CDF/DOC/COMP_UPG/5413
September 23, 2004
Version 1.0

Raw Notes on 24×7 CDF Oracle Database Operations

Rick St. Denis[1]

[1]Glasgow University

Abstract

Oracle databases are complex to administer and operate. Tips and Techniques from "Oracle 24×7" are summarized here with comments on usage in CDF.

Contents

Chapter 1

Introduction

The CDF production Oracle databases reside on two machines, the online machine at B0 and the offline machine at FCC. These production machines and the Oracle databases must be available for the collection and analysis of the data. Ideally, they should never fail and the software should always work. However, this is not realistic, and in crude terms, achieving this requires inordinate expense in manpower and hardware.

Taking as step back, one asks why bother with such a complex system? The answer is to have a coherent set of constants for the operation and analysis of the experiment and to be able to go back to any point in time and understand what the conditions for a specific run or an analysis were.

The alternative of using text files to store constants and to swap them about might appear attractive at first, but becomes a mess as soon as several versions of the files exist. Trying to formalize this amounts to writing one's own database and, in effect, inventing yet another wheel. This path was abandoned in the 1980's along with polyester [2].

In order to understand how to keep the databases running, accessible and with valid data, points from the book "Oracle 24×7" [1] are summarized here with notes on actions that need to be taken within CDF in order to achieve this. This book describes what is necessary to ensure that the database is available 24 hours/day, 7 days/week. It is of course a worthy goal; however, it can only be achieved at great cost.

This note is to be viewed as a raw core dump of the information. Gathering this together and prioritizing and implementing the thoughts expressed here are the subjects of future notes. The present aim is to provide the basis for discussion of how to achieve smooth operation.

The sections follow that of the book and can be lumped together as follows:

- Understanding requirements of CDF and how to deal with emergencies,
- Configuration of the CPU's, Operating systems, Networks and the usage with Applications and Data,
- Setting up the databases, upgrading, migrating to new machines, backups, recovery, archiving, starting and stopping,
- Maintenance and Disk Space management,
- Troubleshooting,

- High Availability – getting from 30 days to 3 hours of downtime per year,
- The softwares toolkit to monitor.

Throughout the text, various notations are made to indicate followup action and to help in follow up notes. These actions are defined as:

ACTION:	action to be taken on this
IMPACTS API DESIGN:	impact API design
CHECK BOOK:	Look in book
DEVELOPMENT TASK:	interesting development program
ERROR RECOVERY POINT:	useful for error recovery consideration
FORCDF:	applicable to CDF
COST:	a point that adds cost
DATA FILE CORRUPTION HINT:	related to data file corruption
TREAT AS HV TRIP:	serious enough that data taking should stop: like a high voltage trip in a detector
HUH?	something confusing
INT SAME SIZE AS PRD:	demands an integration machine with the same configuration as production
MAINTENANCE POINT:	useful to note when maintaining
MONITOR TASK:	should be added to automated monitoring
RANDOM NEURON FIRED:	a thought hit at this point: here it is
ORACLE OS Parallel:	supports idea that oracle is an operating system in itself
PARAMETER CONSTRAINT:	a parameter and justification
SCALE Reference:	scales of parameters or databases, also rules of thumb
TIP:	a nice thought
TUNE:	should be in tuning procedures
IMPACTS TABLE DESIGN:	should be considered by anyone thinking about making a table
DATABASE RECOVERY HINT:	impacts recovering the database
NEEDED FOR REPLICATION:	related to CDF replication needs
SQL IMPACT:	impacts the writing and design of SQL
UPGRADE HINT:	useful for knowing when upgrading the database

Chapter 2

Understanding requirements of CDF and how to deal with emergencies

2.1 Chapter 1: Introduction

This chapter is broad reaching and covers a number of topics, including general hints on what 24×7 means and what to do if there are errors.

2.1.1 General Hints

- 90% efficiency = 1 day off each 10 or 36.5 days off per year. This is easy to achieve. But 99.9% is 8 hours off per year.
- Need a Service Level Agreement (SLA): Is it 90%? How is the time distributed
- Need to analyze the problems of what can go wrong and list them
- Need an escalation list for when things go wrong: whom to contact first, then as outage gets worse, whom to contact next from there.
- Need a distribution list for outages
- Analyze our usage of the database: We need a simulation then of that usage to test loads. For this we use other databases.
- We need to analyze hits on the database
- Essential to keep a non-oracle alternative since oracle may want to get more money once we are addicted to them
- Essential to evaluate the ability to pull out of using Oracle on the timescale of one week when Oracle suddenly decides to raise its licensing fee.

2.1.2 When one HAS to turn off Oracle

There are a number of things that cannot be done during the time when a database is available for use:

- patch software
- reorganize tables
- create/rebuild indexes
- VALIDATE STRUCTURE to ensure there is no corruption

2.1.3 The components of a DB System

- We are not in a stable configuration until we monitor. Hard to insist on determining usage of Database, since habits in CDF are just developing NOW. The commissioning run will be vital to understanding these usage patterns.
- There is often a struggle between performance and down time. We can monitor the database to death, working proactively to look for hints of trouble and leave no I/O or CPU for using the database.
- How do we analyze the database crash – do we need a logger program?
- We need to package the database for farms like level 3
- Cost is not JUST money but intractable mysteries in our data like the tracking params not understood or drops in efficiency.
- We are in danger due to lack of a group that consists of the following GURUS:
  - Hardware
  - OS
  - Network
  - DBMS
  - Applications/users
  - Data
- We use a 3 tier structure of expertise: DBA, Physicists liaisons responsible for specific schema, applications people using the database.

2.1.4 Error Analysis

- Manual glitch: pilot error. Measures taken: restrict accounts on our integration and production machine to externally authenticated group accounts.
- MTBF (Mean Time Between Failure) Need numbers for our hardware.
- environmental: Building not available: FCC or B0. Fire, Flood.
- Autoswitch: **DEVELOPMENT TASK:** can we do this to a data warehouse?

2.1.5 Ideal 24×7:

- Robust
- Transparent failover
- Data integrity
- Affordable
- Optimize use of resources
- Easy to implement
- Maintainable
- Performant

2.1.6 Service Level Agreement:

- Must indicate clearly the operations mode
- Costs in dollars of standby and others.
- Need cost of lost beam
- Adaptable: Write as CDF notes and modify Versions. Need to evaluate usage and put into note.
- Specify MTBF MTTR (Mean Time To Recover) based on cost in beam time. Harder to evaluate physicist patience and getting something to present even if wrong.
- Nonambiguous like:

Time	Online?	Beam?	Uptime required	Application running
8-18	Yes	No	Extremely Critical	Run Control

- Interaction with Oracle agreement: last run snapshotted on disk?
- Need to specify steps for each failure type: hence need a list of possible failure types(chapter 6).
- Note the following levels of operation:

99%	3 days 15.5 h
99.9%	9 h
99.99%	50 m
99.999%	5 m

  Useful table to keep in mind when looking at time things take...
- Compare to maintenance in table 1-7: compute times based on size.
- note roll-off vs. partitioning

2.1.7 The Team Needed and the Measure of Performance in CDF

Issues with regard to people and responsibility:

- 24×7 requires high level commitment of a team
- Need a list of responsibilities:
  Time function People responsible
- Measure for CDF: Integrated Luminosity to tape and percentage of good luminosity for analysis.
- Vacation management: who goes when? This rubs on physicists[1].
- Escalation list for emergencies: Problem To call title

2.2 Chapter 2: Emergencies

2.2.1 Items that Influence the Duration

Complexity influences the frequency of emergencies:

- size
- number of schema and segments
- relations
- number of users and batch usage
- number of tablespaces and files
- components involved

2.2.2 How to evaluate the likelihood of emergency:

- look for signs of imminent failure
- switch to failover – **FORCDF:** should we use ACEs in CDF
- assess damage – **DEVELOPMENT TASK:** assement tools needed
- list solutions
- apply solution
- get back to normal
- do post mortem

DEVELOPMENT TASK: We need to develop a checklist that we are doing this for each procedure, as well as detailed documentation (web) of these occurrences. Tools to document seem useful.(automatically fill into html, names of dump files, etc.)

An emergency list is given on page 45.

[1] What physicist gets vacation anyway?

2.2.3 Suggestions to Control Emergency

- Component availability Clause: what has to be available and to whom. **CHECK BOOK: FORCDF:** restrict activity on database to access from online controller only. Need an administrator and put into requirements.
- UPS **FORCDF:** this is in online, what about offline
- Geographic separation
- Evaluation of emergencies done before they occur
- 2GB data file can take 25 minutes to make **ACTION:** evaluate this for our configuration.
- **ACTION:** ODSDB to keep log, monitor of emergencies and a "red book" to pull in case of emergency: big print, checklist, assumes we are idiots (panic). Evaluate ACE readiness.
- **ACTION: FORCDF:** Develop ACE training
- **ACTION:** Create emergencies and drill as part of ODS and ace training

Chapter 3

Configuration of the CPU's, Operating systems, Networks and the usage with Applications and Data

This section covers chapters 3-6.

