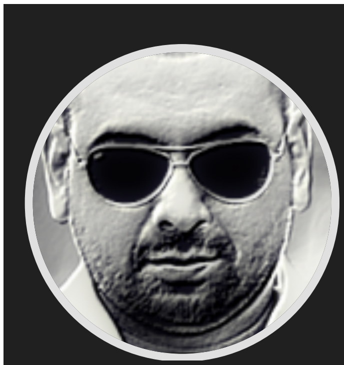


Nabeel Khan

An Interactive Technologist

Contact
me for
more info

About Me Professional Summary Technical Qualifications Experience

Blog Contact Me

Previous Next

Using Oracle AMM (Automatic Memeory Managment) 11g

The basic memory structures associated with Oracle Database include:

System Global Area (SGA)

The SGA is a group of shared memory structures, known as SGA components, that contain data and control information for one Oracle Database instance. The SGA is shared by all server and background processes.

Program Global Area (PGA)

A PGA is a memory region that contains data and control information for a server process. It is nonshared memory created by Oracle Database when a server process is started. Access to the PGA is exclusive to the server process. There is one PGA for each server process. Background processes also allocate their own PGAs. The total PGA memory allocated for all background and server processes

Find me on:

Recent Posts

Islamic Invocations with Urdu Translation

September 1, 2011

New Features of Firefox 5

June 22, 2011

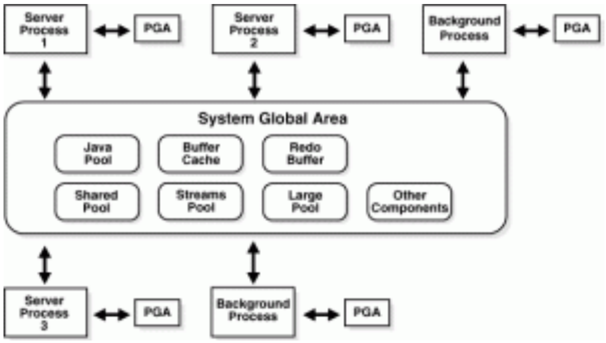
Difference between utlchain.sql and utlchn1.sql

attached to an Oracle Database instance is referred to as the total instance PGA memory, and the collection of all individual PGAs is referred to as the total instance PGA, or just instance PGA.

It contains global variables and data structures and control information for a server process. example of such information is the runtime area of a cursor. Each time a cursor is executed, a new runtime area is created for that cursor in the PGA memory region of the server process executing that cursor.

The performance of complex long running queries, typical in a DSS environment, depend to a large extent on the memory available in the Program Global Area (PGA) which is also called work area.

Below figure illustrates the relationships among these memory structures.



Oracle Memory Structure

Memory management has evolved with each database release

Oracle 9i

Beginning with Oracle9i, the dynamic SGA infrastructure allowed for the sizing of the Buffer Cache, Shared Pool and the Large Pool without having to shutdown the database. Key features being
Dynamic Memory resizing
DB_CACHE_SIZE instead of DB_BLOCK_BUFFERS
DB_nK_CACHE_SIZE for multiple block sizes
PGA_AGGREGATE_TARGET Introduction of Automatic PGA Memory management

Oracle Database 10g

Automatic Shared Memory Management (ASMM) was introduced in 10g. You enable the automatic shared memory management feature by setting the SGA_TARGET parameter to a non-zero value.

Oracle Database 11g

March 22, 2011

Using Oracle AMM (Automatic Memeory Managment) 11g

March 21, 2011

Enable Tracing Oracle Listener

March 21, 2011

+1

g+1 Recommander ce contenu

Find us on Facebook

Automatic Memory Management is being introduced in 11g. This enables automatic tuning of PGA and SGA with use of two new parameters named MEMORY_MAX_TARGET and MEMORY_TARGET.

The most important SGA components are the following:

> *DB_CACHE_SIZE*
> *DB_KEEP_CACHE_SIZE*
> *DB_RECYCLE_CACHE_SIZE*
> *DB_nK_CACHE_SIZE*

Database Buffer Cache: The database buffer cache is the portion of the SGA that holds copies of data blocks read from datafiles. All users concurrently connected to the instance share access to the database buffer cache.

> *LOG_BUFFER*

Redo Log Buffer: The redo log buffer is a circular buffer in the SGA that holds information about changes made to the database. This information is stored in redo entries. Redo entries contain the information necessary to reconstruct, or redo, changes made to the database by INSERT, UPDATE, DELETE, CREATE, ALTER, or DROP operations. Redo entries are used for database recovery, if necessary.

