|  |
| --- |
|  |
| Partitioning |

Contents

[1. Hands-on task 3](#_Toc412572569)

[1.1. Adding partition 3](#_Toc412572570)

[1.2. Coalesce partition 3](#_Toc412572571)

[1.3. Dropping partition 3](#_Toc412572572)

[1.4. Merging partition 3](#_Toc412572572)

[1.5. Moving partition 3](#_Toc412572572)

[1.6. Splitting partition 3](#_Toc412572572)

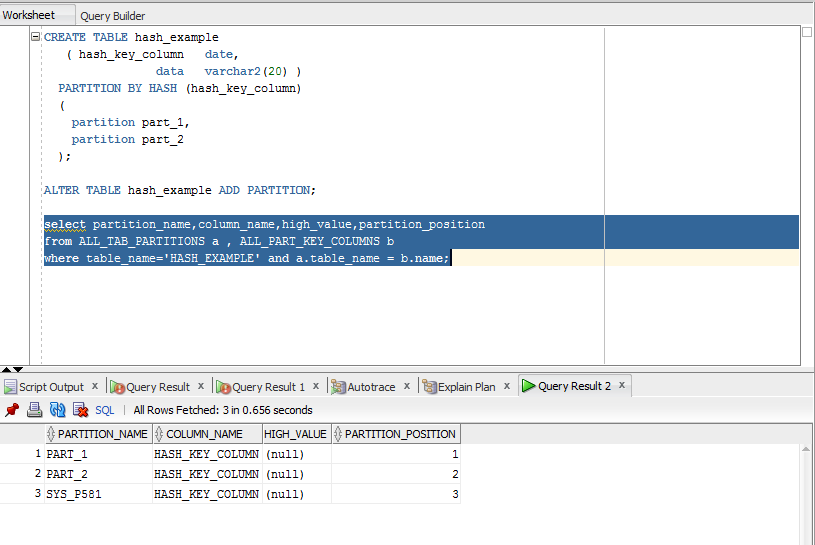
[1.7. Truncating partition 3](#_Toc412572572)

[2. analytical task 3](#_Toc412572573)

# Hands\_On Task

* 1. Adding partition(hash partitioning)

While adding a partition to a hash-partitioned table, the database populates the new partition with rows rehashed from an existing partition (selected by the database) as determined by the hash function. Using add partition adds a new hash partition whose partition name is system generated(SYS\_P581)



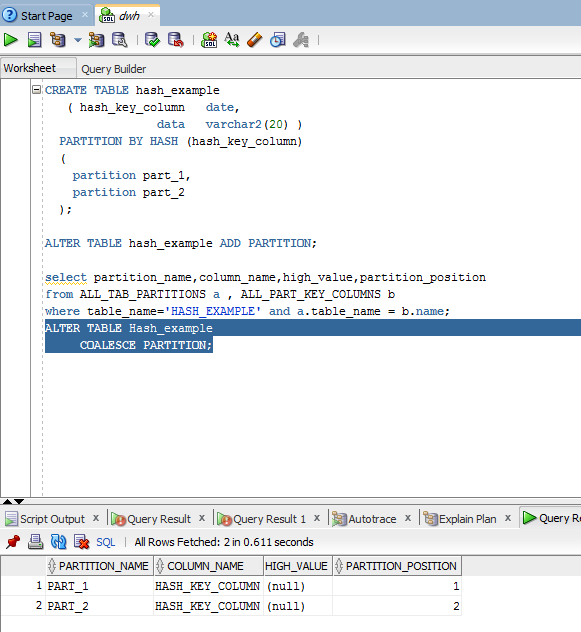
* 1. Coalesce partition(Used only for hash partitioning)

The ALTER TABLE ... COALESCE PARTITION statement is used to coalesce a partition in a hash-partitioned table. The following statement reduces by one the number of partitions in a table by coalescing a partition:

ALTER TABLE Hash\_example

COALESCE PARTITION;

In this example oracle union partitions(part\_2 and SYS\_P581 )



* 1. Drop partition

Created table with range partition because it is impossible to drop hash partition.