3.1 Chapter 3: Hardware Configuration

For the Hardware configuration we need a CDF note describing the deployment of the Online and the offline machines. This note needs to be updated when there are upgrades. A "Differences from last version" should be maintained to allow quick reading for those that have been through it already. We should aim to get our work on how CDF has been set up to NIM/CHEP. The online development and integration should (have been) be identical to the production machine to allow real simulation. This comes as a theme throughout 24×7.

To have 24×7 service, one really wishes to have mirrored databases at multiple sites. **ACTION:** We should understand the costs of the various HA (High availability) solutions offered throughout the text and see what the state of the art is in understanding these quantitatively. Ideally we would like to know the cost/% availability for availability above 90% and which of the various items kick in at what time. We can estimate this ourselves by looking at the various recovery times given and the estimate frequency of occurance. The cost of the solution can be estimated and from the previous things (MTBF, MTTR) we can compute the cost. This goes against the items mentioned earlier for savings of "physics" and beam time, also measured as luminosity on tape and luminosity used in analysis.

We also need to consider the usage of the online as fall-back for offline and offline as fall back for online. For the case of an online machine failure, the offline production can be slowed down to accommodate the online. Besides, the online needs heavy activity only at begin and end of run. This would imply that the online database people care about the offline. It is consistent with an integrated approach to CDF operations.

3.1.1 The Extremes of Database Requirements

Some terms:

- OLTP = Online Transaction Processing System
- DSS = Decision Support System

OLTP is characterized by lots of small queries at a very high rate with read, write, modify, delete. DSS is characterized by batch processing and lots of reading but very little modification. They represent the two ends of the spectrum of database usage and as such put contradictory requirements on database design. Real applications are an admixture.

- Do not use readahead for OLTP applications
- Use multithreading – do we
- Use Hot swapping.**FORCDF:** This is in the online, but what about offline?
- MPP to be avoided, use NUMA

3.1.2 Knowing Hardware

Get to know the hardware. **ACTION: FORCDF:** We need to get these specs document for our machines.

- number of disks
- number of cpu's

Disk

Tips and suggestions on disk:

- SCSI, internal, external
- **FORCDF:** Do our disks have consistent size with striping in online and offline? See page 80
- **SCALE Reference:** Oracle claims that small to medium to large disk is 1-4 GB. The user tablespace disk should be big.
- **TIP:** You want internal disks
- **TIP:** Put redo log and data file with table/index on fastest disk
- use external disks if system is old
- **TIP:** use a 1:1 disk:controller ratio **FORCDF:** what do we have?

- Access speed params: **FORCDF: ACTION:** Evaluate against ours

seek time	$<$	20 ms
speed	$>$	7200 rpm
read rate	$\simeq$	8-10 MB/s **HUH?** (bit or byte?)
$\langle$ latency $\rangle$	$<$	4.5 ms

- Take care of the number in a cabinet and the expandability of a cabinet. Disk is cheap, but when going over the limit of a cabinet, it can take a long time to get the next one.

Tips and suggestions on CPU:

- parallel query option (PLQ) takes advantage of CPU **FORCDF:** do we implement this?
- **FORCDF:** what is our max number
- SPIN_COUNT: This is used later on (section 5.1.7) for waiting on latches to get resources. Using this with less sleep eats CPU but reduces latch contention. Need to be sure to keep CPU up to allow for this. Heavy load not only uses raw CPU but this spills over into other items.
- **FORCDF:** CPU could be an issue in external sites
- CACHE should be at least 2MB **CHECK BOOK:** .

Tips and suggestions on RAID

- Raid 0:
  - Disk storage
  - no redundancy
  - no parity check.
  - High performance, low MTBF
- Raid1:
  - two disks written for mirroring
  - most secure: 100% reliability
  - write is slower
  - read faster
  - high MTBF
  - combine 0+1: striping and faster read
- Raid 5:
  - parity used to reconstruct data
  - data and parity spread on all disks

  - read is better
  - write must compute parity
  - Good for a data warehouse (read)
  - 20% overhead instead of the 100% of Raid 1

- Raid S:
  - Stripes parity but not data
  - write cache

- Raid 7:
  - Stripes data but not parity
  - write cache

Summary of Raid Options

- 5/S/7: good for DSS (read intensive applications)
- 0+1: good for OLTP (constant read/write)

Size of Disk Array

- If Raid 5/S/7 MTBF low but number of disks small. 0+1 OK
- OLTP Read small and random, don't want Read-Ahead cache.
- read-ahead cache good for full table and full index scans
- **PARAMETER CONSTRAINT:** OS BLOCK SIZE = DB_BLOCK_SIZE
- **CHECK BOOK:** write cache may be deleting, but what about in farms
- use striping via **FORCDF:** VERITAS
- Striping better than read ahead cache
- **FORCDF:** Farm needs different from user I/O means offline database has conflicting constraints
- **FORCDF:** implies we want 0+1
- **FORCDF: ERROR RECOVERY POINT:** VERITAS has deferred write to allow for error recovery: 15 min delay.

Strip Sizes

- Decide which: OLTP or DSS
- **PARAMETER CONSTRAINT:** consider OS and DB_BLOCK_SIZE: The DB_BLOCK_SIZE must be a multiple of the OS block size. The strip size must be a multiple of the OS block size:
  DB_BLOCK_SIZE = $n\times$ OS BLOCK SIZE
  STRIPE SIZE = $m\times$ OS BLOCK SIZE
  FORCDF: check case for all machines
- striping can be horizontal (across controllers) or vertical (across disk)

Tape Drives

- RMAN
- Legato
- number of tape drives
- **PARAMETER CONSTRAINT:** match disk and tape I/O must be equal and in multiples of volume strip size
- 80GB: 4 hours can become 2 hours if the media are matched

3.1.3 Architecture

Architecture should take into account which components are hot swappable.

- **CHECK BOOK:** OPS for MPP.
- Avoid MPP unless you have a VLDB **FORCDF:** don't, yet.

NUMA

Some terms:

- UMA is Uniform Memory Access and NUMA is Non-Uniform Memory Access
- Uniprocessor: Single Processor NOT an option for Databases **FORCDF:** but we rely on this for offline!
- SMP: Symmetric multi processor: cluster of SMP good for database
- UMA SMP limited to 12 processors
- NUMA MPP is an option

NUMA has local groups of memory where communication between groups is done with a shared local bus. More like MPP

An OLTP database should not use NUMA because it leads to a low MTBF.

3.2 Chapter 4: Operating Systems

Unix is the operating system of choice and is of course what is used in CDF.

3.2.1 Kernel

ACTION: FORCDF: The kernel parameters should be audited and documented.

- cannot be swapped out or paged out
- for more users, and files, needs customization
- can be trimmed if raw file access is done: for OPS (oracle parallel server) raw devices are essential) **FORCDF:** unlikely we would want raw devices
- need to check /monitor logs: **FORCDF:** sysadmin/ace interaction?
- for SUN, log is in /var/adm/messages

Set These Params High to avoid reboot

- NPROC
- MAXUSERS
- NFILE
- NINODE
- VLIMIT (biggest oracle export dump size determines this)

Decrease these parameters for Raw Disk Usage

- BUFPAGES
- NBJF
- BUFHWM (high water mark)
- SYSPTSIZE (page table) **PARAMETER CONSTRAINT:** related to the buffer cache4.1.2

Swapping and Page parameters

- LOTSFREE paging starts
- DESFREE desperation swapping
- MINFREE out of memory
- MAXSLP when to become swapped

Semaphore parameters

- SEMMNI sets
- SEMMSL number/set
- SEMMAP number
- SEMMNS total
- SEMMNU undo entries

Shared Memory

- SHMMAX Max Shared Memory **PARAMETER CONSTRAINT:** > SGA of biggest oracle
- SHMMIN min
- SHMNNI numbef of segments
- SHMSEG max segments/process
- SHMALL max concurrent number

PARAMETER CONSTRAINT: Need to know the OS logical and physical blocksizes. The DB_BLOCK_SIZE is related to the OS block size. **PARAMETER CONSTRAINT: FORCDF: The implication is not only that we need to get a kits server installed but that that the FERMI REDHAT is also tuned to this!**

3.3 Raw Devices

Raw Devices are unformated disk partitions. Oracle can take responsibility for handling the disk **It becomes evident that Oracle plays the role of an operating system on top of an operating system: this is a general observation [3] and is not restricted to Raw Devices; however, it becomes very evident here.**

3.3.1 Some Unix file types

- Regular
- Devices /dev
- Links
- named pipes
- sockets

3.3.2 Comparison of Cooked and Raw devices:

- cooked: block and character device file → sysdriver
- raw:
  - Character only → own drive
  - only accommodates one regular file per raw partition
  - owned by user under which oracle was installed
  - mandatory under Oracle Parallel Server (OPS)
  - Helps if disk i/o is a big deal
  - better to have enough disks and controllers so that disk i/o is handled that way
  - check untuned SQL before taking a dramatic step like "going raw"

  RANDOM NEURON FIRED: FORCDF: Do we want a cdf_web so we can block the cdfonprd database during runs

3.3.3 Pros and Cons of Raw

pro

- bypass filesystem buffer
- more memory available

con

- loose flexibility to move files that are used a lot
- must maintain partitioning yourself
- UNIX commands cp, rm, mv don't work
- no buffer

3.3.4 Tips if you Go Raw

- Assign as many partions as you have tablespaces. This makes extents much harder
- Have to decide on sizes: a nightmare in maintaining
- End up with a mix of raw and cooked: and need to back up
- Use 3 sizes (ie. standard sizes)
- Put online redo logs on raw devices (A *must* for OPS)
- Do not use cylinder 0. This overwrites the disk partition information
- Create symbolic links
- See book: Making the Decision to use UNIX Raw Devices [4]

3.4 System Capacity and Bottlenecks

Run CRON to monitor the system:

- RAM availability
- page swap
- swap space
- i/o
- uptime system load averages over last 1, 5, 15 minutes
- top sar vmstat

Problems can arise from bad SQL: **MONITOR TASK:** Monitor the performance using standard SQL. The CDF API should help in this standardization. **ACTION:** This kind of monitoring needs to be done for a simulated load.

