1

Performance Tuning Overview

This chapter provides an introduction to performance tuning and contains the following sections:

- Introduction to Performance Tuning
- Introduction to Performance Tuning Features and Tools

Introduction to Performance Tuning

This guide provides information about tuning Oracle Database for performance. Topics discussed in this guide include:

- Performance Planning
- Instance Tuning
- SQL Tuning

> **See Also:**
>
> *Oracle Database 2 Day + Performance Tuning Guide* to learn how to use Oracle Enterprise Manager Cloud Control (Cloud Control) to tune database performance

Performance Planning

Refer to the topic Database Performance Fundamentals before proceeding with the other parts of this documentation. Based on years of designing and performance experience, Oracle has designed a performance methodology. This topic describes activities that can dramatically improve system performance, such as:

- Understanding Investment Options
- Understanding Scalability
- System Architecture
- Application Design Principles
- Workload Testing, Modeling, and Implementation
- Deploying New Applications

Instance Tuning

Diagnosing and Tuning Database Performance discusses the factors involved in the tuning and optimizing of an Oracle database instance.

When considering instance tuning, take care in the initial design of the database to avoid bottlenecks that could lead to performance problems. In addition, you must consider:

- Allocating memory to database structures
- Determining I/O requirements of different parts of the database
- Tuning the operating system for optimal performance of the database

After the database instance has been installed and configured, you must monitor the database as it is running to check for performance-related problems.

Performance Principles

Performance tuning requires a different, although related, method to the initial configuration of a system. Configuring a system involves allocating resources in an ordered manner so that the initial system configuration is functional.

Tuning is driven by identifying the most significant bottleneck and making the appropriate changes to reduce or eliminate the effect of that bottleneck. Usually, tuning is performed reactively, either while the system is in preproduction or after it is live.

Baselines

The most effective way to tune is to have an established performance baseline that you can use for comparison if a performance issue arises. Most database administrators (DBAs) know their system well and can easily identify peak usage periods. For example, the peak periods could be between 10.00am and 12.00pm and also between 1.30pm and 3.00pm. This could include a batch window of 12.00am midnight to 6am.

It is important to identify these peak periods at the site and install a monitoring tool that gathers performance data for those high-load times. Optimally, data gathering should be configured from when the application is in its initial trial phase during the QA cycle. Otherwise, this should be configured when the system is first in production.

Ideally, baseline data gathered should include the following:

- Application statistics (transaction volumes, response time)
- Database statistics
- Operating system statistics
- Disk I/O statistics
- Network statistics

In the Automatic Workload Repository, baselines are identified by a range of snapshots that are preserved for future comparisons. See "Automatic Workload Repository".

The Symptoms and the Problems

A common pitfall in performance tuning is to mistake the symptoms of a problem for the actual problem itself. It is important to recognize that many performance statistics indicate the symptoms, and that identifying the symptom is not sufficient data to implement a remedy. For example:

- Slow physical I/O

  Generally, this is caused by poorly-configured disks. However, it could also be caused by a significant amount of unnecessary physical I/O on those disks issued by poorly-tuned SQL.
- Latch contention

Rarely is latch contention tunable by reconfiguring the instance. Rather, latch contention usually is resolved through application changes.

- Excessive CPU usage

  Excessive CPU usage usually means that there is little idle CPU on the system. This could be caused by an inadequately-sized system, by untuned SQL statements, or by inefficient application programs.

When to Tune

There are two distinct types of tuning:

- Proactive Monitoring
- Bottleneck Elimination

Proactive Monitoring

Proactive monitoring usually occurs on a regularly scheduled interval, where several performance statistics are examined to identify whether the system behavior and resource usage has changed. Proactive monitoring can also be considered as proactive tuning.

Usually, monitoring does not result in configuration changes to the system, unless the monitoring exposes a serious problem that is developing. In some situations, experienced performance engineers can identify potential problems through statistics alone, although accompanying performance degradation is usual.

Experimenting with or tweaking a system when there is no apparent performance degradation as a proactive action can be a dangerous activity, resulting in unnecessary performance drops. Tweaking a system should be considered reactive tuning, and the steps for reactive tuning should be followed.

Monitoring is usually part of a larger capacity planning exercise, where resource consumption is examined to see changes in the way the application is being used, and the way the application is using the database and host resources.

Bottleneck Elimination

Tuning usually implies fixing a performance problem. However, tuning should be part of the life cycle of an application—through the analysis, design, coding, production, and maintenance stages. Often, the tuning phase is left until the database is in production. At this time, tuning becomes a reactive process, where the most important bottleneck is identified and fixed.

Usually, the purpose for tuning is to reduce resource consumption or to reduce the elapsed time for an operation to complete. Either way, the goal is to improve the effective use of a particular resource. In general, performance problems are caused by the overuse of a particular resource. The overused resource is the bottleneck in the system. There are several distinct phases in identifying the bottleneck and the potential fixes. These are discussed in the sections that follow.

Remember that the different forms of contention are symptoms that can be fixed by making changes in the following places:

- Changes in the application, or the way the application is used
- Changes in Oracle
- Changes in the host hardware configuration

Often, the most effective way of resolving a bottleneck is to change the application.

