Using Perceptive Content with an Oracle Database

Best Practices

Version: 7.3.x

Written by: Product Knowledge, R&D
Date: June 2019

Hyland

Table of Contents

About the Oracle server

Perceptive Content works with several different databases, including Oracle. Oracle presents unique challenges due to the vast configuration possibilities that exist. This document provides basic guidelines for configuring your Oracle database based on industry-wide best practices, testing, and client feedback. You can also refer to the “Quick Reference Guide” at the end of this document, which provides the information in a checklist format.

Reference this guide while you are creating your Oracle database. For more information about creating your database, refer to the *Perceptive Content Installation and Setup Guide*.

Getting started

The following sections contain information you should consider when creating or configuring an Oracle database.

- Oracle instance
- Host configuration
- Version and patches

Oracle instance

Every Oracle database is associated with an Oracle instance. When you create a database on the server, Oracle allocates an area in memory called the System Global Area (SGA) and starts one or more Oracle processes. This combination of the SGA and the Oracle processes is an Oracle instance. The memory and processes of an instance manage the associated database's data and serve the users of the database.

While not required, we recommend that your Perceptive Content instance be dedicated to serving the Perceptive Content database and application only. This dedication provides you with the flexibility to make specific configuration changes as necessary for the Perceptive Content instance or database without having to factor in other environments.

Host configuration

A number of different operating systems can host the Oracle database including Windows, AIX, HP/UX, Solaris, and Linux. To ensure adequate support for your environments, use a platform that your technical staff can most effectively support based on their experience as well as operating costs.

Version and patches

Check the *Perceptive Content Technical Specifications* for the required database version. For product technical specifications and system requirements, refer to the *Technical Specifications* document available in the Customer Portal on the Perceptive Software website. Refer to these specifications before updating your database version as well.

Depending on the release of Oracle, you might need to apply patches to address any bugs that affect your specific environment. After applying patches to your environments, ensure the bug is resolved and no new issues arise.

Set up

During the creation or configuration of your Oracle database, perform the tasks in the following sections.

- Oracle parameters
- System Global Area
- Storage
- INUSER schema
- Tablespace
- Undo/Rollback segment management
- Redo log
- Control files
- Archiving redo logs
- Change default passwords

Oracle parameters

The following parameters ensure that Perceptive Content performs as intended and helps prevent configuration issues. Each connection that the application makes to the database also uses these parameters by altering the session after it is connected. You can view your current parameter settings with the `SHOW PARAMETER [parameter_name]; command: Parameter_name` is optional. If you do not specify a parameter, all parameters appear.

```
NLS_DATE_FORMAT = 'RRRR-MM-DD'
NLS_TIME_FORMAT = 'HH24:MI:SS.FF'
NLS_TIMESTAMP_FORMAT = 'RRRR-MM-DD HH24:MI:SS.FF'
NLS_COMP=LINGUISTIC
NLS_SORT=BINARY_CI
OPTIMIZER_MODE = ALL_ROWS
QUERY_REWRITE_INTEGRITY = TRUSTED
QUERY_REWRITE_ENABLED = TRUE
CURSOR_SHARING = EXACT
OPTIMIZER_INDEX_COST_ADJ=1
```

System Global Area

The System Global Area (SGA) is a group of shared memory structures that contains data and control information for the Oracle database. When an Oracle database instance starts, it allocates an SGA. The following chart represents a starting point for your Oracle SGA.

Value	Non-Dedicated	Dedicated
Total memory	1042 MB	70% of total RAM (for example, 700 MB for 1GB RAM)
Shared pool	256 MB	25% of SGA
Buffer cache	512 MB	50% of SGA
Large pool	32 MB	25% of SGA
Java pool	32 MB	32 MB
PGA aggregate target	200 MB	20% of SGA

Adjust these values accordingly based on your environment and usage of the database and application. You can also use Oracle Enterprise Manager's (OEM) Advisor Central to check the SGA Size Advice for your database. We recommend that you enable Automatic Shared Memory Management to let Oracle determine the best way to allocate the SGA between each component. If you have available memory, consider allocating additional memory by setting the Maximum SGA Size parameter. The Maximum SGA Size specifies the maximum memory that the database can allocate. Specifying the Maximum SGA Size adds flexibility to increase the Total SGA Size up to the Maximum SGA Size if needed.