3.4.1 Items in Oracle that Require Memory

PARAMETER CONSTRAINT:

- SORT_AREA_SIZE
- SPIN_COUNT
- SGA

DEVELOPMENT TASK: ORACLE OS Parallel: Is there a system alert like OEM?

3.4.2 What goes Where and Fiddling Parameters

- Distribute amongst many servers → keep non-Oracle stuff off the Oracle server
- **ACTION: FORCDF:** remove non-Oracle processses, like a web server! Ask Randy to check performance and run remote and local requests
- **ORACLE OS Parallel:** Do not set priorities on Oracle processes: Oracle will handle this and you can get serious problems. Oracle uses preemption of processes according to a schedule. It has built-in scheduling functions. There will be confusion over performance if one lowers the priority of processes. It is easy to lower the priority of the wrong process.
- Do not use processor affinity
- Detect user levels
- Observe production etiquette
- **MONITOR TASK:** Production monitoring uses CPU and other resources

3.4.3 Diagnosing the System

- Page Fault
  - **MONITOR TASK:** Examine swap out, page out, swap in, page in , page fault
  - use vmstat and sar
  - vmstat 5 5 means poll 5 times in 5 seconds
  - ignore the first row
- Memory leaks
  - **MONITOR TASK:** look for bug reports on metalink
- Swap Space
  - **PARAMETER CONSTRAINT:** should be 2 to 4 times the physical memory
  - for > 1G use $\times 2$
  - for < 1G use $\times 4$
  - Solaris has a "virtual swap file system" but Oracle will have trouble if this is used.
  - Distribute swap across your fastest disks
  - Use striping: Raid 0
  - Check if there is a limit on swap file size
  - Solaris 2.7 has ability to do 64-bit **FORCDF:** What are our memory capabilities?
  - Lock shared memory in RAM
- Disk and I/O
  - Take care which tables are on which disks
  - **ACTION: FORCDF:** There are 3rd party software solutions for monitoring. Needs evaluation, **RANDOM NEURON FIRED:** connection with farms, L3?
  - iostat 5 5 give 5 intervals every 5 seconds
  - iostat -x 5 2 t pools 5 second interval
  - if no iostat, use sar **FORCDF:** what do we have

3.4.4 Reducing I/O Bottlenecks

- striping
- VERITAS provides excellent striping
- **MONITOR TASK:** v$database and v$filestat **CHECK BOOK:** check hot files

- **ACTION:** Get a map of disk to unix display
- Use journaling on OS
- Monitor free disk space. **FORCDF:** Roll off protects against lack of disk space. The problem is that disk is cheap, but you have to order, install and ensure there is a cabinet slot available. When an emergency occurs and you need 1 G, it is mighty expensive.
- Clean up directories with sizes in multiples of 512K.
- Truncate logs (take care that you cannot DELETE them! Otherwise the file pointers are still active but you cannot see the file. You helplessly watch the disk fill up and have to bounce oracle to get the file pointers reset.)
- mirror the root file system.
- $>$ 2G requires large file support: take care on 32 bit machines **FORCDF:** This could be an issue in the exported databases on linux
- examine /var/admin/messages (sun solaris). **FORCDF:** the offline time? cdf operations? use tail -f in CDF control room?

3.5 Chapter 5: Networks

There are two basic problems that arise in networks

- takes too long to get data (or goes down)
- accuracy problems

3.5.1 MONITOR TASK: Tools to Monitor Networks

- Ping
- cable analyzer **FORCDF:** Do we have one?
- Don't use NFS
- Do not use DB server as an NFS server at the same time
- Use a subnet: **FORCDF: ACTION:** get diagrams of networks
  - Put NT and our webserver on the same subnet
  - Put Online on same subnet
  - Put warehouse on same subnet
  - put validation database on subnet with the validation machine

  → This implies that all offline machines are on the same subnet
- **PARAMETER CONSTRAINT:** check QUEUESIZE

- **PARAMETER CONSTRAINT:** NAGLE disable this
- **PARAMETER CONSTRAINT:** set packet size in SQL NET to match MTU
- Not Overloaded **CHECK BOOK:**
  - netstat tool
  - FIN_WAIT_2 → can crash Kernel
  - **FORCDF:** Do we have a timeout?
  - This could bring us down in remote replication. See page 164
  - netstat -i:
    - input error: indicates faulty hardware
    - output error: indicates problem on local net interface
    - collisions: indicate network overload **FORCDF:** check collisions at runstart

3.5.2 Customizing SQL*NET/Net8

- **PARAMETER CONSTRAINT:** For high volume need to increase the listener QUEUESIZE in listener.ora. This relates to the bandwidth and hence concurrent connections
- SUN: Uses ndd to set. Queue is 32. This is 32 concurrent requests. Max in SQLNET 2.1 is 1024 concurrent processes. **FORCDF:** we don't have this many licenses!
- logged in as ROOT on Solaris do
  ndd -set /dev/tcp tcp_ionn_req_max 1024
  then need to change /etc/rc2.d to be sure it is picked up at reboot. Need to reconfigure listener. See page 168.
- **PARAMETER CONSTRAINT:** Disable Nagle Algorithm. It introduces a 200ms delay. use tcp.no_delay = true in protocol.ora or disable at the OS level. **FORCDF:** easier to handle the oracle params at oracle level.

3.5.3 Constraints and Parameter Interrelations Introduced by Network

Set the SQLNET packet to match the protocol MTU **CHECK BOOK:**Ṡome terms

- TCP = Transmission Control Protocol
- IP = Internet Protocol
- MTU = Maximum Transmission Unit. Must be the least common denominator of all units: Etherent is 1518 bytes, token net is 4440, X25 is 4Kb) **RANDOM NEURON FIRED:** Could cause headaches at remote sites if different nets than ethernet is used. Need to add: supported only for optimal performance with Fermi RedHat and Ethernet?

ACTION: FORCDF: These items should all be checked aginst the CDF ideas

- **COST:** Configure specific path for access to Oracle. Backup path is recommended but this gives a "higher cost of service"
- Need to set Net 8 packet to MTU size. In Version 2.3, can set SDU, TPU sizes in listener.ora and tnsnames.ora. **TUNE:** This gives a 5% improvement with tuning and tcp.no_delay=true.
- Check tnsnames.ora for TDU, SDU
- Use a sniffer (protocol analyzer) to check fragmentation.
- For retrieval of large number of rows, can use a set arraysize. **IMPACTS API DESIGN:** This impacts the API design. Can use the table size estimates to determine the deafault argument and use the argument in code to tune access. This becomes important when retrieving more than 1500 bytes. Need to take the following into consideration:
  - Number of rows
  - row size
  - protocol MTU size
  - NTE8 SDU/TDU size

3.5.4 Tips for Managers on Networking

- Document all parameters **FORCDF:** electronic logbook
  - Keep a change log
  - Keep a repair log
- System administrators should keep a copy of these documents at home in case of power/system failure (electronic log not so easy)
- DBA to keep in touch: **ACTION:** read 24×7 .
- Manage workload
- SLA with vendor to provide support **FORCDF: ACTION:** document
- Heat monitors, enviromental monitors and controls **FORCDF:** control room, unified with l3, farms
- No Friday changes
- Cell phones
- No FOOD in computer room **FORCDF: ACTION:** signs on 3rd floor b0. What about fcc?

3.6 Chapter 6: Applications and Data

Much of the CDF work with the database is isolated from most physicists via the API. This allows a large degree of control over the kinds of queries to the database. Some additional considerations arise that should be applied to the API.

