EXAM OBJECTIVE: AUTOMATIC DATABASE DIAGNOSTIC MONITOR

Oracle 10g provides a number of new methods to monitor and tune the Oracle database. These techniques make the task of tuning considerably simpler. The two methods presented by Oracle with regard to tuning the database are:

- **Automatic Database Diagnostic Monitor (ADDM)** – This feature allows the DBA to identify bottlenecks within the database. Analysis is performed top down, first identifying symptoms and then refining them to reach the root causes of performance problems. The feature in addition to other manageability components can be used to make recommendations to resolve bottlenecks.
- **Server Generated Alerts** – Using this feature the Oracle Server can automatically detect certain undesirable alarming situations. The Oracle server can be configured to respond to such situations and alert the DBA if the situation does arise.
- **Reactive Monitoring** - The DBA can make use of the powerful new data sources as well as performance-reporting capabilities of the Oracle 10g database to detect and resolve bottlenecks.

THE AUTOMATIC DATABASE DIAGNOSTIC MONITOR

This new feature of Oracle 10g is used to proactively monitor the database. It is a server-based. It ties up all sorts of relevant information in the database to perform a detailed analysis and generate a list of recommendations.

The types of problems that ADDM considers include:

- CPU bottlenecks
- Undersized Memory Structures
- I/O capacity issues
- High load SQL statements
- High load PL/SQL execution and compilation, as well as high load Java usage
- RAC specific issues
- Sub-optimal use of Oracle by the application
- Database configuration issues \
- Concurrency issues
- Hot objects and top SQL for various problem areas

Within the database statistics are gathered automatically every 60 minutes and stored in the automatic workload repository (AWR), in the form of snapshots. The information gathered is very precise. The ADDM is scheduled to run automatically by the **MMON background process**. Every time a snapshot is taken, the ADDM is triggered to do an analysis of the period corresponding to the last two snapshots.

The results generated by the analysis are stored in the AWR and is accessible through the Enterprise Manager Console.

It is possible to invoke the ADDM from the EM console or manually. This option gives you the ability to run an analysis across any two snapshots. By doing this, ADDM provides tuning recommendations on the user-defined time period.

The methodology used by ADDM is based on a wait and time statistic model. The model is helpful in determining where time is spent within the database. It uses a tree structure to represent all possible tuning issues. The root of the tree represents the symptoms and going down to the leaves helps identify the root causes. The ADDM walks down a tree using the time-based thresholds for each node. If the time-based threshold is not exceeded for a specific node, the ADDM prunes the corresponding subtree.

ADDM Output

The output generated by ADDM is best viewed using the Enterprise Manager Console. After an analysis, the ADDM reports its findings along with the impact on the database. The impact is measured in terms of time spent. The ADDM findings include symptoms and root causes. A number or recommendations are also generated. Recommendations generated by ADDM include:

- Hardware changes
- Database configuration
- Schema changes
- Application changes
- Using other

To get a more detailed analysis/recommendation about a specific problem, a call could be made to advisors, such as the segment or undo advisor. The ADDM output is stored in the Automatic Workload Repository for future reference.

ADDM analysis results are represented as FINDINGS. An example of ADDM analysis results is shown below:

```
FINDING 1: 31% impact (7798 seconds)
-----------------------------------
SQL statements were not shared due to the usage of literals. This
resulted in
additional hard parses which were consuming significant database time.

RECOMMENDATION 1:
Application Analysis, 31% benefit (7798 seconds)
  ACTION: Investigate application logic for possible use of bind
variables
    instead of literals. Alternatively, you may set the parameter
```

```
    "cursor_sharing" to "force".
  RATIONALE: SQL statements with PLAN_HASH_VALUE 3106087033 were found to
be
    using literals. Look in V$SQL for examples of such SQL statements.
```

Details of the latest ADDM run can be obtained from the EM console. First select the **Database Home page** -> **Diagnostic Summary region** -> **Performance Findings Link** -> **ADDM Findings Page.** Specific areas of interest on this page are the **View Report** Button. This gives you details of the performance analysis in the form of a text report. By clicking on a specific issue, you are directed to **Performance Finding Details page**.