Because the SGA stores data in memory for fast access, the SGA should be within the main memory. If you swap pages of the SGA to disk, then the data is no longer quickly accessible. On most operating systems, the disadvantage of paging significantly outweighs the advantage of a large SGA.

Paging occurs when an operating system transfers memory-resident pages to disk solely to allow new pages to be loaded into memory. Many operating systems page to accommodate large amounts of information that do not fit into real memory. On most operating systems, paging reduces performance.

Use your operating system utilities to examine the operating system, to identify whether there is a lot of paging on your system. If there is significant paging, then the total memory on the system might not be large enough to hold everything for which you have allocated memory. Either increase the total memory on your system, or decrease the amount of memory allocated.

Storage

Oracle recommends the SAME approach (Stripe and Mirror Everywhere). Raid 1+0 is recommended only for the database file. Raid 1 is recommended for the others. The following table provides Oracle's recommended RAID use with each type of Oracle database file.

RAID	Type of RAID	Control File	Database File	Redo Log File	Archive Log File
0	Striping	Avoid	OK	Avoid	Avoid
1	Shadowing	Best	OK	Best	Best
1+0	Striping and Shadowing	OK	Best	Avoid	Avoid
3	Striping with static parity	OK	OK	Avoid	Avoid
5	Striping with rotating parity	OK	Best if RAID 0-1 are not available	Avoid	Avoid

INUSER schema

The INUSER schema/user owns the Perceptive Content database objects. The schema consists of tables, indexes, and constraints. Most of the columns are defined as VARCHAR2, TIMESTAMP(6) and NUMBER data types. The IN_SCRIPT.DATA column, which is a CLOB, is the only column in the schema that uses a large object data type.

Determine the Perceptive Content database schema version by checking the PRODUCT_VERSION field of the most recent record in the IN_PRODUCT_MOD_HIST table by looking at the top row returned by the following query:

```
SELECT * FROM IN_PRODUCT_MOD_HIST ORDER BY MOD_TIME DESC;
```

Tables

By default, tables reside in the DATA tablespace and are owned by the INUSER account. You can move individual tables into different tablespaces as necessary for general maintenance, to help offload I/O between tablespaces, or to make use of different block sizes for tables.

Indexes

By default, indexes reside in the INDX tablespace and are owned by the INUSER account. To improve performance in case insensitive searches, functional indexes exist on certain text fields using the NLSSORT function along with BINARY_CI. You can move individual indexes into different tablespaces as necessary for general maintenance, to help offload I/O between tablespaces or to make use of different block sizes for tables.

Constraints

The Perceptive Content database uses all the standard constraint types to enforce uniqueness and referential integrity including Primary Keys, Foreign Keys, Unique, and Not Null check constraints. Certain columns, such as date fields, are populated with default values.

Tablespace

The installation instructions set up two tablespaces: DATA and INDX. Depending on how you use the application, you may need to create additional tablespaces for certain tables. To help distribute I/O across multiple drives or luns, move indexes into their own tablespace.

TEMP tablespace

By default, TEMP tablespaces are locally managed and use tempfiles. We recommend that you use an LMT TEMP tablespace with tempfiles.

Datafiles

You can designate tablespaces as either BIGFILE or SMALLFILE tablespaces. Use SMALLFILE tablespaces to provide you with the flexibility to create multiple datafiles for a single tablespace. BIGFILE tablespaces can contain only one file, whereas traditional SMALLFILE tablespaces can contain up to 1,022 files.