3.6.1 Control of the Application

- **IMPACTS API DESIGN:**Modify the API with a maxrows default coming from the java and overrideable in the application. This can provide for a default level of protection. It may not work for all memory overwrites.
- **ACTION:**DBA's need to gain a more intimate understanding of the CDF applications
- **ACTION:**Need to list all applications. They fall into the following categories:
  - those that use C++ templates – online, offline – and are using generation. The DBA will get a uniform application layer and, via the developer, can exhibit a level of control. This is at the cost of not having all the Oracle Designer bells and whistles: **ACTION: IMPACTS API DESIGN:** the desire then is to upgrade the api so that the designer parameters are specified. Alternatives are to modify the current templates or go to XML.
  - Those that use JDBC or some java interface are more restricted to online. Here the users are directly accessing the database with some level of interface. Control may be less.
  - DBANA database analysis tool that reads the database using the code generation tools.
  - The database browser that uses JDBC/SQL. **ACTION:** It is important to understand what kinds of things it does, especially the full table scanning. It may need to be restricted and we may need to make it do intelligent queries to see if the database is in some access state. We may wish to consider some kind of priority table that processes access to determine what they can do. It definitely should live on a machine other than the database server.
- **ACTION:**Need to list the applications by the form on table 6-1 page 187:
  - Application
  - Functionality
  - Nature
  - Who
  - Tables
  - Max/Min
  - Compoonent
  - Shared tables

→ This kind of information can be code generated into HTML

- If we have read/write tables and other that are mostly readonly, we should sort them in separate databases and use replication in a master-slave fashion. This implies, for example that we want DBANA to run on the offline machines for analysis. When DBANA does writes, it should run online! For our application, we:
  - analyze the table
  - write/read it offline
  - establish the quality of the calibration
  - write it online

  This can lead to synchronization problems and would drive a need for advanced replication. Hence, it would violate the rules, but since there are write periods (calibration and normal end run) and read periods (start run) sychronization will make this possible.

- We need to understand the balance of planned and ad-hoc queries.

A quote: (page 186)
"If your end users are surgeons or nuclear physicists, they may have more urgent requirements"

RANDOM NEURON FIRED: Danger of web browser: can we get outside trouble? Can an insider screw up the calibration by making a big query. Need at least some level of password protect.

3.6.2 What applications need

- priority: **ACTION:** create priority and registration tables. There to some level as processname in the calibration schema. Need to consider the other schema.
- Tuning
  - Must be done during and after development
  - Code Review:**MONITOR TASK:**
    * **PARAMETER CONSTRAINT:** TKPROF: SQL_TRACE and TIMED_STATISTICS=TRUE required on an integration database – perhaps on the validation database as well
    * Always have a test instance (ie. integration) that mirrors production. Does this mean we need a fourth instance?
    * Simulate a production run
      · tunes monitor
      · can be done nightly as validation
      · can shut down dev nightly
  - Independence of Future Schema Changes
    * View on top of each table (readonly?)

- Avoid select(*) but select specific columns that application needs **IMPACTS API DESIGN:** does API do this?
- Use record variables **HUH?**
- Use table driven application logic:Place code into a table and use dynamic sql. Causes a 5% performance deterioration.

– Transaction Splitting to clone Data
  - Keep copies of tow tables and write to both
    - **DATA FILE CORRUPTION HINT:** is this a DFC help for the corruption worries?
    - need to consider this on each table

– use OCI (we use OTL as better to maintain)

– Implement Failover Capabilities
  - **FORCDF:**online ←→ offline?
  - Designate a standby. **FORCDF:** again, online/offline? Does the CID give a problem? Does this imply the online is really the be all end all? This would imply that the online database group does care dearly about the offline.
  - Autoswitch:? **FORCDF:** use run control parameter/button and trained ace **ACTION:** again need training of aces
  - For offline batch processing, what would we do? Autoswtich to warehouse **ACTION:** evaluate
  - Failover for hung or dead processes?

– Middle Tier
  - As a router
  - Need Multiple middle tiers: degrades MTBF
  - Allow direct connection
  - Turn it around. Use direct MOSTLY and then add middle tier on critical tables.
  - Example: The database is extracted into flat files and shipped to the level 3 nodes. These nodes do not access oracle. The shipping procedure is controlable. **FORCDF:** Should the same mechanism be used for the farms?
  - Implies limited usage instead of a general or commercial middle tier.

– Session State information
  - If a failover occcurs after this was set, then you can get confused. **FORCDF:** Do we have any applications that change the session information? Do we need to abort and redo?

– Data Integrity
  - **HUH?** Can we get a new primary log, some information
  - Usually commit after all rows are modified. The Data Catalog is susceptible. **ACTION:** Work out failure scenarios and test/simulate

    * List of failure codes to consider:
        - 1033
        - 1034
        - 1089
        - 1092
        - 3113
        - 3114

      IMPACTS API DESIGN: Can we add recovery to the API?
- Automatic failover option
    * Can map service names on to multiple instances **FORCDF:** we need to test this
    * Need to have tnsnames.ora as failover. This would allow us to distribute later while updating tnsnams later. We can get into nasty staleness problems when failover occurs and our backup is old.
- Transport Application Failure (TAF)
    * Failover callbacks can be done. The call back handles the failover condition **IMPACTS API DESIGN:**
    * can use static failure (automatic is given above)
    * use the callback
        - must define and register the callback
        - **HUH?** must signifiy this
        - **IMPACTS API DESIGN:** Possible with the calibration API
        - Possible with hardware database
    * **FORCDF:** Auto failover to offline implies a demand on advanced replication. This is a lot of effort.
    * preconnect can be used to get quicker into failover: in tnsnames.ora or

nameserver:	TYPE	NONE	no failover
		SESSION	redo statements
		SELECT	online select failover
	METHOD	BASIC	online failover
		PRECONNECT	acquires secondary at same time as primary

      Can use ALTER SESSION DISCONNECT SESSION
    * MTS (Multithreaded Server) recommended
    * tnsnames should not explicitly give SID. Use LOCAL_LISTNER
    * **TIP:** Must use OCI8 Libraries to take TAF from setting only tnanames.ora! **ACTION: IMPACTS API DESIGN:** check OTL
    * Really only works for select
    * Can use SHUTDOWN TRANSACTIONAL to be able to smoothly failover.
- Failover via Middle Tier

  A middle tier:
    * Accommodates a large user population

    * does load balancing
    * coordinates access to multiple systems
    * insulates the users from the business logic
    * results in higher availability

  Example: Browser-WebServer-Database
  OAS, Oracle Application Server is a platfor for COBRA, Common Object Request Broker Architecture.

  - **PARAMETER CONSTRAINT:** have MAXEXTENTS set UNLIMITED but not for rollback and temporary tablespace.
  - Expedite data loads. Keep the indexes to a minimum when doing bulk transfers.**IMPACTS API DESIGN: TIP: MONITOR TASK:** if > 15% of a full table is needed, do a full table scan, not an index scan. **ACTION:** we need to understand this page 225.
  - Disable expensive constraints during the loads. WE may want to do this if the API also is insisting on constraints anyway.
  - Page 226 **HUH?** What do they mean by delayed constraint?
  - Use a procedural language **IMPACTS API DESIGN:** we do this with the code generation, but what about the rest. Suggest use of PL/SQL, JAVA, C instead of pure SQL to do a join. This trades CPU against I/O
  - Consider a Cost Based Optimizer **MONITOR TASK: HUH?**
  - **MONITOR TASK:** dump SQL commands of users into a file and monitor use v$sqltext_with_newlines
  - **MONITOR TASK:** look at jobs being run concurrently offline. See if they are doing anything
  - Use version control code (Ie. we use CVS)

3.6.3 Managing Indexes

This is a rather confusing subject that is subtle and needs review on all tables. The usage patterns are essential and the **MONITOR TASK:** tools to examine the indexes are needed.

There are two major index types and variations on each of them:

- B* tree
  - Partitioned
  - Reverse-key
  - Index organized Tables
  - Clustered Indexes
- Bitmapped

In addition, they come under various other categories – not all combinations of the above types and the following are valid

- Local/Global
- Prefixed/non-prefixed
- Abstract data type
- Function based
- Application domain

3.6.4 B*Tree Indexes

Properties:

- look up a pointer into large amount of information
- A leaf contains multiple physics rows
- Balanced: each leaf equidistant from root
- Can cause splits of branches – so sometimes insert suddenly takes a while. Hence try to presplit so that the user gets uniform response.

TIP: Tips:

- Avoid indexing columns subject ot high update rates
- Avoid multicolumn index if there is a high delete rate. Use a combination of partitioning and truncation.
- Avoid index on low cardinality **FORCDF:** evaluate the cardinality of the indexes as described in **MONITOR TASK:**r̃efcardinalsins
- Null values never stared in single column indexes – **IMPACTS API DESIGN:** already controlled by API?

3.6.5 PCTFREE and indexes

- Determines the extent of a leaf split
- used at index creation
- Says how much is index, how much is real information. For example, PCTFREE=20 means 20% index, 80% table data.
- **IMPACTS TABLE DESIGN:** use larger value for OLTP and Popular table
- **IMPACTS TABLE DESIGN: MAINTENANCE POINT:** Large values increas the index creation time. This in turn implies a longer amount of downtime needed in maintenance.