> *SHARED_POOL_SIZE*
> *SHARED_POOL_RESERVED_SIZE*
> *RESULT_CACHE_SIZE **

Shared Pool:
The shared pool portion of the SGA contains the library cache, the dictionary cache, the result cache, buffers for parallel execution messages, and control structures.

> *LARGE_POOL_SIZE*

Large Pool: Used for allocating session memory for shared server, Oracle XA, or parallel query buffers or for RMAN.

> *JAVA_POOL_SIZE*

Java Pool : Java pool memory is used in server memory for all session-specific Java code and data within the JVM.

> *STREAMS_POOL_SIZE*

Streams Pool : The streams pool is used exclusively by Oracle Streams. The Streams pool stores buffered queue messages, and it provides memory for Oracle Streams capture processes and apply processes.

> ** RESULT_CACHE_MAX_SIZE is new component which has been introduced as part of 11g Memory architecture. The result cache is composed of the SQL query result cache and PL/SQL function result cache, which share the same infrastructure.Results of queries and query fragments can be cached in memory in the SQL query result cache. The database can then use cached results to answer future executions of these queries and query fragments. Similarly PL/SQL Function Result can also be cached.*

You have to use RESULT_CACHE_MODE initialization parameter which determines the SQL query result cache behavior. The possible initialization parameter values are MANUAL and FORCE.

You can use various memory management types from following:

For Both the SGA and Instance PGA – Automatic Memory Management
For the SGA - Automatic Shared Memory Management
For the Instance PGA – Automatic PGA Memory Management

1.Automatic Memory Management (SGA & PGA)

Oracle Database can manage the SGA memory and instance PGA memory completely automatically by setting 2 parameters, MEMORY_MAX_TARGET and MEMORY_TARGET. You designate only the total memory size to be used by the instance, and Oracle Database dynamically exchanges memory between the SGA and the instance PGA as needed to meet processing demands. This capability is referred to as automatic memory management. With this memory management method, the database also dynamically tunes the sizes of the individual SGA components and the sizes of the individual PGAs. To do so (on most platforms), you set only a target memory size initialization parameter (MEMORY_TARGET) and optionally a maximum memory size initialization parameter (MEMORY_MAX_TARGET).

If you create your database with Database Configuration Assistant (DBCA) and choose the basic installation option, automatic memory management is enabled. If you choose advanced installation, Database Configuration Assistant (DBCA) enables you to select automatic memory management.

Switching to Automatic Memory Management

1)Check the current values configured for SGA_TARGET and PGA_AGGREGATE_TARGET.

```
SQL> show parameter TARGET

NAME              TYPE       VALUE
——————————    —————    —————
archive_lag_target            integer       0
db_flashback_retention_target    integer       1440
fast_start_io_target       integer       0
fast_start_mttr_target       integer       0
memory_max_target       big integer    0
memory_target                big integer    0
parallel_servers_target       integer       256
pga_aggregate_target       big integer    1G
sga_target          big integer    1G
```

```
SQL> show parameter cache_size

NAME              TYPE       VALUE
——————————    —————    —————
client_result_cache_size    big integer    0
db_16k_cache_size       big integer    0
db_2k_cache_size       big integer    0
db_32k_cache_size       big integer    64M
db_4k_cache_size       big integer    64M
db_8k_cache_size       big integer    0
db_cache_size          big integer    1520M
db_flash_cache_size       big integer    0
db_keep_cache_size       big integer    0
db_recycle_cache_size       big integer    0
```

Add the values of pga_aggregate_target and sga_target.

2)Decide on a maximum amount of memory that you would want to allocate to the database which will determine the maximum value for the sum of the SGA and instance PGA sizes. In our case we decide to set to 808M

3)Change the parameter in initialization parameter file.