Undo/Rollback segment management

While you can use the Undo Advisor to determine what Oracle recommends for the undo_retention parameter for your environment, we recommend that you use the following undo management settings and parameters as a starting point.

Name	Value
undo_management	AUTO
undo_retention	9000
undo_tablespace	<Undo Tablespace Name>

As you increase the undo_retention parameter, you must also increase the size of the undo tablespace to allow for the higher retention. You need to adjust these parameters appropriately in the event you encounter ORA-1555 errors or other errors related to rollback segments or undo retention.

Redo logs

We recommend that you maintain at least three redo log groups with two members per group. Make sure each member of a redo log group is located on different physical disks for effective multiplexing.

The size of your redo logs depends on your volume. If you have a very high rate of DML activity (inserts, updates, and deletes), increase the size of your redo log files to prevent them from switching too often. This increase prevents the need for additional overhead, which ultimately degrades performance. Similarly, do not create large redo logs because they do not switch often enough during peak hours. If your database crashes or you lose all members of an active redo log group, large redo logs increase the potential for higher data loss. It is recommended that you create your redo logs to approximately 256MB or 512MB so that the log switches about every 15 minutes during peak hours.

Control files

Ensure your database has at least two control files located on separate disks. If a control file fails, you can restore it using the intact copy of the control file from the other disk. Specify the disk locations in the database's initialization parameter file. Control files store the status of the database physical structure, which is crucial to database operation. Make sure you back up your control files during your scheduled backups and after any configuration changes such as changes to control files, redo log, or datafiles.

Archiving redo logs

If you are running Oracle in archive log mode, verify the archive destination is large enough to contain about five days' worth of normal archivelog data. Maintaining this space helps to ensure the destination does not fill up and cause the database to hang. Ensure the archive destination is on a different drive, and do not make the archive destination a backup drive or a drive used for datafiles.

Change default passwords

Change the default passwords for all the user accounts that get created during installation including SYS, SYSTEM, SYSMAN, DBSNMP, and OUTLN to help prevent intrusion. When defining passwords, be sure to define complex passwords containing mixed cases, letters, numbers, and special characters.

Tuning

The following sections define how to configure your database after creating it and how to perform basic setup steps.

- Oracle parameters
- Block sizes

INI_TRANS

By default, Oracle sets INI_TRANS to 1 for tables and 2 for indexes. In some cases, you may benefit by increasing this parameter to a higher number. Set this option to 5 or so as a good starting point. The maximum setting for this parameter is 255, assuming there is enough free space in the block to accommodate the transaction. Each transaction uses 24 bytes. You can modify a table and optionally rebuild it to increase INI_TRANS.

PCT_FREE

You can increase the PCT_FREE parameter on certain tables to help improve performance in various ways. For example, you can increase PCT_FREE in order to reserve a percentage of each block so that rows can grow as needed due to updates, so that row chaining is less likely to occur. Increasing the amount of free space also allows for more ITL's (Interested Transaction List) slots in each block. Increasing PCT_FREE along with smaller block sizes can further reduce the number of rows stored in a single block thus potentially reducing block level contention.

Note Setting the PCT_FREE parameter too high results in wasted block space.

Block sizes

The default Oracle block size is 8K. If you are experiencing excessive row lock contention or block contention on the IN_OSM_TREE, IN_WF_QUEUE and IN_LOCK tables, all of which are small tables with small rows sizes, you can consider moving those tables and their respective indexes into 2K tablespaces. Moving the tables and indexes may help to reduce the level of contention on those tables.

Maintenance

The following sections define how to maintain your database after creating it.