3.6.6 Partitioned Indexes

- Allow easy truncation
- GLOBAL and LOCAL indexes.
  - Local: specify table partition
  - global: multiple table partitions

  Example Table has 52 partitions. (weeks of the year). If indexed with 12 index partions, these are global. If indexed with 52, then these are local partitioned indexes.
- Considerations
  - intense OLTP with large number of rows
  - high availability: No index rebuild
  - partition elimination: ie. move stuff to disk.
  - do not index on keys prone to constant update
  - **SCALE Reference:** theory: thousands are ok

3.6.7 HUH? Prefixed and Non-Prefixed

- prefixed:Index contains the partition key
- **IMPACTS TABLE DESIGN: TIP:** Local prefixed indexes should be created as often as possible

3.6.8 Reverse Key Index

Used to eliminate hot spots where the index key is used in more than run case **FORCDF:** Perhaps the CID and run number will do this to us. **MONITOR TASK:** better look at what is happening to the indexes. Use ALTER SESSION SET EVENTS ... TREEDUMP

3.6.9 Index Organized Tables (IOT)

B*Tree on a Primary key:

- If Primary key *is* an index, then we are happy. **FORCDF:** this is the TSID and the CID.
- Cannot alter the table structure
- Primary key is needed
- non-primary indexes are not available
- Row chaining occurs for large rows.

3.6.10 Bit Mapped Indexes

- low selectivity
- bitmap stream, up to 30 columns
- **MONITOR TASK: IMPACTS TABLE DESIGN:** look at % of unique tables
- Need to be sure of % rows with unique

3.6.11 IMPACTS TABLE DESIGN: MONITOR TASK: Cardinality and when to use what kind of index

no cardinality	index uesless	
low	bitmap	Bitmaps get expensive if used often, bad idea.
high	use B*Tree	

3.6.12 Other indexing

- Abstract data types and their indexes: identify scalar types **HUH?**
- Function based indexes **FORCDF:** perhaps a good option for classifying dead channels in calibration database
- application domain indexes: created on mulitmedia data

3.6.13 Pearls of Wisdom on threads, shared libraries and indexes

- "Indexing is an art, not a science"
- Use threads rather than processes since it takes too much time to start a process
- Use shared libraries since they save memory. They do cause some delay at run time. Worst on first call. **FORCDF:** how much runtime delay do we end up with?

3.7 Physical Design

Identify segments to be considered in design:

- disk
- backup/recover
- periodic maintenance
- performance
- availability

PARAMETER CONSTRAINT: Impacts rollback segment, temporary segment, redo log and overall size. **ACTION:** We should use tables on pages 254-257 to see how much

- in our schema
- in designer
- forgotten
- /api/ generateable

3.7.1 The Top 20 Checklist in Physical Design

1. Parallel query option
2. Cost based optmizer
3. Multiple pools: recycle pool if big. Keep pool if small
4. partitioned segments
5. Fast full index scans (examples in use)
6. Bitmapped indexes – for low selectivity (cardinality) column
7. Reverse key indexes for lopsided B*Tree
8. Index organized tables **FORCDF:** TSID, CID
9. Full table/ index scan
   PARAMETER CONSTRAINT:
   DB_FILE_MULTIBLOCK_READ_COUNT*DB_BLOCK_SIZE $\geq$ 64K
10. NONLOGGING/UNRECOVERABLE usage for very large loads Archiving only in production, but test performance on a test.
11. Never
    join and subquery execution path. **TUNE: PARAMETER CONSTRAINT:** enable HASH_JOINS, SEMI_JOINS, ANTI_JOINS,
12. constraints declare and defer
13. **PARAMETER CONSTRAINT:** FREELISTS, FREEGROUPS AND INITRANS if multiple inserts to occur concurrently on a single segment. **FORCDF:** CID in the CALIBRUNLISTS what about datafile catalog and hardware database?
    - FREELISTS: number of concurrent processes
    - FREELIST_GROUP: number of insert operations across instances
    - INITRANS=FREELISTS
14. Rollback Segments: make one medium and large, disable the large. Enable the large only for specific jobs or extend and let large jobs use them. Consider creating one mammoth 100MB or 500 MB disable all others and enable this. **FORCDF:** for silicon

15. clusters prejoin related tables for medium sized tables related to one another. Small tables too insignificant.

16. Denormalization of fact tables: don't do this since it may need too much joining

17. **IMPACTS TABLE DESIGN:** large tables and indexing. Do not use index extensively. Analyze potential use patterns

18. Equality Predicates and Usage of newer index types on date columns. If time not needed, store only dd/mm/yy: consider denormalizing.

19. Long Data types. Don't use log, long raw, use varchar and LOB

20. Materialized Views **IMPACTS TABLE DESIGN: FORCDF:** examine for our views

    - allows material to be sumarized once and used many times
    - can be refreshed each time or incrementally
    - related to snapshots
    - requires cost based optimizer

3.8 Application Related Tips

- Enforce Standards
- **MONITOR TASK:** Monitor with v$sqlarea and v$sqltext
- optimize oft-used queries **FORCDF: IMPACTS API DESIGN:** check other databases
- **MONITOR TASK:** deadlock search Large row insert by one program and others trying to get in can create deadlock. **FORCDF:** Potential in data file catalog.
- DBA must montitor **MONITOR TASK:** Monitor USER_DUMP_DEST and BACKGROUND_DUMP_DEST
- Do not use discrete transactions **HUH?**
- Partition

3.9 DATA FILE CORRUPTION HINT:Data

Corruption of data has a large potential for downtime. Things to consider when thinking about data corruption:

- What is corruption
- source
- how spread amongst system resources

- which applications run
- who end users are
- frequency of access
- if the data are valid
- **MONITOR TASK:** if checks can be made for logical or physical corruption
- Need to understand schema. Need to know the proper driving tables. Pick the least number of rows **HUH?CHECK BOOK:**

The data should be categorized:

- group segments
- backup requirements:table 6-9 on page 269
- segment size **SCALE Reference:**

XL	> 5 MegRow	> 1 GByte
L	1-5 MegRow	> 512 MByte
M	10KRow-1 MegRow	> 100 MByte
S	< 10KRow	< 100 GByte

- **ACTION: IMPACTS TABLE DESIGN:** Growth: for partitioning may want future partitions to have a small extent. **PARAMETER CONSTRAINT: SCALE Reference:** $n \times$DB_BLOCK_SIZE, $n < 10$.
- **MONITOR TASK:IMPACTS TABLE DESIGN:** Put rapidly growing large segments in a separate tablespace
- **PARAMETER CONSTRAINT:IMPACTS TABLE DESIGN:** Initial and next extents should be multiples of DB_BLOCK_SIZE. There should be a predefined number. Table 6-11 is helpful here. Can use the STORAGE clause in designer.
- **IMPACTS TABLE DESIGN: PARAMETER CONSTRAINT:** PCTINCREASE set to 0
- **IMPACTS TABLE DESIGN: PARAMETER CONSTRAINT:** PCTFREE: low is 5. If doing few updates, keep low. for lots of updates make high (Prevents fragmentation)
- **IMPACTS TABLE DESIGN:PARAMETER CONSTRAINT:** INITRANS=FREELISTS max number of concurrent transactions. should be bigger than the default of 1.

Chapter 4

Setting up the databases, upgrading, migrating to new machines, backups, recovery, archiving, starting and stopping

4.1 Chapter 7: Install/Configure/Customize

The goals are:

- performance
- maintainability
- availability

4.1.1 Server Configuration

For this it is recommended to follow the OFA standard. This is "Optimal Flexible Architecture" not to be confused with "Optimally Formalized Folklore for Basic Architecture Server Entities", OFFBASE. The directory structure is in Figure 7-1. This should be uniform across all instances. **FORCDF:** is this the case? The reason is to allow uniform easy maintenance. More details can be found in the DBA handbook, Chapter 4 [5].

- Use config.ora : permanent parameters are here. The init.ora parameters:
  - DB_BLOCK_SIZE
  - DB_NAME
  - BACKGROUND_DUMP_DEST
  - IFILE

RANDOM NEURON FIRED: Can we overcome manpower shortage with self-training

Use crdb_SID.sql and crdb2_SID.sql

- gives a checklist
- documents values
- rename to crdb_SID where SID is value in $ORACLE_SID

Attributes of crdb2_SID:

- nonsystem tablespace creation
- rollback creation
- run catalog catrpoc catdbsyn
- SYS and SYSTEM passwords
- database user, role profile creation
- pupbld.sql (Product User Profile)
- to run: ALTER DATABASE, ALTER SYSTEM
- update if system parameters chnage.
- **MONITOR TASK:** examine logs when it is run.
- **MONITOR TASK:** need logs for database audits

Ensure disk for 2 weeks of trace files and alert logs

FORCDF: need to check this in our case, connected with usage. Oracle software disk should be separate from these logs. By default it is not the case **ACTION:** . Software needs could be interrupted by a log. Background, core and user dump should be kept away from software directories.

Consider OS block size and applications for DB_BLOCK_SIZE

PARAMETER CONSTRAINT:

- DB_BLOCK_SIZE = $n\times$ OS Block Size, $n \geq 1$.
- large database scans: big block size (DSS)
- small hits: smaller block size.(OLTP)
- **FORCDF:** what is our table usage (look at each) How does it influence our exported databases.
- Large blocksize causes contention over data.
- **DEVELOPMENT TASK:** Can play with PCTFREE, FREELIST AND INITRANS if one has a large blocksize and wants it smaller for a specific table.