SQL Tuning

Many application programmers consider SQL a messaging language, because queries are issued and data is returned. However, client tools often generate inefficient SQL statements. Therefore, a good understanding of the database SQL processing engine is necessary for writing optimal SQL. This is especially true for high transaction processing systems.

Typically, SQL statements issued by online transaction processing (OLTP) applications operate on relatively few rows at a time. If an index can point to the exact rows that are required, then Oracle Database can construct an accurate plan to access those rows efficiently through the shortest possible path. In decision support system (DSS) environments, selectivity is less important, because they often access most of a table's rows. In such situations, full table scans are common, and indexes are not even used. This book is primarily focussed on OLTP applications.

See Also:

- *Oracle Database SQL Tuning Guide* for detailed information on the process of tuning and optimizing SQL statements
- *Oracle Database Data Warehousing Guide* for detailed information on decision support systems (DSS) and mixed environments

Query Optimizer and Execution Plans

When a SQL statement is executed on an Oracle database, the query optimizer determines the most efficient execution plan after considering many factors related to the objects referenced and the conditions specified in the query. This determination is an important step in the processing of any SQL statement and can greatly affect execution time.

During the evaluation process, the query optimizer reviews statistics gathered on the system to determine the best data access path and other considerations. You can override the execution plan of the query optimizer with hints inserted in SQL statement.

Introduction to Performance Tuning Features and Tools

Effective data collection and analysis is essential for identifying and correcting performance problems. Oracle Database provides several tools that allow a performance engineer to gather information regarding database performance. In addition to gathering data, Oracle Database provides tools to monitor performance, diagnose problems, and tune applications.

The Oracle Database gathering and monitoring features are mainly automatic, managed by Oracle background processes. To enable automatic statistics collection and automatic performance features, the `STATISTICS_LEVEL` initialization parameter must be set to `TYPICAL` or `ALL`. You can administer and display the output of the gathering and tuning tools with Oracle Enterprise Manager Cloud Control (Cloud Control), or with APIs and views. For ease of use and to take advantage of its numerous automated monitoring and diagnostic tools, Cloud Control is recommended.

See Also:

- *Oracle Database 2 Day DBA* to learn how to use Cloud Control to manage Oracle Database
- *Oracle Database 2 Day + Performance Tuning Guide* for a quick lesson on how to use Cloud Control to tune database performance, and Enterprise Manager Cloud Control 13c for details on everything Cloud Control has to offer.
- *Oracle Database PL/SQL Packages and Types Reference* for detailed information on the `DBMS_ADVISOR`, `DBMS_SQLTUNE`, `DBMS_AUTO_SQLTUNE`, and `DBMS_WORKLOAD_REPOSITORY` packages
- *Oracle Database Reference* for information about the `STATISTICS_LEVEL` initialization parameter

Automatic Performance Tuning Features

The Oracle Database automatic performance tuning features include:

- Automatic Workload Repository (AWR) collects, processes, and maintains performance statistics for problem detection and self-tuning purposes. See "Automatic Workload Repository".
- Automatic Database Diagnostic Monitor (ADDM) analyzes the information collected by AWR for possible performance problems with the Oracle database. See "Overview of the Automatic Database Diagnostic Monitor".
- SQL Tuning Advisor allows a quick and efficient technique for optimizing SQL statements without modifying any statements. See *Oracle Database SQL Tuning Guide*.
- SQL Access Advisor provides advice on materialized views, indexes, and materialized view logs. See *Oracle Database SQL Tuning Guide*.
- End to End Application tracing identifies excessive workloads on the system by specific user, service, or application component. See *Oracle Database SQL Tuning Guide*.
- Server-generated alerts automatically provide notifications when impending problems are detected. See *Oracle Database Administrator's Guide* to learn how to monitor the operation of the database with server-generated alerts.
- Additional advisors that can be launched from Oracle Enterprise Manager Cloud Control (Cloud Control), such as memory advisors to optimize memory for an instance. The memory advisors are commonly used when automatic memory management is not set up for the database. Other advisors are used to optimize mean time to recovery (MTTR), shrinking of segments, and undo tablespace settings. To learn about the advisors available with Cloud Control, see *Oracle Database 2 Day + Performance Tuning Guide*.
- The Database Performance page in Cloud Control displays host, instance service time, and throughput information for real time monitoring and diagnosis. The page can be set to refresh automatically in selected intervals or manually. To learn about the Database Performance page, see *Oracle Database 2 Day + Performance Tuning Guide*.

Additional Oracle Database Tools

This section describes additional Oracle Database tools that you can use for determining performance problems.

V$ Performance Views

The V$ views are the performance information sources used by all Oracle Database performance tuning tools. The V$ views are based on memory structures initialized at instance startup. The memory structures, and the views that represent them, are automatically maintained by Oracle Database for the life of the instance.

> **Note:**
>
> Oracle recommends using the Automatic Workload Repository to gather performance data. These tools have been designed to capture all of the data needed for performance analysis.

> **See Also:**
>
> - " Instance Tuning Using Performance Views" for more information about diagnosing database performance problems using the V$ performance views
> - *Oracle Database Reference* for more information about dynamic performance views