Practice

Managing Database Storage Structures

Practice Target

In this practice, you will experience managing Oracle database storage structures, specially tablespaces.

Specifically, you will perform the following:

- Examine segments, extents, and data blocks
- Manage tablespaces
- Explore CDB tablespaces and datafiles

Practice Assumptions

You have the `srv1` and its **CDB** database up and running.

Preparing for the Practice

In the following steps, you will create the script files that will be used in this practice.

1. Start a Putty session to `srv1` as `oracle`

2. Run the following code to create a script that inserts data into a testing table.

The script inserts total x rows into `TEST_TABLE`, where x is passed to the script.

```
cat > /home/oracle/insertrows.sql <<EOF
BEGIN
 FOR I IN 1..&1 LOOP
  INSERT INTO TEST_TABLE ( RNAME ) VALUES (DBMS_RANDOM.STRING('U', 10));
 END LOOP;
 COMMIT;
END;
/
EOF
```

3. Run the following code to create a script that displays the segment information for the testing table.

```
cat > /home/oracle/display_segment.sql <<EOF
col SEGMENT_TYPE for a10
col TABLESPACE_NAME for a10

SELECT SEGMENT_TYPE, TABLESPACE_NAME, EXTENTS, BYTES/1024 KB, BLOCKS
FROM USER_SEGMENTS
WHERE SEGMENT_NAME='TEST_TABLE';
EOF
```

4. Run the following code to create a script that displays the datafiles of `HRTBS` tablespace.

```
cat > /home/oracle/display_hrtbs_datafiles.sql <<EOF
SELECT FILE_ID, BYTES/1024/1024 SIZE_MB, INCREMENT_BY*8/1024 INC_BY_MB,
ROUND(MAXBYTES/1024/1024,2) MAX_MB, AUTOEXTENSIBLE
FROM DBA_DATA_FILES
WHERE TABLESPACE_NAME='HRTBS';
EOF
```

Examining Segments, Extents, and Data Blocks

In the following steps, you will create a testing table to study the segments, extents, and data blocks. The target of this practice section is to have a practical understanding of those concepts.

5. Invoke SQL*Plus and login to the database as sys.

```
sqlplus / as sysdba
```

6. Retrieve the standard data block size.

The standard data block size in our environment database is 8 KB. If we want to change this size, we must re-create the database.

```
SHOW PARAMETER BLOCK_SIZE
```

7. Login to PDB1 as HR.

Note: The username you passed to the SQL*Plus is case-insensitive. Whereas, the password is (by default) case sensitive.

```
conn hr/ABcd##1234@//srv1/pdb1.localdomain
```

8. Run the following statement to create a testing table.

```
CREATE TABLE TEST_TABLE (
 ROW_ID NUMBER GENERATED ALWAYS AS IDENTITY,
 RNAME VARCHAR2(10)
);
```

9. Check on which tablespace the table was created.

The table is created in the USERS tablespace because it is the HR default tablespace.

```
SELECT TABLESPACE_NAME FROM USER_TABLES WHERE TABLE_NAME='TEST_TABLE';
```

10. Check if a segment was created for the table.

No segment is created because the table is empty.

```
@ /home/oracle/display_segment.sql
```

11. Insert 10 rows into the testing table.

```
@ /home/oracle/insertrows.sql 10
```

12. Check if a segment was created from the table.

A segment is created. It was created in the users tablespace (because this is the HR's default tablespace). It contains one extent of 8 blocks with a total size of 64 KB.

```
@ /home/oracle/display_segment.sql
```

13. Insert 1000 rows into the testing table.

```
@ /home/oracle/insertrows.sql 10000
```

14. Display the segment information for the testing table.

The segment now contains 4 extents, 32 blocks with a total size of 256 KB.

```
@ /home/oracle/display_segment.sql
```

15. Delete all the rows in the table.

```
DELETE TEST_TABLE;
COMMIT;
```

16. Display the segment information for the testing table.

Although we deleted all the rows, the data blocks are still there (and taking up space from the tablespace).

```
@ /home/oracle/display_segment.sql
```

Managing Tablespaces

In the following steps, you will create a new tablespace and perform some tablespace management tasks on it.