Data Dictionary Views containing ADDM information

Data Dictionary View	Description
DBA_ADVISOR_TASKS	This view provides basic information about existing tasks, such as the task Id, task name, and when created.
DBA_ADVISOR_LOG	This view contains the current task information, such as status, progress, error messages, and execution times.
DBA_ADVISOR_RECOMMENDATIONS	This view displays the results of completed diagnostic tasks with recommendations for the problems identified in each run.
DBA_ADVISOR_FINDINGS	This view displays all the findings and symptoms that the diagnostic monitor encountered along with the specific recommendation.

EXAM OBJECTIVE: AUTOMATIC SHARED MEMORY MANAGEMENT

In Oracle 10g, the memory components are collectively known as the System Global area. Some of the memory components also referred to as pools need to have their size configured properly. The main pools are the shared pool, the large pool, the database cache size and the Java Pool. The appropriate individual sizes need to configured based on the overall SGA size. In Oracle 10g, all you need to specify is the total amount of memory to be used by all SGA components. The database uses this value to redistribute memory between the various components based on the current workload of the database.

- The new initialization parameter (SGA_TARGET) can include all the memory in the SGA, manually and automatically seized components and any internal allocations during startup.

- The initialization parameters used in Oracle 9i and earlier namely DB_CACHE_SIZE, LARGE_POOL_SIZE, SHARED_POOL_SIZE and JAVA_POOL_SIZE are now considered auto-tuned SGA parameters.

- The Automatic Shared Memory Management feature uses a background process called Memory Manager (MMAN). This background process is responsible for coordinating the sizes of the various memory components dynamically based on the current workload.

GUIDELINES

- Automatic Shared memory management is configured by setting the SGA_TARGET initialization parameter. If the parameter is set to the default value of 0, then automatic shared memory management is disabled. If set to a non-zero value then Oracle automatically sizes of the shared pool, the large pool, the java pool and the Database buffer cache accordingly.
- Within the SGA the buffers whose sizes can be configured manually are the Log Buffer (LOG_BUFFER), Keep and Recycle Pools (DB_KEEP_CACHE_SIZE and DB_RECYCLE_CACHE_SIZE), A streams pool (STREAMS_POOL_SIZE) which is new Oracle 10g , fixed SGA and other internal allocations and caches to accommodate non-standard tablespace blocks (DB_*n*k_CACHE_SIZE, *n*=2,4,8,16,32).

- The initialization parameter STATISTICS_LEVEL must be set to TYPICAL or ALL.
- The SGA_TARGET should be less than the SGA_MAX_SIZE initialization parameter. It can be reduced upto a minimum value defined for an individual auto-tuned component. Any change you make to the SGA_TARGET only affects the auto-tuned initialization parameters.
- You can disable automatic shared memory tuning by setting the SGA_TARGET to value of 0.

CONFIGURING AUTOMATIC SHARED MEMORY MANAGEMENT.

You can configure Automatic Shared Memory Management either:

- Manually
- Using Enterprise Manager Console

- The Enterprise Manager can be used to configure Automatic Shared Memory Management. To do so, select the **Database -> Administration Tab ->**

Memory Parameters -> SGA Tab -> Enable button. Specify the total SGA size in Megabytes.

- The ALTER SYSTEM command may be used to configure Automatic Shared Memory Management manually.

SQL> ALTER SYSTEM SET SGA_TARGET = 500M SCOPE=spfile;

Also modify the initialization parameter listed below to zero, or completely remove them from the parameter file.
SHARED_POOL_SIZE=0
LARGE_POOL_SIZE=0
JAVA_POOL_SIZE=0
DB_CACHE_SIZE=0

EXAM OBJECTIVE: USING AUTOMATIC OPTIMIZER STATISTICS COLLECTION

In earlier versions of Oracle, the task of gathering statistics for various objects was the database administrator's responsibility. Absent or stale statistics could result in in-efficient execution plans being generated. The Automatic Optimizer Statistics Collection feature automates this task and relieves the DBA of this task.

GUIDELINES

- To ensure that statistics will be gathered, you would need to set the parameter STATISTICS_LEVEL to either TYPICAL or ALL. When this is done, a job is started at instance startup to automatically collect statistics. If the STATISTICS_LEVEL parameter is set to BASIC, then statistics will be gathered only for certain objects, and all objects that have stale statistics will be skipped.

