180
DBMS_SHARED_POOL

The DBMS_SHARED_POOL package provides access to the shared pool, which is the shared memory area where cursors and PL/SQL objects are stored. DBMS_SHARED_POOL enables you to display the sizes of objects in the shared pool, and mark them for keeping or not-keeping in order to reduce memory fragmentation.

This chapter contains the following topics:

- Overview
- Operational Notes
- Summary of DBMS_SHARED_POOL Subprograms

DBMS_SHARED_POOL Overview

The procedures provided here may be useful when loading large PL/SQL objects. When large PL/SQL objects are loaded, users response time is affected because of the large number of smaller objects that need to be aged out from the shared pool to make room (due to memory fragmentation). In some cases, there may be insufficient memory to load the large objects.

DBMS_SHARED_POOL is also useful for frequently executed triggers. You may want to keep compiled triggers on frequently used tables in the shared pool.

Additionally, DBMS_SHARED_POOL supports sequences. Sequence numbers are lost when a sequence is aged out of the shared pool. DBMS_SHARED_POOL is useful for keeping sequences in the shared pool and thus preventing the loss of sequence numbers.

DBMS_SHARED_POOL Operational Notes

To create DBMS_SHARED_POOL, run the DBMSPOOL.SQL script. The PRVTPOOL.PLB script is automatically executed after DBMSPOOL.SQL runs. These scripts are *not* run by as part of standard database creation.

Summary of DBMS_SHARED_POOL Subprograms

This table lists the DBMS_SHARED_POOL subprograms and briefly describes them.

Table 180-1 ***DBMS_SHARED_POOL Package Subprograms***

Subprogram	Description
ABORTED_REQUEST_THRESHOLD Procedure	Sets the aborted request threshold for the shared pool
KEEP Procedure	Keeps an object in the shared pool
MARKHOT Procedure	Marks a library cache object as a hot object
PURGE Procedure	Purges the named object or specified heap(s) of the object

Table 180-1 (Cont.) *DBMS_SHARED_POOL Package Subprograms*

Subprogram	Description
SIZES Procedure	Shows objects in the shared pool that are larger than the specified size
UNKEEP Procedure	Unkeeps the named object
UNMARKHOT Procedure	Unmarks a library cache object as a hot object

ABORTED_REQUEST_THRESHOLD Procedure

This procedure sets the aborted request threshold for the shared pool.

Syntax

```
DBMS_SHARED_POOL.ABORTED_REQUEST_THRESHOLD (
   threshold_size NUMBER);
```

Parameters

Table 180-2 ABORTED_REQUEST_THRESHOLD Procedure Parameters

Parameter	Description
`threshold_size`	Size, in bytes, of a request which does not try to free unpinned (not "unkeep-ed") memory within the shared pool. The range of `threshold_size` is 5000 to ~2 GB inclusive.

Exceptions

An exception is raised if the threshold is not in the valid range.

Usage Notes

Usually, if a request cannot be satisfied on the free list, then the RDBMS tries to reclaim memory by freeing objects from the LRU list and checking periodically to see if the request can be fulfilled. After finishing this step, the RDBMS has performed a near equivalent of an '`ALTER SYSTEM FLUSH SHARED_POOL`'.

Because this impacts all users on the system, this procedure "localizes" the impact to the process failing to find a piece of shared pool memory of size greater than `thresh_hold` size. This user gets the 'out of memory' error without attempting to search the LRU list.

KEEP Procedure

This procedure keeps an object in the shared pool. Once an object has been kept in the shared pool, it is not subject to aging out of the pool. This may be useful for frequently used large objects. When large objects are brought into the shared pool, several objects may need to be aged out to create a contiguous area large enough.

Syntax

```
DBMS_SHARED_POOL.KEEP (
   name         VARCHAR2,
   flag         CHAR DEFAULT 'P');
```

```
DBMS_SHARED_POOL.KEEP (
   schema          VARCHAR2,
   objname         VARCHAR2,
   namespace       NUMBER,
   heaps           NUMBER,
   edition_name    VARCHAR2  DEFAULT NULL);

DBMS_SHARED_POOL.KEEP (
   hash            VARCHAR2,
   namespace       NUMBER,
   heaps           NUMBER);
```

Parameters

Table 180-3   KEEP Procedure Parameters

Parameter	Description
`name`	The name of the object to keep.
`flag`	A character string indicating the kind of object to keep in the shared pool. The string is not case sensitive. This parameter is optional. If the parameter is not specified, the package assumes that the first parameter is the name of a package/procedure/function and will resolve the name. • Set to 'P' or 'p' to fully specify that the input is the name of a package/procedure/function. • Set to 'Q' or 'q' to specify that the input is the name of a sequence. • Set to 'R' or 'r' to specify that the input is the name of a trigger. • Set to 'T' or 't' to specify that the input is the name of a type. • Set to 'JS' or 'js' to specify that the input is the name of a java source. • Set to 'JC' or 'jc' to specify that the input is the name of a java class. • Set to 'JD' or 'jd' to specify that the input is the name of a java shared data. • Set to 'JR' or 'jr' to specify that the input is the name of a java resource. • Set to 'C' or 'c' to specify that the input is the name of a cursor.
`schema`	The user name or the schema to which the object belongs to.
`objname`	The name of the object to keep.
`namespace`	A number indicating the library cache namespace in which the object has to be searched for.
`heaps`	The heaps to keep. For example, if heap 0 and heap 6 are to be kept.
`edition_name`	The name of the edition that the target object resides in. This parameter is optional.
`hash`	A 16-byte hash value for the object.