- Index rebuilds
- Backup
- Statistics
- Automatic Database Diagnostics Monitor (ADDM) Reporting

Index rebuilds

We recommend that you rebuild indexes regularly on dynamic tables because they can become fragmented over time due to inserts, updates, and deletes (DML). You can run the following command to check if an index is a good candidate for re-indexing. Running the analyze command with the validate structure clause puts an exclusive lock on the table so run this command during off-peak hours to minimize the impact to users.

```
analyze index inuser.<Index_Name> validate structure;

select name as IndexName,
       height,
       lf_rows,
       del_lf_rows,
       DEL_LF_ROWS_LEN,
       USED_SPACE,
       PCT_USED
   from index_stats;
```

After you run the command and gather the data, we recommend rebuilding an index if it meets the following criteria.

- Index levels (height) is greater than 3
- PCT_USED is less than 75%.
- Rows deleted are greater than 20% (space is not automatically reused).
- Index is unclustered and performance is degrading (causing increases in number of blocks to be read).

Backup

Backing up your data regularly is one of the most important tasks of database administration. We recommend that you perform daily exports of your database. However, for maximum recoverability we recommend that you configure your database to run in archivelog mode and then schedule daily online backups using RMAN to disk or tape. If you backup to disk, you should have a third party media management software to transfer your RMAN backups to tape.

As with any backup solution, perform periodic tests to ensure you can restore and recover your database to a predefined point in time. This process helps you prepare for recovery from various situations. If you decide to use RMAN, you can use the control file to keep track of backups or you can create another database instance to store your RMAN catalog and repository. To ensure maximum recoverability, if you have the resources, create a second database for RMAN on a different physical server.

Statistics

There are two types of automatic statistics collected in Oracle by default. Statistics stored in the Automatic Workload Repository (AWR) and statistics collected for the Cost-Based Optimizer (CBO). The following sections provide a high-level explanation of each type of statistic and provide some basic recommendations and examples.

Automatic Workload Repository (AWR)

AWR consists of operational statistics serving diagnostic, self-tuning and database tuning purposes. AWR statistics reflect the state of the whole database and various areas within the database during specific periods. It includes information about wait events, latches, enqueues, CPU consumption, SGA components, PGA, various statistics and metrics, most expensive SQLs by different measures, and so on. AWR automatically collects these statistics and you can access them in AWR reports.

By default, the AWR takes a snapshot of performance related statistics every 60 minutes and retains the statistics for seven days. The default and recommended Collection Level is "TYPICAL" to ensure statistics are collected for all objects that don't have statistics as well as any objects whose statistics are considered stale due to 10% of the table or index changing due to DML (inserts, update, and deletes).

Modifying AWR Snapshot Settings

You can adjust the interval, retention, and captured Top SQL of snapshot generated for a specified database ID, but these adjustments can affect the precision of the Oracle diagnostic tools. The INTERVAL setting affects how often in minutes that snapshots are automatically generated. The RETENTION setting affects how long in minutes that snapshots are stored in the workload repository.

The TOPNSQL setting affects the number of Top SQL to flush for each SQL criteria (Elapsed Time, CPU Time, Parse Calls, Shareable Memory, and Version Count). The value for this setting is not affected by the statistics or flush level and overrides the system default behavior for the AWR SQL collection. It is possible to set this setting to MAXIMUM to capture the complete set of SQL in the cursor cache. However, setting it to MAXIMUM or a high value may lead to possible space and performance issues since there will be more data to collect and store. To adjust the settings, use the MODIFY_SNAPSHOT_SETTINGS procedure as shown in the following example:

```
BEGIN
  DBMS_WORKLOAD_REPOSITORY.MODIFY_SNAPSHOT_SETTINGS( RETENTION => 43200,
                 INTERVAL => 60, TOPNSQL => 100, DBID => 4249447005);
END;
/
```

Note STATSPACK also provides this type of statistic but it does not store statistical data in the AWR structures. You should not run STATSPACK if you are gathering AWR statistical data due to the potential of performance degradation when both are running. If necessary, you can uninstall Statspack by running the spdrop.sql script.