17. Check if the OMF is enabled for the datafiles.

The command retrieves the following error:

```
ORA-00942: table or view does not exist
```

This error is returned because the HR user does not have the privilege to query the system views.

```
show parameter DB_CREATE_FILE_DEST
```

18. Login to PDB1 as SYSTEM

```
conn system/ABcd##1234@//srv1/pdb1.localdomain
```

19. Check if the OMF is enabled.

The OMF is enabled. This means we can create a tablespace without specifying a datafile to it.

```
show parameter DB_CREATE_FILE_DEST
```

20. Run the following code to create a new tablespace.

```
CREATE TABLESPACE HRTBS;
```

21. Check the attributes of the created tablespace.

The EXTENT_MANAGEMENT is LOCAL and the SEGMENT_SPACE_MANAGEMENT is AUTO. Those are the default settings and the recommended settings.

```
SELECT EXTENT_MANAGEMENT, SEGMENT_SPACE_MANAGEMENT
FROM DBA_TABLESPACES
WHERE TABLESPACE_NAME='HRTBS';
```

22. Check the datafile name created for the tablespace.

The datafile is created in the following path format:

```
<OMF>/<CDB DB name>/<pdb GUID>/datafile/<datafile name>
```

```
SELECT FILE_NAME FROM DBA_DATA_FILES WHERE TABLESPACE_NAME='HRTBS';
```

23. Check the attributes of the datafile created for the tablespace.

The initial size of the datafile is 100M. It gets incremented by nearly 100M when it needs to. It is auto-extensible up to the maximum size of nearly 32G.

Note: The INCREMENT_BY column represents the number of blocks used for autoextension increment.

```
@ /home/oracle/display_hrtbs_datafiles.sql
```

24. Increase the size of the datafile to 110M.

Obtain the file id from the output of the preceding step.

```
ALTER DATABASE DATAFILE <file id> RESIZE 110M;
```

25. Chang the increment value for the datafile to 10M.

Observe that to change a datafile attribute, we use the ALTER DATABASE statement, not the ALTER TABLESPACE statement. In addition to that, there is no ALTER DATAFILE command.

```
ALTER DATABASE DATAFILE <file id> AUTOEXTEND ON NEXT 10M ;
```

26. Verify that the changes take effect.

```
@ /home/oracle/display_hrtbs_datafiles.sql
```

27. Let's try adding a datafile to the tablespace that does not belong to the OMF.

When you add a datafile to the tablespace, you can set the datafile attributes only. But you cannot set the tablespace attributes like EXTENT MANAGEMENT because it is already set when the tablespace was created.

```
ALTER TABLESPACE hrtbs ADD DATAFILE '/home/oracle/hrbts2.dbf' SIZE 10M
AUTOEXTEND OFF ;
```

28. Verify that the datafile is added.

```
@ /home/oracle/display_hrtbs_datafiles.sql
```

29. Try adding a datafile to the tablespace that complies with the OMF.

The code uses the OMF to save the generated datafile under the OMF directory. In the same time, the code overrides the default size attribute and makes it 20M (instead of 100M).

```
ALTER TABLESPACE hrtbs ADD DATAFILE SIZE 20M;
```

30. Verify that the datafile is added.

```
@ /home/oracle/display_hrtbs_datafiles.sql
```

31. Login to PDB1 as HR

```
conn hr/ABcd##1234@//srv1/pdb1.localdomain
```

32. Try moving the testing table to the HRTBS tablespace.

```
ALTER TABLE TEST_TABLE MOVE TABLESPACE HRTBS;
```

33. Verify that the table is saved in the HRTBS tablespace.

```
SELECT TABLESPACE_NAME FROM USER_TABLES WHERE TABLE_NAME='TEST_TABLE';
```

34. Display the segment information for the testing table.

Although the table is empty, after moving it to another tablespace, it got recreated in the tablespace with a single extent. This means the table got compacted.