Exceptions

An exception is raised if the named object is not found.

Usage Notes

There are two kinds of objects:

- PL/SQL objects, triggers, sequences, types, and Java objects, which are specified by name.

- SQL cursor objects which are specified by a two-part number (indicating a location in the shared pool).

For example:

```
DBMS_SHARED_POOL.KEEP('scott.hispackage')
```

This keeps package HISPACKAGE, owned by SCOTT. The names for PL/SQL objects follow SQL rules for naming objects (for example, delimited identifiers and multibyte names are allowed). A cursor can be kept by DBMS_SHARED_POOL.KEEP('0034CDFF, 20348871','C'), 0034CDFF being the ADDRESS and 20348871 the HASH_VALUE. Note that the complete hexadecimal address must be in the first 8 characters.

MARKHOT Procedure

This procedure marks a library cache object as a hot object.

Syntax

```
DBMS_SHARED_POOL.MARKHOT (
   schema         VARCHAR2,
   objname        VARCHAR2,
   namespace      NUMBER DEFAULT 1,
   global         BOOLEAN DEFAULT TRUE,
   edition_name   VARCHAR2 DEFAULT NULL);

DBMS_SHARED_POOL.MARKHOT (
   hash          VARCHAR2,
   namespace     NUMBER DEFAULT 1,
   global        BOOLEAN DEFAULT TRUE);
```

Parameters

Table 180-4 MARKHOT Procedure Parameters

Parameter	Description
schema	User name or the schema to which the object belongs
objname	Name of the object
namespace	Number indicating the library cache namespace in which the object is to be searched. Views, such as USER_OBJECTS and DBA_OBJECTS, reflect the namespace as a number column, as do most dictionary tables such as OBJ$.
global	If TRUE (default), mark the object hot on all Oracle RAC instances
hash	16-byte hash value for the object
edition_name	Denotes the name of the edition that the target object resides in. This parameter is optional.

Exceptions

ORA-06502: An exception is raised if the named object cannot be found due to incorrect input

ORA-04043: An exception is raised if the named object cannot be found (bad namespace, or hash input)

Usage Notes

If a package or type's specification is marked hot or unhot, then the corresponding package or type body will be implicitly marked as hot or unhot.

Users can examine column, V$DB_OBJECT_CACHE.PROPERTY, to see whether or not the object has been marked hot. The values for PROPERTY are:

- HOTCOPYnnn- An object that is a hot copy with integer identifier of 'nnn'. For example, HOTCOPY5, HOTCOPY94, and HOTCOPY125.
- HOTCOPY-As above but the identifier is unknown
- HOT-The "root" kgl object that has been marked as hot
- NULL-A normal object

PURGE Procedure

This procedure purges the named object or specified heaps of the object.

Syntax

```
DBMS_SHARED_POOL.PURGE (
   name         VARCHAR2,
   flag         CHAR DEFAULT 'P',
   heaps        NUMBER DEFAULT 1);

DBMS_SHARED_POOL.PURGE (
   schema       VARCHAR2,
   objname      VARCHAR2,
   namespace    NUMBER,
   heaps        NUMBER,
   edition_name VARCHAR2 DEFAULT NULL);

DBMS_SHARED_POOL.PURGE (
   hash         VARCHAR2,
   namespace    NUMBER,
   heaps        NUMBER);
```

Parameters

Table 180-5 PURGE Procedure Parameters

Parameter	Description
name	Name of the object to purge. The value for this identifier is the concatenation of the address and hash_value columns from the v$sqlarea view. This is displayed by the SIZES procedure. Currently, TABLE and VIEW objects may not be purged.

Table 180-5 (Cont.) PURGE Procedure Parameters

Parameter	Description
`flag`	A character string indicating the kind of object to purge from the shared pool. The string is not case sensitive. This parameter is optional. If the parameter is not specified, the package assumes that the first parameter is the name of a package/procedure/function and will resolve the name. • Set to 'P' or 'p' to fully specify that the input is the name of a package/procedure/function. • Set to 'Q' or 'q' to specify that the input is the name of a sequence. • Set to 'R' or 'r' to specify that the input is the name of a trigger. • Set to 'T' or 't' to specify that the input is the name of a type. • Set to 'JS' or 'js' to specify that the input is the name of a java source. • Set to 'JC' or 'jc' to specify that the input is the name of a java class. • Set to 'JD' or 'jd' to specify that the input is the name of a java shared data. • Set to 'JR' or 'jr' to specify that the input is the name of a java resource. • Set to 'C' or 'c' to specify that the input is the name of a cursor.
`heaps`	Heaps to be purged. For example, if heap 0 and heap 6 are to be purged: 1<<0 | 1<<6 => hex 0x41 => decimal 65, so specify heaps =>65. Default value is `1`, that is, heap 0 which means the whole object would be purged
`schema`	User name or the schema to which the object belongs
`objname`	Name of the object to purge
`namespace`	Parameter is a number indicating the library cache namespace in which the object is to be searched
`hash`	16-byte hash value for the object
`edition_name`	The name of the edition that the target object resides in. This parameter is optional.