- Large block size requires bigger redo log **ACTION:** compute needs
- Small block size causes blob/clob to chain: **FORCDF:** SVX needs big.

Problems that arise from mismatch

- bad performance
- suboptimal i/o
- cache
- segment fragmentation(Section 5.2.1)
- rowchaining(Section 5.2.7)

Ensure contiguous blocks/write are high

MONITOR TASK:PARAMETER CONSTRAINT:

- CCF_IO_SIZE: contiguous blocks/write during database creation 1-131072.
- Use LOG_CHECKPOINT_TIMEOUT and or LOG_CHECKPOINT_INTERVAL to ensure instance recovery SLA is met
  - LOG_CHECKPOINT_TIMEOUT determine seconds if only a couple of writes amongs reads, then huge redo search checkpoint no triggers.
  - LOG_CHECKPOINT_INTERVAL determine how much of redo log you rely on
- **PARAMETER CONSTRAINT: TIP: TUNE: SCALE Reference:** Rule of thumb: 1ckpt/5Mb or 20 min. **FORCDF:** what do we do
- check points use physical disk I/O and affect performance
- Check recovery time with a shutdown abort or integration instance **RANDOM NEURON FIRED:**UPS does a shutdown abort. Check time to recover. database performance vs recovery time

Create > 3 mirrored redo logs for high activity DB

- **FORCDF:** spread all redo logs files in a group into different disks/controllers
- LGWR fills log ans swithces to next group
- ARCH reads for acrhving.
- AIX wanted it too, to mirror. ARCH saw it and cried. Cure with two redo logs per group **ORACLE OS Parallel:**
- **FORCDF:** Second placed on non-mirrored volume. Do we do this?
- under heavy usage end up with redo not written and run out of groups – ckpt interaction. See page 292.

- **FORCDF:** relate the peak write rates to this and check if susceptible during this period.

Conclusion: 2 logs/group with one on a non-mirrored system DATABASE RECOVERY HINT: Control files:

- mirror the control files **FORCDF:** had this problem
- 3 on different disks/controllers
- DB crash guaranteed if it cannot write to even one copy
- To recover, write over corrupt one with valid, recovery automatic at startup

Place redo logs so ARCH and LGWR don't contend

- LGWR writes to next redo log group when a log switch occurs
- ARCH immdiately crates an archive of redo log
- **PARAMETER CONSTRAINT:** ARCHIVE_LOG_DEST must be a different disk/controller than the redo logs. Picture on page 295/296.
- ARCH is slower than LGWR so contention can get difficult
- **MONITOR TASK:** watch for this during peak and simulation
- **RANDOM NEURON FIRED:** We should put together a trivial pursuit for oracle to train us.

Configure Adequate Redo Latches

- latches:
  - redo allocation
  - redo copy
  - redo write
- **PARAMETER CONSTRAINT:**LOG_SMALL_ENTRY_MAX_SIZE not in 8.1.6 but in 8.0.5
- **HUH?** page 299 explanation seems backwards
- **PARAMETER CONSTRAINT:** number of latches = LOG_SIMULTANEOUS_COPIES = CPU_COUNT (set automatically)
- **RANDOM NEURON FIRED:** 24×7 must have > 1 processor hence offline is not 24×7 .
- for high activity latch params are critical
- **MONITOR TASK:** check v$sysstat: details page 301

4.1.2 Ensure DBWR can keep up wixth database loads

PARAMETER CONSTRAINT: Parameters summarized on page 317. DBWR:

- flushes buffer during a checkpoint
- cleans dirty buffers when clean not enough
- primary background process
- starts up after DMON
- starts up before SMON
- **MONITOR TASK:** in v$process view has PID=3
- moves changed (dirty) date to files
- keeps lists:
  - LRU
  - LRUW
  - Hash Bucket Chain List

Buffers in LRU:

- Frequency: at tail of linked list
- dirty: write at next ckpt by DBWR
- Pinned dirty: Being written to
- Pinned Clean: latched

Buffers in LRUW

- Dirty data block headers
- used by DBWR to work. If it has time, looks into LRU. If too busy, server may pop the dirty ones in LRU to LRUW.
- writes in _DB_BLOCK_WRITE_BATCH units
- LRUWSIZE _DB_LARGE_DIRTY_QUEUE /mon/ page 304 to see this

The hash bucket chain is the map of data-block buffer cache. **ACTION: MONITOR TASK:** Need a test instance and simulations of load to study the DBWR.

- Page 308-309 for list of observations to make.
- Negative values mean rollovers.
- **MONITOR TASK:** page 309 for monitor of DBWR params. catpef.sql creates scripts

- free buffer waits: server forced to wait for a free buffer
- buffer busy waits: because buffers were busy **SCALE Reference: TIP: MONITOR TASK:** BBW/(consist get + db block gets) < 5%.
- write capable **HUH?** : need to read on block writes to by checkpoint
- DBfile: indication of i/o bottlenecks as server waited for these to complete:
  - segread
  - single write
  - scattered read
  - parallel read

  Some of this is a checkpoint baseline rate.

A number of parameters need to be set and monitored:

- **MONITOR TASK:** v$latch
- cache buffer handles: can't get a latch for a clean buffer. DBWR throughput so bad it wrote but did not free the buffer
- cache buffer chains: lots of database addresses in a bucket. need oracle support:_DB_BLOCK_HASH_BUCKETS. **MONITOR TASK:** page 311.
- cache buffer lru chain: may be DBWR works suboptimally. This is the unit to clean dirty buckets **SCALE Reference:** > 1000 bad
- enable synchronous i/o: if going to a variety of disk controllers Uses polling to see if asynchronous i/o worked **PARAMETER CONSTRAINT:** DISK_ASNC_IO. **FORCDF:** sun: default behaviour kernelized asynchronous i/o. Puts requirement on memory.
- list i/o: process i/o in a list. **FORCDF:** sun does this, what about linux?
- **PARAMETER CONSTRAINT:** MULTIPLE_DBWR_SLAVES if no AIO **FORCDF:** not sun, but linux?
- **PARAMETER CONSTRAINT:** MULTIPLE_DBWR_PROCESSES : 8.0.4 was DB_WRITER_PROCESSES
- **FORCDF: MONITOR TASK:** increase LRU latches: **PARAMETER CONSTRAINT:** DB_WRITER_PROCESSES=DB_LRU_LATCHES=number of processors. Listed as ora_dbwr_SID.
- **FORCDF:** "Have a decent number of cpu's" **SCALE Reference:**8 is good. 1 or 2 is low. if about 4 and no asynch i/o use slaves. What does this imply for linux? **RANDOM NEURON FIRED:** what happens if linux is shut off: does it recover? Tough for external sites. Do we require UPS?

- **PARAMETER CONSTRAINT:** increase DB_BLOCK_CHECKPOINT_BATCH to 8. This puts a strain on memory. max blocks written in a check point
- **PARAMETER CONSTRAINT:** increase DB_FILE_MULTIBLOCK_READ_COUNT: **CHECK BOOK:** may ignor max database blocks you can pack into a single read. Depends on OS and:
  - DB_BLOCK_SIZE
  - DB_BLOCK_BUFFER
  - processes
- want **MONITOR TASK:** 32-64K/read. much of unix takes 64K as upper limit. **PARAMETER CONSTRAINT:** cannot be > DB_BLOCK_BUFFER/4
- increase DB_FILE_SIMULTANEOUS_WRITES. DBWR writes in groups or batch. This is max number of simltaneous writes/file. useful if data files striped amongst controllers of if OS can consolidate writes to i/o. **SCALE Reference:** set to between 4 and 24.

The parameters for initialization are summarized in page 317.

Partition buffer cache into multiple pool buffers

MONITOR TASK:

- This depends on the segment usage patterns – and again needs monitoring.
  - keep - referenced time and time again
  - recycle - referenced once
  - default - most used
- BUFFER_POOL_KEEP
- BUFFER_POOL_RECYCLE
- **MONITOR TASK:** query on page 322
- Allocate segments to pools in BUFFER_POOL_KEEP, region of CREATE TABLE/INDEX: **FORCDF:** CALIBRUNLISTS
- **MONITOR TASK:** v$buffer_pool_statistics

4.1.3 Use process slaves to reduce i/o bottlenecks during high database activity

- _IO_SLAVES_DISABLED=TRUE (oracle support)

- if no asynchronous i/o usually, but can do if therse is AIO.

