRED
827      WEST WEST_RECENT
828
829      9 rows selected.
830      SQL> insert into composite_list_rng values(1,'cse','OR',2,'10-feb-2018');
831      1 row created.
832
833      SQL> insert into composite_list_rng values(2,'MEC','OR',22,'10-feb-2018');
834      1 row created.
835
836      SQL> select * from composite_list_rng partition(west);
837      CUST_ID CUST_NAME CU AM TIME_ID
838      -----
839      1 cse OR 2 10-FEB-18
840      2 MEC OR 22 10-FEB-18
841
842      SQL> select * from composite_list_rng subpartition(west_original);
843
844      CUST_ID CUST_NAME CU AM TIME_ID
845      -----
846      1 cse OR 2 10-FEB-18
847      2 MEC OR 22 10-FEB-18
848

```