Exceptions

`ORA-6570`: An exception is raised if the named object cannot be found

`ORA-6570`: An object cannot be purged it marked as permanently kept

Usage Notes

All objects supported by the KEEP Procedure are supported for `PURGE`.

PURGE_ROW_CACHE Procedure

This procedure purges the row cache as identified by cache id, or a specific object within the selected cache.

Syntax

```
DBMS_SHARED_POOL.PURGE_ROW_CACHE (
   cacheid number, hash varchar2 DEFAULT null);
```

Parameters

Table 180-6 PURGE_ROW_CACHE Procedure Parameters

Parameter	Description
`cacheid`	Indicates which cache to purge.
`hash`	The hash value of a targeted object to purge within the named cache. The default value (`NULL`) purges the entire cache.

SIZES Procedure

This procedure shows objects in the `shared_pool` that are larger than the specified size. The name of the object is also given, which can be used as an argument to either the `KEEP` or `UNKEEP` calls.

Syntax

```
DBMS_SHARED_POOL.SIZES (
   minsize NUMBER);
```

Parameters

Table 180-7 SIZES Procedure Parameters

Parameter	Description
`minsize`	Size, in kilobytes, over which an object must be occupying in the shared pool, in order for it to be displayed.

Usage Notes

Issue the SQLDBA or SQLPLUS `'SET SERVEROUTPUT ON SIZE XXXXX'` command prior to using this procedure so that the results are displayed.

UNKEEP Procedure

This procedure unkeeps the named object.

Syntax

```
DBMS_SHARED_POOL.UNKEEP (
   name         VARCHAR2,
   flag         CHAR DEFAULT 'P');

DBMS_SHARED_POOL.UNKEEP (
   schema       VARCHAR2,
   objname      VARCHAR2,
   namespace    NUMBER,
   edition_name VARCHAR2 DEFAULT NULL);

DBMS_SHARED_POOL.UNKEEP (
   hash         VARCHAR2,
   namespace    NUMBER);
```

WARNING:

This procedure may not be supported in the future if automatic mechanisms are implemented to make this unnecessary.

Parameters

Table 180-8 UNKEEP Procedure Parameters

Parameter	Description
`name`	The name of the object to unkeep.
`flag`	A character string indicating what kind of object to keep the name identifies. The string is not case sensitive. This parameter is optional. If the parameter is not specified, the package assumes that the first parameter is the name of a package/procedure/function and will resolve the name.
`schema`	The user name or the schema to which the object belongs to.
`objname`	The name of the object to unkeep.
`namespace`	A number that indicates the library cache namespace in which the object has to be searched for.
`edition_name`	The name of the edition that the target object resides in. This parameter is optional.
`hash`	A 16-byte hash value for the object.

Exceptions

`ORA-06502`: An exception is raised if the named object cannot be found

UNMARKHOT Procedure

This procedure unmarks a library cache object as a hot object.

Syntax

```
DBMS_SHARED_POOL.UNMARKHOT (
   schema          VARCHAR2,
   objname         VARCHAR2,
   namespace       NUMBER  DEFAULT 1,
   global          BOOLEAN DEFAULT TRUE,
   edition_name    VARCHAR2 DEFAULT NULL);

DBMS_SHARED_POOL.UNMARKHOT (
   hash           VARCHAR2,
   namespace      NUMBER DEFAULT 1,
   global         BOOLEAN DEFAULT TRUE);
```

Parameters

Table 180-9 UNMARKHOT Procedure Parameters

Parameter	Description
`schema`	User name or the schema to which the object belongs
`objname`	Name of the object
`namespace`	Number indicating the library cache namespace in which the object is to be searched
`global`	If `TRUE`, unmark the object hot on all Oracle RAC instances. The default value of this parameter is `TRUE`.
`hash`	A 16-byte hash value for the object
`edition_name`	Denotes the name of the edition that the target object resides in. This parameter is optional.

Exceptions

`ORA-06502`: An exception is raised if the named object cannot be found due to incorrect input

`ORA-04043`: An exception is raised if the named object cannot be found (bad namespace, or hash input, or non-existent object)

Usage Notes

If a package or type's specification is marked hot or unhot, then the corresponding package or type body will be implicitly marked as hot or unhot.