https://christian-gohmann.de/2020/09/17/cleanup-of-temporary-segments/

**Cleanup of TEMPORARY segments**

In this short blog post, I will explain how to get rid of TEMPORARY segments in a permanent tablespace.

**Table of Contents** [[hide](https://christian-gohmann.de/2020/09/17/cleanup-of-temporary-segments/)]

**What is a TEMPORARY segment?**

When you hear the term TEMPORARY, you might find about segments in a temporary tablespace, for example when you join tables or perform a sort operation that does not fit into the memory.

But such segments can also exist in a permanent tablespace. Such a segment is created, when you perform one of the following operations.

* ALTER TABLE … MOVE
* CREATE TABLE … AS SELECT
* CREATE INDEX …
* ALTER INDEX … REBUILD

During these operations, a TEMPORARY segment is created. When the operation terminates unexpectedly, these segments are not cleaned up.

**Identify TEMPORARY segments**

With the help of the following queries, TEMPORARY segments can be identified for a specific tablespace. The first query reads the information from DBA\_SEGMENTS.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | SQL> SELECT segment\_name, segment\_type, header\_file, relative\_fno, header\_block,  bytes/POWER(1024, 2) AS "SIZE\_MB"  FROM dba\_segments  WHERE tablespace\_name = 'USERS'  AND segment\_type = 'TEMPORARY';    SEGMENT\_NAME SEGMENT\_TYPE HEADER\_FILE RELATIVE\_FNO HEADER\_BLOCK SIZE\_MB  -------------------- -------------------- ----------- ------------ ------------ ----------  395.159978 TEMPORARY 395 4 159978 4  395.605266 TEMPORARY 395 4 605266 20454 |

The naming convention of the segment name is **<Header File ID>.<Header Block ID>**.

As an alternative to DBA\_SEGMENTS, the table SEG$ can be quiered directly.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | SQL> SELECT file#, block#, type#, ts#  FROM seg$  WHERE type#=3  AND ts# = (SELECT ts# FROM v$tablespace WHERE name = 'USERS');    FILE# BLOCK# TYPE# TS#  ---------- ---------- ---------- ----------  4 159978 3 6  4 605266 3 6 |

TEMPORARY segments occupy space in the tablespace and should be cleaned up.

**Cleanup of TEMPORARY segments**

Normally the user process who created the TEMPORARY segment is responsible for its cleanup (e.g. convert the TEMPORARY segment to an INDEX segment). For sort operations or if the session was terminated, SMON performs a cleanup of these segments. **But it seems that SMON does not drop all “types” of TEMPORARY segments – bug or expected behavior?!**

PL/SQL package [DBMS\_SPACE\_ADMIN](https://docs.oracle.com/en/database/oracle/oracle-database/19/arpls/DBMS_SPACE_ADMIN.html) is used to clean up TEMPORARY segments manually. In the first step, the segment **395.605266** is marked as corrupt. **The relative file number is required at this point – use the value of column RELATIVE\_FNO.**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | SQL> BEGIN  DBMS\_SPACE\_ADMIN.SEGMENT\_CORRUPT(  tablespace\_name => 'USERS',  header\_relative\_file => 4,  header\_block => 605266,  corrupt\_option => DBMS\_SPACE\_ADMIN.SEGMENT\_MARK\_CORRUPT  );  END;  / |

After the segment is marked as corrupt, it can be dropped.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | SQL> BEGIN  DBMS\_SPACE\_ADMIN.SEGMENT\_DROP\_CORRUPT(  tablespace\_name => 'USERS',  header\_relative\_file => 4,  header\_block => 605266  );  END;  / |

At this point the segment is dropped, but its occupied space is not visible in DBA\_FREE\_SPACE.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | SQL> SELECT SUM(bytes)/POWER(1024,3) AS "FREE\_SPACE"  FROM dba\_free\_space  WHERE tablespace\_name = 'USERS';    FREE\_SPACE  ----------  337.322449 |

As final step, a rebuild of the bitmaps of the tablespace is required.