Optimizer statistics (for Cost Based Optimizer)

The CBO maintains statistics about various database objects such as tables and indexes in both the application schemas and Oracle internal schemas (SYS, SYSTEM, and so on). These statistics include the number of rows in each table, number of distinct values in columns (densities), average lengths of columns, distribution of values within a column (histograms), and so on.

The Oracle Cost-Based optimizer uses these statistics for calculating the optimal execution plans for a query. The query and the configuration parameters constitute the input for the optimizer to determine the best execution plan for each SQL statement. The default for GATHER_STATS_JOB is responsible for collecting optimizer statistics. If this job is disabled for any reason, use other user defined DBMS_STATS scripts instead. You can use the following command to disable the default statistics job if you are planning to use custom scripts:

```
Connect as SYS and EXEC DBMS_SCHEDULER.DISABLE('GATHER_STATS_JOB');
```

The goal when gathering statistics is to generate them with as much statistical accuracy as possible, therefore 100% sample sizes are suggested because any reduction in sample size affects accuracy. Note that 100% sample sizes are potentially time consuming and you need to plan carefully to fit the statistics gathering activities within the existing maintenance window. If it becomes necessary to reduce the percentage, try using an estimate of at least 30%. Gathering new optimizer statistics should maintain or improve existing execution plans, but some query performance may degrade. By default, maintain statistics for the last 30 days.

Make sure that all objects (tables and indexes) have gathered statistics. An easy way to achieve this is to use the CASCADE parameter, and user the METHOD_OPT parameter to ensure that any columns with skewed data distribution will collect histograms at sufficient resolution. Oracle recommends a conservative and planned approach of adding a histogram only if it is required rather than collecting column statistics on all columns. Using the default column statistics setting of AUTO means that DBMS_STATS decide which columns to add histograms to produce a better plan. If statistics are not completely up to date, then the presence of histograms can cause trouble when parsing values that are out of range, between values for "frequency" histograms. In these circumstances, the optimizer can result in inaccuracies.

In earlier versions, the default setting for the METHOD_OPT parameter was "FOR ALL COLUMNS SIZE 1" which collects only a high and a low value and effectively meant that there were no detailed column statistics. In some cases, the effect of a histogram is adverse to the generation of a better plan so you might set this parameter to "FOR ALL COLUMNS SIZE 1" and later adjust to “FOR ALL COLUMNS SIZE AUTO” for cases where column data is known to be skewed and column statistics are known to be beneficial.

Gather statistics for your entire database at periodic intervals (weekly or monthly depending on how fast your data changes). We recommend that you collect deep statistics initially using a 100% sample size if possible and then gather statistics on objects as they become stale or after significant changes are made.

Example Commands for Gathering Statistics

The following examples use a 100% sample size and include indexes and basic histogram analysis. You can adjust these as necessary for your environment. You may have to reduce the sample size if your database is very large or you could run out of temp space or exceed your maintenance window.

Gathering statistics for an individual table:

```
EXEC DBMS_STATS.GATHER_TABLE_STATS (OWNNAME => 'INUSER', TABNAME => 'IN_ALARM',
ESTIMATE_PERCENT => 100,  CASCADE => TRUE,  METHOD_OPT => 'FOR ALL COLUMNS SIZE 1');
```

Gathering statistics for all objects in a schema:

```
EXEC DBMS_STATS.GATHER_SCHEMA_STATS (OWNNAME => 'INUSER', CASCADE =>TRUE,
ESTIMATE_PERCENT => 100, METHOD_OPT => 'FOR ALL COLUMNS SIZE 1');
```

Gathering statistics for all objects in the database:

```
EXEC DBMS_STATS.GATHER_DATABASE_STATS (ESTIMATE_PERCENT => 100, CASCADE => TRUE,
METHOD_OPT => 'FOR ALL COLUMNS SIZE 1');
```

System statistics

System statistics allow the optimizer to consider a system’s I/O and CPU performance and utilization. For each candidate plan, the CBO optimizer computes estimates for I/O and CPU costs. It is important to know the system characteristics to pick the most efficient plan with the optimal proportion between I/O and CPU cost. System CPU and I/O characteristics depend on many factors and do not stay constant all the time. The gathered statistics are:

- Single block readtime in ms.
- Multiblock readtime in ms.
- CPU speed in mhz.
- Average multiblock_read_count in number of blocks.

