Oracle Core mechanisms of the database with Jonathan Lewis

Duration: 2 Days

What you will learn

This note outlines the content of a two-day seminar based on the contents of the book “Oracle Core” by Jonathan Lewis. The eight chapters of the book are reflected in the eight headings listed below – and the material presented in the course of the two days will follow this pattern.Unlike previous courses – which had a fixed timetable and were strictly set to four equal length sessions per day – this course will have a flexible timetable where the number and duration of pauses will depend on the amount of interaction from the attendees. There will, however, be a lunch break, and at least one morning and one afternoon break.

Related Training

Required Prerequisites

This seminar is not for beginners

To get the greatest benefit from the seminar you will need to have some idea of the basic concepts of undo, redo, and parsing

Course Objectives

Understand how data change, with undo and redo, work

Understand how transactions and read-consistency are implemented

Understand the implications of latch (and mutex) activity, and the effects of locking

Course Topics

Chapter 1: Overview
A brief introduction to the principal files used in an Oracle database
The key processes and what they do with those file
The way that Oracle uses memory

Chapter 2: Redo and Undo
The mechanisms that Oracle uses to change data, and reverse out change
Redo change vectors and undo records
How the redo and undo mechanisms allow Oracle to increase scalability and reduce blocking
Why we have redo log files but undo tablespaces
The three functions of undo

Chapter 3: Transactions and Consistency

Why writers don’t block readers
Internals of the Undo Segment – undo segment headers and the transaction table
Chaining undo records in undo blocks
Data segment headers and the Interested Transaction List
Commits and the options for cleanout
Two forms of Oracle error ORA-01555
The special handling for LOBs

Chapter 4: Latches, Mutexes and Locks
Why locks are different from latches and mutexes
Methods of using memory: arrays, pointers, hash tables and linked lists
Shared memory and the threat of lost updates
How latches and mutexes protect memory locations from concurrent overwrites
An example of the interaction of latches hash tables, and linked lists – the library cache
Locks as mechanisms for sharing and queueing
How queueing mechanisms lead to deadlocks
Internal structures used for locks. Some common lock types

Chapter 5: Thebuffer cache
Layers of memory structures
Granules and buffer pools, buffers and buffer headers
Working data sets and the database writer(s)
Buffer re-use and the LRU/Touch Count algorithm
Finding the right buffer – the cache buffers chains
Latching and pinning to protect shared memory
Reading data blocks into the buffer, and the creation and use of read consistent copies of blocks

Chapter 6: Writing and Recovery
The main write I/O activity
How DBWR and LGWR work individually, and how they co-operate
The various types of checkpoints and how they anticipate recovery scenarios and minimize the work needed to recover
Ancillary I/O activity – archiving, flashback logging, and change tracking – and how the range of recovery options affects

Chapter 7: Parsing and Optimising
Note particularly that this chapter is not about the arithmetic of the cost-based optimizer, it is about the various structures
How much does Oracle need to know to “understand” an SQL statement, and how is this information handled
What does the shared pool look like, and how does Oracle deal with loading and accessing all the pieces of information it
We start with memory granules to build the shared pool then consider the finer levels of granularity – the sub-pools and “
How does the dictionary cache work, how does the library cache work – and what is a cursor
What is the difference between a parent cursor and a child cursor
What is the difference between an “open” cursor, and a “pinned” cursor

Chapter 8: RAC variations
A very brief look at how Oracle addresses the issues raised when multiple machines are all trying to share the same data
How can we share objects safely when the “shared memory” is distributed across several machines
The global resource directory; master resources and shadow resources and their role in crash recovery
A couple of special cases to consider when designing for RAC