Although moving table to a different tablespace does perform table defragmentation, this method is not the only shrinking method. You will learn later in the course other shrinking techniques.

```
@ /home/oracle/display_segment.sql
```

35. Try inserting data into the table.

HR is able to add data to the tablespace because it has the UNLIMITED TABLESPACE privilege. You will learn about system privileges and tablespace quotas later in the course.

```
@ /home/oracle/insertrows.sql 1000
```

Cleanup

36. Drop the HRTBS tablespace

This operation leads to dropping the TEST_TABLE.

```
conn SYSTEM/ABcd##1234@//srv1/pdb1.localdomain

DROP TABLESPACE HRTBS INCLUDING CONTENTS AND DATAFILES;
```

37. Exit from SQL*Plus and remove the script files created in the practice.

```
exit

rm /home/oracle/insertrows.sql
rm /home/oracle/display_segment.sql
rm /home/oracle/display_hrtbs_datafiles.sql
```

Exploring CDB Tablespaces and Datafiles

In the following section, you will explore the data files in the CDB.

38. Connect to the CDB root as sysdba

```
sqlplus / as sysdba
```

39. Retrieve the default permanent and temporary tablespaces in the root container.

```
col property_name format a30
col property_value format a25

SELECT PROPERTY_NAME, PROPERTY_VALUE
FROM   DATABASE_PROPERTIES
WHERE  PROPERTY_NAME LIKE 'DEFAULT_%TABLE%';
```

40. Retrieve the permanent and temporary tablespaces in the entire database.

Same as the root container, there are SYSTEM, SYSAUX, temporary, and undo tablespaces in PDB1.

Notice that the tablespaces of the Seed container are not retrieved by the dictionary view.

```
col pdb_name format a10

SELECT T.TABLESPACE_NAME, T.CON_ID, P. PDB_NAME
FROM   CDB_TABLESPACES T, CDB_PDBS P
WHERE  T.CON_ID = P.CON_ID (+)
ORDER BY 2,1;
```

41. View all the data files of the entire database.

- Observe that the datafiles of the seed (CON_ID = 2) were not retrieved by the query. Oracle 12.1 introduced the parameter EXCLUDE_SEED_CDB_VIEW to make this view retrieving the datafiles of the root. In Oracle 12.2, this parameter has been made obsolete.
- Observe that the PDB name is not included in the view queried below. To get it, you need to link the view with the CDB_PDBS, as demonstrated in the previous step code.

```
col file_name format a50
col tablespace_name format a8
col file_id format 9999
col con_id format 999

SELECT FILE_NAME, TABLESPACE_NAME, FILE_ID, CON_ID
FROM CDB_DATA_FILES
ORDER BY CON_ID;
```

42. Retrieve the data files from the DBA_DATA_FILES.

Notice only the data files of the root have been retrieved.

```
SELECT FILE_NAME, TABLESPACE_NAME, FILE_ID FROM DBA_DATA_FILES;
```

43. Retrieve the data files using the views V$DATAFILE and V$TABLESPACE.

Notice that all the data files have been retrieved, including the seed data files.

```
col name format a12
SELECT FILE#, T.NAME, T.TS#, T.CON_ID
FROM   V$DATAFILE D, V$TABLESPACE T
WHERE  D.TS#=T.TS# AND D.CON_ID=T.CON_ID
ORDER BY 4,3;
```

44. Change the current container to the seed and retrieve the data file names from DBA_DATA_FILES.

```
ALTER SESSION SET CONTAINER=PDB$SEED;

col FILE_NAME for a80
SELECT FILE_NAME FROM DBA_DATA_FILES;
```

In the following steps, you will execute two methods to display the space usage by the tablespaces.