Using Spfile
============

```
SQL>ALTER SYSTEM SET MEMORY_MAX_TARGET = 8G SCOPE = SPFILE;
SQL>ALTER SYSTEM SET MEMORY_TARGET = 8G SCOPE = SPFILE;
SQL>ALTER SYSTEM SET SGA_TARGET =0 SCOPE = SPFILE;
SQL>ALTER SYSTEM SET PGA_AGGREGATE_TARGET = 0 SCOPE = SPFILE;
```

Using Pfile
===========

If you have started the instance with Pfile, then edit the pfile and set the parameters manually

```
MEMORY_MAX_TARGET = 8G
MEMORY_TARGET = 8G
SGA_TARGET =0
PGA_AGGREGATE_TARGET = 0
```

In case you do not specify any value for MEMORY_MAX_TARGET and only use MEMORY_TARGET then database automatically sets MEMORY_MAX_TARGET to the value of MEMORY_TARGET. **BUT If you only set MEMORY_MAX_TARGET the database will not be in AMM mode, till you set MEMORY_TARGET**

If you omit the line for MEMORY_TARGET and include a value for MEMORY_MAX_TARGET, the MEMORY_TARGET parameter defaults to zero. After startup, you can then dynamically change MEMORY_TARGET to a non-zero value, provided that it does not exceed the value of MEMORY_MAX_TARGET.

MEMORY_MAX_TARGET is a static parameter i.e it cannot be changed Dynamically and Instance has to be bounced for modifying the value. So ensure that you have set it to appropriate value. I have changed values to the following for demonstration purpose.

```
SQL>ALTER SYSTEM SET MEMORY_MAX_TARGET = 8G SCOPE=SPFILE;
SQL>ALTER SYSTEM SET MEMORY_TARGET = 8G SCOPE=SPFILE;
SQL>ALTER SYSTEM SET SGA_MAX_SIZE = 7G SCOPE=SPFILE;
SQL>ALTER SYSTEM SET SGA_TARGET = 7G SCOPE=SPFILE;
SQL>ALTER SYSTEM SET PGA_AGGREGATE_TARGET = 0 SCOPE=SPFILE;
SQL>ALTER SYSTEM SET SHARED_POOL_SIZE = 0 SCOPE=SPFILE;
SQL>ALTER SYSTEM SET JAVA_POOL_SIZE = 0 SCOPE=SPFILE;
SQL>ALTER SYSTEM SET DB_CACHE_SIZE = 0 SCOPE=SPFILE;
SQL>ALTER SYSTEM SET DB_CACHE_SIZE = 0 SCOPE=SPFILE
```

```
*
ERROR at line 1:
ORA-32017: failure in updating SPFILE
ORA-00383: DEFAULT cache for blocksize 8192 cannot be reduced
to zero
```

If you get above error then try:

```
SQL> ALTER SYSTEM RESET DB_CACHE_SIZE SCOPE=SPFILE;
System altered.

ALTER SYSTEM SET DB_4K_CACHE_SIZE = 0 SCOPE=SPFILE;

ALTER SYSTEM SET DB_32K_CACHE_SIZE = 0 SCOPE=SPFILE;
```

I have to adjust the memory manual if I need to set the lower limit even when I am using AMM

4)Shutdown and startup the database

```
SQL> shutdown immediate;
Database closed.
Database dismounted.
ORACLE instance shut down.
SQL> startup mount
ORACLE instance started.

Total System Global Area 7482626048 bytes
Fixed Size                2214416 bytes
Variable Size          2281702896 bytes
Database Buffers        5167382528 bytes
Redo Buffers             31326208 bytes

SQL> show parameter target

NAME                              TYPE       VALUE
------------ ---- -----------
archive_lag_target                 integer     0
db_flashback_retention_target       integer     1440
fast_start_io_target               integer     0
fast_start_mttr_target              integer     0
memory_max_target                   big integer 8G
memory_target                      big integer 8G
pga_aggregate_target                big integer 1G
sga_target                        big integer 7G
```

The preceding steps instruct you to set SGA_TARGET and PGA_AGGREGATE_TARGET to zero so that the sizes of the SGA and instance PGA are tuned up and down as required, without restrictions. You can omit the statements that set these parameter values to zero and leave either or both of the values as positive numbers. In this case, the values act as minimum values for the sizes of the SGA or instance PGA.

Monitoring and Tuning Automatic Memory Management

The dynamic performance view V$MEMORY_DYNAMIC_COMPONENTS shows the current sizes of all dynamically tuned memory components, including the total sizes of the SGA and instance PGA.
The view V$MEMORY_TARGET_ADVICE provides tuning advice for the MEMORY_TARGET initialization parameter.

```
SQL> select * from v$memory_target_advice order by memory_size;
```

You can also use V$MEMORY_RESIZE_OPS which has a circular history buffer of the last 800 SGA resize requests.

2. Automatic Shared Memory Management – For the SGA

If you want to exercise more direct control over the size of the SGA, you can disable automatic memory management and enable automatic shared memory management.This feature was introduced in 10g with a parameter known as SGA_TARGET. When automatic SGA memory management is enabled, the sizes of the different SGA components are flexible and can adapt to the needs of current workload without requiring any additional configuration.