System statistics help the optimizer make decisions based on cost in proportion to elapsed time. This provides the optimizer with information to better judge if a full table scan or an index access is more appropriate. You should need to gather system statistics initially and then again after any significant changes to the system. Statistics you gather initially and after system changes reflect the baseline. You should also capture statistics during an interval of time when the system has the most common workload.

Unlike when table, index, or column statistics are updated, the Oracle server does not invalidate already-parsed SQL statements when system statistics get updated. Only the new parsed SQL statements use the newly activated statistics. Use the following example to gather system statistics.

```
execute dbms_stats.gather_system_stats('Start');
```

After an adequate amount of time based on workload, use the following example to stop gathering system statistics.

```
execute dbms_stats.gather_system_stats('Stop');
```

To see the statistics, query a table (owned by the SYS schema) called aux_stats$. After gathering system statistics, this table contains a few critical numbers used by the new optimizer algorithms to calculate costs. Note that you need to flush the shared_pool to invalidate existing execution plans. The following query provides the current settings.

```
Select pname, pval1
from sys.aux_stats$
where sname = 'SYSSTATS_MAIN';
The exact list of results depends on your version of Oracle, but the following example
provides typical results:
PNAME                 PVAL1
--------------------------
CPUSPEEDNW      2121.72855
IOSEEKTIM               10
IOTFRSPEED            4096
SREADTIM             2.235
MREADTIM             4.339
CPUSPEED              2134
MBRC                    13
MAXTHR             8578048
SLAVETHR
```

Automatic Database Diagnostics Monitor (ADDM) Reporting

Along with the AWR reports you can also use the ADDM reports to analyze a snapshot range. These reports are a good place to start for getting tips to tune your database. Run the ADDM report (addmrpt.sql) from within OEM or call the report directly from the $ORACLE_HOME/rdbms/admin directory. We recommend the OEM version of the reports because they are more detailed.

Quick Reference Guide

The following checklist provides a quick reference for your Oracle database.

✓	**Task**
	Check the *Perceptive Content Technical Specifications* for your release of Perceptive Content for the required database version.
	Apply any Oracle patches recommended by the *Perceptive Content Technical Specifications*.
	Set total SGA memory size to 1024 MB for non-dedicated or 70% of total RAM for dedicated.
	Set shared pool size to 256 MB for non-dedicated or 25% of SGA for dedicated.
	Set buffer cache size 512 MB for non-dedicated or 50% of SGA for dedicated.
	Set large pool size to 32 MB for non-dedicated or 25% of SGA for dedicated.
	Set Java pool size to 32 MB for dedicated and non-dedicated
	Set PGA aggregate target to 210 MB for dedicated or 20% of SGA for non-dedicated.
	Gather initial server statistics and then disable server statistics gathering.
	If you have performed any significant configuration changes to the database, hosting environment or storage (such as a SAN), gather statistics again.
	Run schema statistics once a week at a minimum of 30%.
	Set up daily data exports of your database.
	Configure the database to run in archivelog mode.
	Schedule daily online backups using RMAN to disk or tape. If you backup to disk, use a third-party media management program to transfer RMAN backups to tape.
	Perform periodic restore and recovery tests.
	Set up at least 2 redo log members and groups with each member on different disks
	Set redo logs to switch about every 15 minutes during peak hours.
	Change the default passwords, if necessary.
	Rebuild indexes periodically.
	Review AWR and ADDM reports periodically.