- In Oracle 10g, there may be certain tables or schemas that you don't want to subject to the automatic statistics gathering feature.
  SQL> EXECUTE DBMS_STATS.LOCK_TABLE_STATS('schema','tablename');
  SQL> EXECUTE DBMS_STATS.LOCK_SCHEMA_STATS('schema');

- You can however keep in mind that it is possible to overwrite statistics even if they are locked by setting the value of the FORCE argument to TRUE.

- If current statistics are resulting in a sub-optimal execution plan it is possible to revert back to previously gathered statistics. This is possible only if the statistics were gathered automatically and not by a DBA issuing an ANALYZE command.

- The procedure RESTORE_TABLE_STATS can be used to restore statistics of a tables as of a specified timestamp.
- To restore statistics of all tables of a schema as of a specified timestamp you can use the RESTORE_SCHEMA_STATS procedure.
- To restore the statistics of all the tables of the database as of a specified timestamp you can use the RESTORE_DATABASE_STATS procedure.
- To restore statistics of all fixed tables as of a specified timestamp you can use the RESTORE_FIXED_OBJECTS_STATS procedure. To do the action you need to have either the SYSDBA role or the ANALYZE ANY DICTIONARY system privilege.
- To restore statistics on all dictionary tables as of a specified timestamp you can use the RESTORE_DICTIONARY_STATS procedure. To do the action you need to have either the SYSDBA role or the ANALYZE ANY DICTIONARY system privilege.
- To restore system statistics as of a specified timestamp you can use the RESTORE_SYSTEM_STATS procedure.

EXAM OBJECTIVE: AUTOMATIC UNDO RETENTION TUNING

In Oracle 9i, in order to simplify the management of undo segments, there were some major changes introduced. The feature was known as the Automatic Undo Management feature. In this feature the Oracle Database was responsible for allocating and managing the undo space among various active sessions.

The two initialization parameters that needed to be configured for automatic undo management were the UNDO_TABLESPACE and the UNDO_RETENTION. In this feature all transactions share a single undo tablespace.

ENHANCEMENTS TO UNDO MANAGEMENT

In Oracle 10g tuning of undo retention is done automatically by collecting database usage statistics and estimating undo capacity needs for the successful completion of queries. You can set a low threshold value for the UNDO_RETENTION parameter so that the system retains the undo for at least the time specified in the parameter, provided that the current undo table space has enough space. Under space constraint conditions, the system may retain undo for a shorter duration than that specified by the low threshold value in order to allow DML operations to succeed.

The RETENTION GUARANTEE clause of the CREATE UNDO TABLESPACE and CREATE DATABASE statements ensures that undo information is not overwritten. This option must be used with caution, because it can cost DML operations to fail if the undo table space is not big enough.

In Oracle 10g, the optimal value for undo retention is automatically determined based on the size of the undo tablespace. The Oracle 10g database can dynamically adjust itself to change undo requirements depending on the system activity.

Further, with the new proactive monitoring infrastructure of Oracle, you are warned or any current or impending problems, such as 'SNAPSHOT TOO OLD' errors and the undo tablespace running out of space.

The undo advisor feature helps to size the undo tablespace appropriately based on workload history.

Lastly, some of the undo related initialization parameters have been eliminated, since they are being calculated automatically.

- MAX_ROLLBACK_SEGMENTS (takes a default value)
- UNDO_SUPPRESS_ERRORS (always TRUE)
- ROW_LOCKING (eliminated)
- SERIALIZABLE (eliminated)
- TRANSACTION_AUDITING (always TRUE)

SETTING THE UNDO_RETENTION INITIALIZATION PARAMETER

The default value for the UNDO_RETENTION parameter is 900. Retention is specified in units of seconds. The parameter determines the low threshold value of undo retention. The system retains the undo for at least the time specified in this parameter. The setting for this parameter should account for any flashback requirements of the system.

SQL> ALTER SYSTEM SET UNDO_RETENTION=2400;

UNDO RETENTION TUNING

The proactive undo retention tuning feature is possible since Oracle 10g collects query duration information every 30 minutes. The undo retention is then tuned for the longest-running query accordingly. There is no need to manually set the UNDO_RETENTION initialization parameter. To proactively monitor the undo tablespace you can set out-of-space alerts to warn you when the Oracle is running out of space in the undo tablespace. You can set warning and critical threshold values.

Automatic Undo retention can be enabled from the EM console. Once you have created a tablespace to hold the undo data, enable automatic undo retention by selecting **Database -> Instance section -> Undo Management -> Automatic Undo retention (Enable button).**