In case you have enabled Automatic Memory Management , then to switch to Automatic Shared Memory Management , please follow below procedure

```
SQL>Alter system set MEMORY_TARGET=0 scope=both;
SQL>Alter system set SGA_TARGET=500M scope=both;
```

3. Automatic PGA Memory Management – For the Instance PGA

While using Automatic memory management , PGA memory is allocated based upon value of MEMORY_TARGET. In case you enable automatic shared memory management or manual shared memory management, you also implicitly enable automatic PGA memory management.

Automatic/Manual PGA memory management is decided by initialization parameter WORKAREA_SIZE_POLICY which is a session- and system-level parameter that can take only two values: MANUAL or AUTO. The default is AUTO.

With automatic PGA memory management, you set a target size for the instance PGA by defining value for parameter named PGA_AGGREGATE_TARGET and sizing of SQL work areas is automatic and all *_AREA_SIZE initialization parameters are ignored for these sessions. This feature is available from 9i.

At any given time, the total amount of PGA memory available to active work areas on the instance is automatically derived from the parameter PGA_AGGREGATE_TARGET. This amount is set to the value of PGA_AGGREGATE_TARGET minus the PGA memory allocated for other purposes (for example, session memory). The resulting PGA memory is then allotted to individual active work areas based on their specific memory requirements.

In case you wish to manually specify the maximum work area size for each type of SQL operator (such as sort or hash-join) then you can enable Manual PGA Memory management.

Set WORKAREA_SIZE_POLICY value to MANUAL and also specify values for *_area_size such as SORT_AREA_SIZE, HASH_AREA_SIZE, BITMAP_MERGE_AREA_SIZE, and CREATE_BITMAP_AREA_SIZE, etc.

Although the Oracle Database 11g supports this manual PGA memory management method, Oracle strongly recommends that you leave automatic PGA memory management enabled.

ηαвєєℓ кнαη Databases Oracle 1 Comment

Share This Story, Choose Your Platform!

Related Posts

One Comment

Abbas June 22, 2011 at 11:22

Thank you very much sir

Leave A Comment

You must be logged in to post a comment.

TAGS

10g 11g AMM 11g asktom Coffee Datafile rename fb Forms 10g Crash hacking hotmail Iceberg investment opportunity Jinitiator Kopi Luwak Kuwait Lifestyle Linux Oracle 11g Oracle 11g Installation on RHEL Oracle 11g R2 Instalaltion on Oracle Linux Oracle Datafile Oracle forms10 Oracle Installation Oracle Listener Trace File Oracle Required Packages Oracle tablespace Oracle Tuning PM Recyclebin Recycle bin contents Redhat Remove WD SmartWare RHEL Scoubt.com Se7en Social Network Sun JPI SUN JRE Sun JRE instead of Oracle Jinitiator Titanic Tuning wget Windows 7 Hacks Windows 7 tweaks Windows Vista ηαвєєℓ кнαη.¢σм

ASK TOM

Pro*C and HINTS

Hi Tom, I have a query with hint which is used in Pro*C like select /*+ index(emp,pk_emp_id) */ from emp where emp_id = l_empid; When we run this Pro*C job and generate a tkprof file this query appears as select /*+ index(emp,pk_emp_id) … […]

How to partition historical data

Hello Tom, I have a data table that is quite large (approx 1,100,000 records) that stores historical data. The structure is the following: CREATE TABLE LINES_MONTHLY_HISTORY (TELEPHONENUMBER VARCHAR2(20 BYTE), SUBACCOUNT … […]

Why is the optimizer so wildly off ?

Hi! I recently had a Problem with 2 SQL-Statements performing an equal query, but with wildly different runtimes. I think I narrowed the problem down to a simple case, where the optimizer is wildly off. The statistics are up to date and fit the actua…

How to control read access to the global context

Hi Tom, I'm wondering if there is a way how to control who has read access to the global context. What I'm trying to achieve is this: Keep encryption key in memory so database sessions can encrypt/decrypt data (DBMS_CRYPTO). I can't stor… […]

What performs better NVL or DECODE for evaluating NULL values

Afternoon, Could anyone tell me which of the following statements would perform better? SELECT 1 FROM DUAL WHERE NVL (NULL, '-1') = NVL

(NULL, '-1') OR SELECT 1 FROM DUAL WHERE DECODE(NULL, NULL, '1', '0') = ... [...]