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | SQL> BEGIN  DBMS\_SPACE\_ADMIN.TABLESPACE\_REBUILD\_BITMAPS(  tablespace\_name => 'USERS'  );  END;  / |

Now the free space increases by the size of the dropped segments.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | SQL> SELECT SUM(bytes)/POWER(1024,3) AS "FREE\_SPACE"  FROM dba\_free\_space  WHERE tablespace\_name = 'USERS';    FREE\_SPACE  ----------  357.300964 |

**References**

* [Temporary Segments In Permanent Tablespaces Aren’t Cleaned For A Long Time (Doc ID 1271120.1)](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1271120.1)
* [How to clear a block corruption in a TEMPORARY segment (Doc ID 1332088.1)](https://support.oracle.com/epmos/faces/DocumentDisplay?id=1332088.1)

**https://askmedawaa.wordpress.com/2020/11/22/shrinking-a-temporary-tablespace-in-oracle-database/**

**1.. First check the Temp Tablespace Total size, used space and free space .**

SELECT TABLESPACE\_NAME,SUM(TABLESPACE\_SIZE)/1024/1024 TOTAL\_SIZE,

SUM(ALLOCATED\_SPACE )/1024/1024 USED\_SPACE,

SUM(FREE\_SPACE)/1024/1024 SPACE\_FREE FROM DBA\_TEMP\_FREE\_SPACE

GROUP BY TABLESPACE\_NAME,TABLESPACE\_SIZE,ALLOCATED\_SPACE,FREE\_SPACE;

**[Afbeelding met tekst, schermopname, Lettertype

Automatisch gegenereerde beschrijving](https://askmedawaa.files.wordpress.com/2020/11/10.png)**

**2.. Now check the Temp tablespace data file location and name etc.**

select FILE\_NAME,TABLESPACE\_NAME,sum(bytes)/1024/1024 Total\_MB from dba\_temp\_files group by FILE\_NAME,TABLESPACE\_NAME;

**[Afbeelding met schermopname, tekst, Lettertype, lijn

Automatisch gegenereerde beschrijving](https://askmedawaa.files.wordpress.com/2020/11/11.png)**

**3.. Now using The SHRINK SPACE clause we can shrink a temporary tablespace,**

Whereas SHRINK TEMPFILE allows the shrink of a temporary file.

ALTER TABLESPACE tablespace SHRINK SPACE or TEMPFILE tempfile\_name [KEEP size];

Syntax:-

ALTER TABLESPACE TEMP1 SHRINK SPACE KEEP 200M;

**[Afbeelding met schermopname, tekst, Lettertype

Automatisch gegenereerde beschrijving](https://askmedawaa.files.wordpress.com/2020/11/12.png)**

ALTER TABLESPACE TEMP1 SHRINK SPACE;

**[Afbeelding met schermopname, tekst, Lettertype, lijn

Automatisch gegenereerde beschrijving](https://askmedawaa.files.wordpress.com/2020/11/100-3.png)**

ALTER TABLESPACE TEMP1 SHRINK TEMPFILE ‘/u001/app/oracle/oradata/prod/temp101.dbf’ KEEP 200M;

**[Afbeelding met schermopname, tekst, Lettertype, lijn

Automatisch gegenereerde beschrijving](https://askmedawaa.files.wordpress.com/2020/11/101-3.png)**

Note :-

The optional KEEP clause defines the lower bound that a tablespace can be shrunk to.

If it is not specified, the system will try to shrink as much as possible, as long as the other storage attibutes are satisfied and you can refer given below sample as TEMP Tablespace was shrinked without KEEP clause;

**[Afbeelding met tekst, schermopname, Lettertype

Automatisch gegenereerde beschrijving](https://askmedawaa.files.wordpress.com/2020/11/200.png)**

Otherwise, shrink will stop once the tablespace/tempfile already reaches the size specified through the KEEP option.