CREATE TABLE history\_part\_r (

history\_id NUMBER(10),

person\_id NUMBER(10) NOT NULL,

country\_id NUMBER(10) NOT NULL,

event VARCHAR2(50),

event\_date date,

prof\_hist\_comments VARCHAR2(2000))

PARTITION BY RANGE (event\_date)

(

PARTITION part\_1 VALUES LESS THAN(to\_date('01-JAN-1000','dd-MON-yyyy')),

PARTITION part\_2 VALUES LESS THAN(to\_date('01-JAN-2000','dd-MON-yyyy')),

PARTITION part\_3 VALUES LESS THAN(to\_date('01-JAN-3000','dd-MON-yyyy'))

);

INSERT INTO history\_part\_r

SELECT

TRUNC(dbms\_random.value(1,100),0),

TRUNC(dbms\_random.value(1,100),0),

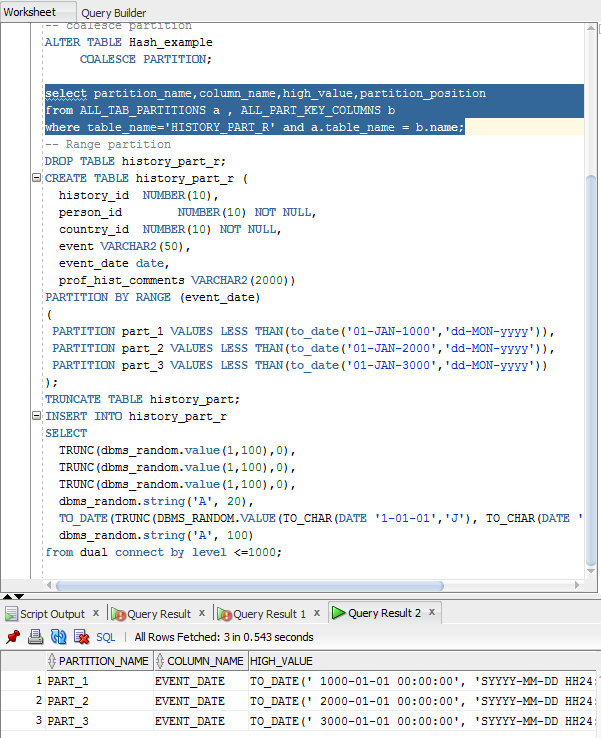
TRUNC(dbms\_random.value(1,100),0),

dbms\_random.string('A', 20),

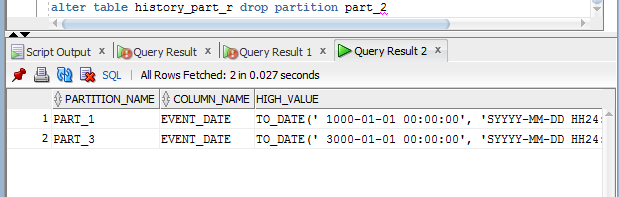
TO\_DATE(TRUNC(DBMS\_RANDOM.VALUE(TO\_CHAR(DATE '1-01-01','J'), TO\_CHAR(DATE '3000-12-31','J'))),'J'),

dbms\_random.string('A', 100)

from dual connect by level <=1000;



Dropped partition (part\_2)



* 1. Merge Partition

Table was created for this situation with 4 partitions

CREATE TABLE four\_seasons

(

one DATE,

two VARCHAR2(60),

three NUMBER

)

PARTITION BY RANGE ( one )

(

PARTITION quarter\_one

VALUES LESS THAN ( TO\_DATE('01-apr-1998','dd-mon-yyyy'))

TABLESPACE quarter\_one,

PARTITION quarter\_two

VALUES LESS THAN ( TO\_DATE('01-jul-1998','dd-mon-yyyy'))

TABLESPACE quarter\_two,

PARTITION quarter\_three

VALUES LESS THAN ( TO\_DATE('01-oct-1998','dd-mon-yyyy'))

TABLESPACE quarter\_three,

PARTITION quarter\_four

VALUES LESS THAN ( TO\_DATE('01-jan-1999','dd-mon-yyyy'))

TABLESPACE quarter\_four

);

create tablespace i\_quarter\_one

DATAFILE 'iquarter1.dat'

SIZE 20M

ONLINE;

create tablespace i\_quarter\_two

DATAFILE 'iquarter2.dat'