DBWR	DBWR_IO_SLAVES
	DB_WRITER_PROCESSES
LGWR	LGWR_IO_SLAVES
ARCH	ARCH_IO_SLAVES
Backup on disk	BACKUP_DISK_IO_SLAVES
Backup to tape	BACKUP_TAPE_IO_SLAVES

- **RANDOM NEURON FIRED:** SGA is System Gloabal Area and PGA is Program Global Area SGA has buffer cache, dictionary cache, redo log, SQL pool
- **MONITOR TASK:** check if server processe are waiting on background processes v$waitstat , v$session_wait, v$sytem_event, v$session_wait. page 325 lists events
- **MONITOR TASK: PARAMETER CONSTRAINT:** enable initialization parameter TIMED_STATISTICS
- **MONITOR TASK: SCALE Reference:** if these are hundreds of seconds, parallelize
- watch out for slave contention

Enable Vector Posting

- Posting is the term by which processes receive information.
- DBWR may be posted by LGWR to write dirty buffers
- pg 327 for info

Set up Dedicated temporary tablespaces

- **PARAMETER CONSTRAINT:** written in SORT_AREA_SIZE units
- for use in **IMPACTS API DESIGN: SQL IMPACT:** ORDER_BY, INDEX CREATION, uses INITITAL/NEXT in tablespace
- may be permanent or temporary: CREATE TABLEPACE ... TEMPORARY

4.1.4 Set up sort areas effectively

- **FORCDF:** for data warehouse
- **PARAMETER CONSTRAINT: SCALE Reference:** in PGA SORT_ARE_SIZE $\geq$ 2 datablocks. Default is 64K if MTS is used. SORT_AREA_RETAINED_SIZE
- **SQL IMPACT:** used in SQL requiring sorts
- uses temporary tablespace: **ORACLE OS Parallel:** equivalent of a page fault.

- **FORCDF: PARAMETER CONSTRAINT:** SORT_ARE_SIZE = $n\times$DB_BLOCK_SIZE
- can set according to a session. No bounce required BUT puts a gun in the hand of a user **IMPACTS API DESIGN:**
- **SCALE Reference:** High value of SORT_ARE_SIZE is 5MB
- **MONITOR TASK:** observe sort area size and temporary segment usage
- page 330, v$sort_segment, v$dba_segments v$sort_usage
- **PARAMETER CONSTRAINT:** set SORT_AREA_RETAINED_SISE = SORT_AREA_SIZE reducing overhead of dynamic memory allocation
- **TIP: UPGRADE HINT:** set high during an upgrade or migration
- ALTER SYSTEM works
- **PARAMETER**
  CONSTRAINT: SCALE Reference: : SORT_READ_FAC=(Average Latency + Average Seek Time + Block Transfer Time)/(Block Transfer Time)
- MTS: SORT_MTS_BUFFER_FOR_FETCH_SIZE now moved to _SHARED_SESSION_SORT_FETCH_BUFFER
- If you need space on disk, SORT_SPACEMAP_SIZE × (total bytes to be sorted/SORT_AREA_SIZE)
- In index creation: **SQL IMPACT:IMPACTS API DESIGN:** SELECT DISTINCT, ORDER BY, GROUP BY

Use Direct Writes for Sorts

ACTION:FORCDF:

- Sorts going to temporary tablespace pass through the cache
- bypass DBWR
- **PARAMETER CONSTRAINT:** SORT_DIRECT_WRITES=TRUE

Use profiles to protect against runaway processes

Resource limits assigned to users at per call or per session level: **FORCDF:** we can set this for the prd int.

- memory
- cpu
- i/o

- idle time
- connect time
- open sessions

FORCDF: In the development cycle, the user has frozen code on prd which can be used for read. tests can be built against the frozen release to write to the text database. Once new constants are decided upon, the user logs into the reserved and runs the values into there. This can be done on a text to oracle transfer if the job would be too involved. **ACTION:** We need tools to do this conversion.

If the user must add or modify tables, then the cut through int/prd must be done, but again, tests are done by reading production and writing to text. new items in text can also be read.

ACTION: FORCDF: IMPACTS API DESIGN: The API and tables need component testing.

Vectorized Reads

- essentially direct i/o to oracle buffer
- enable readv(): good for warehouse, 5% effect **PARAMETER CONSTRAINT:** USE_READV

Enable direct i/o for cached

- get benefit of going raw without going raw
- solaris 7 allows it
- large sorts, backup and restore benefit
- **PARAMETER CONSTRAINT: SCALE Reference:** DB_FILE_DIRECT_IO_COUNT
- **PARAMETER CONSTRAINT: SCALE Reference:** DB_FILE_DIRECT_IO_COUNT × DB_BLOCK_SIZE set to max i/o size (probably 64K)
- should be a multiple of the OS logical block size: **PARAMETER CONSTRAINT:** DB_BLOCK_SIZE= N × OS block size
- **PARAMETER CONSTRAINT:** for raid, should be multiple of volume strip size and largest of all strip sizes

Use initmate shared mory for high concurrent user loads

PARAMETER CONSTRAINT: The parameter is USE_ISM (if system kernel was rebuilt) and is true for Solaris == but what about linux?

Use post-wait drivers for high IPC throughput

- solaris is very good with semaphores
- may be relevent for linux
- **MONITOR TASK:** Test if system calls in user and kernel mode are excessive.

Prepage and lock SGA in main memory

- Oracle builds up memory pages in the SGA
- **PARAMETER CONSTRAINT:** PRE_PAGE_SGA: locks into memory
- **PARAMETER CONSTRAINT:** LOCK_SGA: prevents SGA from being sapped out
- **PARAMETER CONSTRAINT:** LOCK_SGA_AREAS: prevents certain areas from being swapped out
- **MONITOR TASK:** v$sga gives value

Set COMPATIBLE to the current database version

- can downgrade ot a prior instance
- **FORCDF:**Should we use this to install, keep 8.0.5 and bounce
- Must do a normal immediate shutdown

Other tips

- Always install SQL PLUS help
- **PARAMETER CONSTRAINT: MONITOR TASK:** System tablespace freespace $>$ 40% free; use monitor; designer uses the system tablespace. **HUH?** why is designer on the development machine
- **FORCDF: ACTION:** ensure disk volumes have growth space.
- Disk space should be 10% larger for static tables.
- **PARAMETER CONSTRAINT:** set AUTOEXTEND for data files belonging to application tablespaces
- **PARAMETER CONSTRAINT: FORCDF:** list static init params. Need a DB bounce for this. set distributed transactions > 0. RECO constraint reconnects to database after some interval
- Do not set specific EVENTS: may be able to do this without a bounce
- Ensure all unneeded options are disabled **ACTION:** make a list: see page 343.
- **COST:** Ensure you have gold or silver contract: a 24×7cost.
- **COST: FORCDF:** consider 3rd party contract of 10h/wk

4.1.5 SQL*NET/NET8 Configuration

Oracle Names

- mutiple names servers
- keep names in oracle table
- need failover: copy of tnsnames.ora

Out of band breaks w sqlnet or set pull frag high when out of bounds breaks not available

- this is control c. in bound breaks poll.
- **PARAMETER CONSTRAINT:** sqlnet.ora has BREAK_PULL_SKIP to est interval. reduces network load.
- **HUH?** does this imply any connect is a load
- If adhoc queries are made often, need a short polling interval. Need to keep users away!
- try 10^8,10^7,10^6,10^5,etc.
- set out of bound breaks are asynchronous, but takes room in TCP/IP packet **FORCDF:** this should be miniscule
- **ACTION:** Do we use out of bound breaks **PARAMETER CONSTRAINT:** DISABLE_OOB_OFF in sqlnet.ora to enable OOB.

Keep dead Connections to a minimum

- DCD: dead connection detection may also contribute
- Timer indexed probe
- Sends packet from server to client periodically
- **SCALE Reference: PARAMETER CONSTRAINT: MONITOR TASK:** SQLNET.EXPIRE_TIME 10 minutes between probes, but under high traffic, use 30.
- **MONITOR TASK: FORCDF:** monitor use

Manage large user populations with sqlnet features

- MTS: (MultiThreaded Server) large volume, each session in for a short period of time
- Predetermined upper limit of dependent shared servers with threads. Saves startup of processes

- **PARAMETER CONSTRAINT: FORCDF:**MTS_MAX_SERVERS important for offline and data warehouses
- **PARAMETER CONSTRAINT:** increase SHARED_POOL_SIZE: 1KB for each concurrent session **FORCDF:** max determined by license, so how do we compare
- Additional space needed for shared DATA
- If a large batch request where server is needed for a long time, use a dedicated connection: MTS will be looking to see if there are requests if the large batch is there, but cannot fulfill.
- **DEVELOPMENT TASK: PARAMETER CONSTRAINT:** Can configure two service names for one database: SNR = DEDICATED in tnsnames.ora
- **ACTION:** core DBA activities need a dedicated server
- **PARAMETER CONSTRAINT:** Prespawn dispatches (**FORCDF:**at begin run) MTS_DISPATCHES = CEIL(max number concurrent/n) where n is platform dependent
- **PARAMETER CONSTRAINT:** set MTS_MAX_DISPATCHES to be high
- can use ALTER SYSTEM
- **MONITOR TASK:** vmts, vdispatcher, v$shared_server, v$cirucit

Connection pooling for DSS applications with large users

- **FORCDF: DEVELOPMENT TASK:** for offline machine, warehouse
- allows physical connections to be shared.
- after timeout, physical connection given to another
- **PARAMETER CONSTRAINT:** MTS_DISPATCHES

Connection concentration

- **FORCDF:**online
- Lots of simultaneous active connections with small connect time
- use in addition to MTS
- funnel through some physics transport to a shared server
- 1000's of concurrent users: not in our license
- **PARAMETER CONSTRAINT: DEVELOPMENT TASK:** MTS_DISPATCHES = MULTIPLEX

Multiple Listener for listener failover and load balancing

- Each listener configuration or same server on different LOCAL_LISTENER
- **FORCDF:** perhaps needed online? offline?
- listener.ora: two listeners on two ports for one oracle server
- randomize client connections (tnsnames.ora0)
- AUTOMATIC is preferred, MANUAL is available.