45. Run the following query as SYS to display all the tablespaces in the CDB and their space usage.

Because we want to retrieve the information for the entire CDB, we refer to the CDB_* views.

```
conn / as sysdba

set linesize 180
col CON_ID for 99
col TABLESPACE_NAME for A15
col "Free %" for A8

SELECT F.CON_ID
      ,F.TABLESPACE_NAME
      ,F.TOTALSPACE "Size MB"
      ,(F.TOTALSPACE - U.TOTALUSEDSPACE) "Free MB"
      ,ROUND (100 * ( (F.TOTALSPACE - U.TOTALUSEDSPACE) / F.TOTALSPACE)) || '%'
"Free %"
      ,T.MAX_S "Max Size"
FROM   (SELECT   CON_ID, TABLESPACE_NAME, ROUND (SUM (BYTES) / (1024 * 1024))
TOTALSPACE
        FROM     CDB_DATA_FILES
        GROUP BY CON_ID, TABLESPACE_NAME) F
      ,(SELECT   CON_ID, TABLESPACE_NAME, ROUND (SUM (BYTES) / (1024 * 1024))
TOTALUSEDSPACE
        FROM     CDB_SEGMENTS
        GROUP BY CON_ID, TABLESPACE_NAME) U
      ,(SELECT CON_ID, TABLESPACE_NAME, ROUND (MAX_SIZE / (1024 * 1024)) MAX_S
        FROM     CDB_TABLESPACES        ) T
WHERE  (F.TABLESPACE_NAME = U.TABLESPACE_NAME AND F.CON_ID= U.CON_ID)
  AND  (F.TABLESPACE_NAME = T.TABLESPACE_NAME AND F.CON_ID = T.CON_ID)
ORDER BY CON_ID ;
```

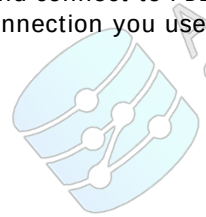
46. Run the following query at PDB1 to display all its tablespaces and their space usage.

Because we want to retrieve the information at the PDB level, we refer to the DBA_* views.

```
ALTER SESSION SET CONTAINER=PDB1;

SELECT F.TABLESPACE_NAME
      ,F.TOTALSPACE "Size MB"
      , (F.TOTALSPACE - U.TOTALUSEDSPACE) "Free MB"
      ,ROUND (100 * ( (F.TOTALSPACE - U.TOTALUSEDSPACE) / F.TOTALSPACE)) || '%'
"Free %"
      ,T.MAX_S "Max Size"
FROM   (SELECT   TABLESPACE_NAME, ROUND (SUM (BYTES) / (1024 * 1024)) TOTALSPACE
        FROM     DBA_DATA_FILES
        GROUP BY TABLESPACE_NAME) F
      ,(SELECT   TABLESPACE_NAME, ROUND (SUM (BYTES) / (1024 * 1024))
TOTALUSEDSPACE
        FROM     DBA_SEGMENTS
        GROUP BY TABLESPACE_NAME) U
      ,( SELECT TABLESPACE_NAME, ROUND (MAX_SIZE / (1024 * 1024)) MAX_S
        FROM     DBA_TABLESPACES) T
WHERE  F.TABLESPACE_NAME = U.TABLESPACE_NAME(+)
  AND  F.TABLESPACE_NAME = T.TABLESPACE_NAME;
```

47. Open SQL Developer and connect to PDB1 as SYSTEM. Then display the tablespace space usage by right-clicking on the connection you used under the connection node > select **Manage Database**.



Summary

- By default, when a segment is created, no extent is created for it. Extents and blocks are created for the table when the first row is inserted into the segment.
- By default, newly created tables are saved into the owner's default tablespace.
- OMF facilitates creating and managing tablespaces.
- Even when OMF is enabled, we can create datafiles for tablespaces in a directory different than the OMF setting.
- When a segment is moved to a different tablespace, if the segment contains gabs in its data blocks (due to delete or update operations), the segment gets compacted. Its new size equals to only the size of data contained in the segment.
- Data files data dictionary views exclude the Seed container datafiles.