SIZE 20M

ONLINE;

create tablespace i\_quarter\_three

DATAFILE 'iquarter3.dat'

SIZE 20M

ONLINE;

create tablespace i\_quarter\_four

DATAFILE 'iquarter4.dat'

SIZE 20M

ONLINE;

CREATE INDEX i\_four\_seasons\_l ON four\_seasons ( one,two )

LOCAL (

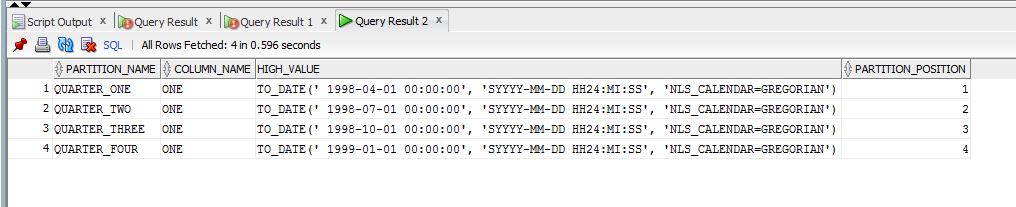
PARTITION i\_quarter\_one TABLESPACE i\_quarter\_one,

PARTITION i\_quarter\_two TABLESPACE i\_quarter\_two,

PARTITION i\_quarter\_three TABLESPACE i\_quarter\_three,

PARTITION i\_quarter\_four TABLESPACE i\_quarter\_four

);

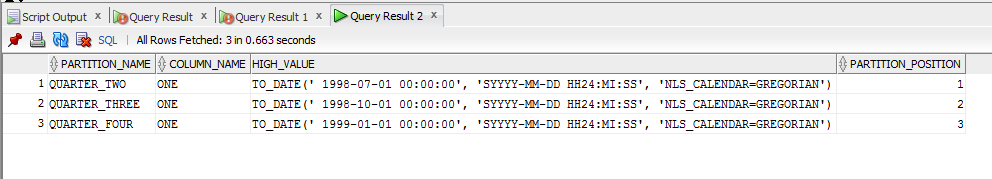


Merge quarter\_one and quarter\_two:

ALTER TABLE four\_seasons

MERGE PARTITIONS quarter\_one, quarter\_two INTO PARTITION quarter\_two

UPDATE INDEXES



Rebuild index for quarter\_two, which has been marked unusable because it has not had all of the data from Q1 added to it.

Rebuilding the index corrects this.

ALTER TABLE four\_seasons MODIFY PARTITION

quarter\_two REBUILD UNUSABLE LOCAL INDEXES;

* 1. Move Partition

Before moving partition quarter\_four it is situated in quarter\_four tablespace



ALTER TABLE four\_seasons MOVE PARTITION Quarter\_four

TABLESPACE tbs\_pdb\_test NOLOGGING COMPRESS;



After that quarter\_four is situated in tbs\_pdb\_test

* 1. Splitting partition

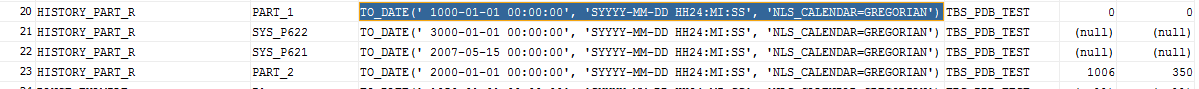
Before splitting partitions:



ALTER TABLE history\_part\_r

SPLIT PARTITION FOR(TO\_DATE('01-MAY-2007','dd-MON-yyyy'))

AT (TO\_DATE('15-MAY-2007','dd-MON-yyyy'));



After splitting partition appear one more partition (between 3000 and 2007)

* 1. Truncating partition

Situation before truncating partition :

ANALYZE TABLE history\_part\_r COMPUTE STATISTICS;

SELECT table\_name, partition\_name, high\_value, tablespace\_name, blocks, num\_rows

FROM user\_tab\_partitions ORDER BY TABLE\_NAME;



ALTER TABLE history\_part\_r TRUNCATE PARTITION part\_1;



As we can see after truncating part\_1 in this partition no existing rows