Prespawn dedicated server processes

- **PARAMETER CONSTRAINT: DEVELOPMENT TASK:** listener.ora: PRESPAWN_MAX, PRESPAWN_LIST see page 359
- **HUH?** how do to spawn: send command?
- **FORCDF:** possible application for cdf online at start of run or offline for farm start of run

4.2 Chapter 8: Upgrades, Downgrades, Reorganizations and Migrations

- Upgrade: database version change
- Downgrade: third party code does not work
- Reorganization: need to defragment tables
- Migration: need better/bigger machine **FORCDF:** online dev, offline everything now needs this

4.2.1 When to Upgrade

- Scalability: when you outgrow
  - CPU (4 to 8 cpu)
  - OS
  - oracle release
- knowledge base: checked with others about export
- Oracle support tells you
- Testing
- **SCALE Reference:** Upgrade > 6 mo after release but within a year
- **SCALE Reference:** Major migration takes two hours

4.2.2 How to Upgrade

Practice in a test release

- **INT SAME SIZE AS PRD:** should mirror production
- **INT SAME SIZE AS PRD:** Allows you to predict time for the upgrade
- **INT SAME SIZE AS PRD:** Removes surprises
- **INT SAME SIZE AS PRD:** test new environment

Analyze each step

- Determine unit of migration: software, redo log, data, etc.
- Utilities exist for each release so may not have to do an unload/reload
- **ACTION:**make this into a checklist!

Steps to Upgrade

- Unload data from source database
- loading data to target: need to check if self-referential integrity is present. This is hard.
- Need to pick a period, a "delta"
- Need then to migrate new rows entered after the first delta was set
- Restrict outage to last bit of data
- Do during low activity
- A timestamp is helpful in the tables **ACTION: FORCDF:**survey this: **NEEDED FOR REPLICATION:** CREATE TABLE AS SELECT * FROM x WHERE CREATE_DATE < d
- If no timestamp, at least have a sequential key (**FORCDF:**CID)
- May need to do in chunks:
  - to reduce load on running system
  - to fit into offpeak number of units
  - number of rows
  - physical resources
  - activity on database
- **MONITOR TASK:** need monitors to determine how to go on this decision.
- During outage the following is done:

- DROP a
- CREATE TABLE a AS SELECT * FROM b WITH CREATE_DATE $\geq$ d

- apply referential integrity check: hard to do.
- **FORCDF:**we can write and test migration scripts for used and valid sets. CALIBRUNLISTS is the main table.
- **FORCDF:DEVELOPMENT TASK:** can we get generic transport scripts?

Transform data, if needed, at source, target or combination of both

- **FORCDF:**can do this during runing if needed – but slow control may be a problem since it sends errors randomly
- **FORCDF:DEVELOPMENT TASK: NEEDED FOR REPLICATION:** The scripts for this are similar to the replication scripts to MSQL

Copy from source to target

- copy to allow fallback
- speed depends on bandwidth
- can introduce boolean columns to indicate the new migration.
- Bit Map index the rows.
- Need to reset after migration

Load Data into Target DB

- Need to consider resource contention
- **FORCDF:**This should be fine if CDF is designed for large spikes and the spikes come at well defined times or can be scheduled.
- load with unrecoverable/nologging

Post Migration Testing

- Do on core tables
- Random Sample

Summary of Steps

- Determine unit of migration
- Unload data from source
- Transform data at source, target or both
- Copy data from source to target
- Load data into target
- Validate data

4.2.3 Example: Gotcha's from Oracle 7 to 8 Migration

While CDF is doing all Oracle 8, this is included as an example of the kinds of things that need to be done when CDF is faced with the next migration. **ACTION:** make a checklist of the following subsections.

4.2.4 Before

- Get bug checklist from Oracle Support and compare to our needs
- Perform hot (or cold, if possible) backup
- Prepare scripts to check for invalid constraints
- check that special users needed by oracle don't already exist **FORCDF:**take care with user names: perhaps CDF_xxx
- ensure rollack segment not to hit MAXENTENTS
- **NEEDED FOR REPLICATION:** ensure advanced replication is off
- ensure no pending transactions
- check free space on SYSTEM tablespace
- check free space where control files are maintained
- check free space for data
- delete obsolete params in old init.ora
- list heavily used tables (for later reference)
- list applications depending on row id (this is 7 to 8 specific but there can be items that need to fall under the category of "extra migration-specific checks)
- make all tablespaces offline

4.2.5 During

- Spool SQL to a log file and check: grep ORA
- Do not access database when migration runs

4.2.6 After

- Put tablespaces online
- Search for invalid constraints (need scripts for this)
- Search for invalid stored procedures, functions, packages, views and triggers.
- Compile stored procedures, functions, packages, views and triggers.
- Recompute all statistics using the cost based optimizer
- Check for indexes with STATUS=UNUSABLE in DBA_INDEXES
- recreate invalid indexes
- perform a full table scan to /dev/null (specific to oracle 7/8)
- bounce database to clear buffers
- validate data

Tips to Enhance Load/Unload Performance

- Perform loads without indexes and with constraints disabled.
- Rebuild indexes afterwards to prevent fragmentation
- Data are loaded, sorted, build indexes. Use NOSORT: this will sort if the table is not sorted.
- parallelize index creation
- build indexes UNRECOVERABLE/NOLOGGING
- under unix, use named pipes: see page 382
- Use single task versions (need to link them!) of export and import, but ask Oracle Support if ok. A 0% to 20% gain in time.
- Use OCI to write to flat files and SQL Loader
- Parallelize with PQO and start parallel instance of load and unload
- Spread out output across disks/controllers
- parallel CREATE TABLE AS SELECT in UNRECOVERABLE/NOLOGGING for data sets

- **NEEDED FOR REPLICATION:** Use Simle Replications as a tool
- Avoid additional layers: ODBC, SQLNET, etc.
- For large Databases, directly copy data files to target system:
  - if NO reorganization and no major schema differences:
    * same directory structure
    * same environment variables
    * use latest hot backup and init file
    * create new control files
  - bring new database into existence
    1. Modify init files
    2. start as NOMOUNT
    3. Run control create script
    4. Open Database
    5. Create Redo
    6. Backup
    7. Shut down and restart mounted
    8. Copy source redo logs and apply
    9. Take 15 min delay or tag or use a CSD
    10. apply last set of redo logs
    11. open for business
- If you use a triple mirror you can break the mirror, move the disk to another system, resilver and direct there. Useful for a VLDB.

4.2.7 Case Study

Conditions of Study

- Sun 3000 to 6000 migration
- Required < 4 hr downtime
- Old system had 8 cpus and 1GB memory on Raid 5.
- New system had 12 cpus and 2 GB memory on Raid S and Raid 1 with Veritas
- Database had 120GB size with 80 Gb data, 56 tables, 4 large, all others < 100K
- The team consisted of a DBA, consultant, technical architect network adminstrator, two system administrators and a boss phoning to interrupt progress.
- Basic strategy was to migrate then defragment

Broad Strategy

- Enhance network between machines
- make database on target
- no indexes
- no constraints
- Establish delta via a logical barrier
- Migrate 4 tables to get rid of fragmentation
- Create/Enable constraints
- Rebuld Indexes and Enable referential integrity
- Disable application access
- validate with raw counts
- restart

Before

1. Upgraded network to 100 MBps FDDI
2. Created Database as per plan: 120 GB on 18 disks
3. Datafiles with intensive writes on Raid 1
4. REDO on Raid 1
5. Tablespaces and user id's created
6. Make table create scripts
7. grant roles and profiles
8. write code for new columns
9. write code for triggers
10. take a 5 minute outage to alter the old instance and add the column. Done at 2am on Sunday, announced to users 1 week ahead
11. migrate old data
    - used pCTAS in UNRECOVERABLE/NOLOGGING with same name in a diffferently named schema (user)
    - used named pipes to export. Took 6 hours
    - dropped temporary tables

- created indexes: presplit them with PCTFREE=40
- all constraints BUT foreign keys created

12. Scripts to capture rest of data
13. Scripts to import for 4 tables, parallelized. Utilized from user to user. Used another user for the other 52 tables Used named pipes in direct mode.
14. wrote scripts for remaining 52. Used UNRECOVERABLE/NOLOGGING mode, separate scripts for index creation to parallelize, multiple indexes were created sequentially. Analysis of the parallelism is complex.
15. Scripts written for foreign key enable
16. tnsnames.ora modified

During

- started at 23:55 Saturday night
- announced the outage at same time (2 weeks earlier) as first outage announcement
- Put up a notice to users during outage
- Ran 4 table scripts concurrently
- Ran 5 export/import scripts for rest of tables for 4 big and rest
- network was the bottleneck
- Index creation of 52 tables
- Foreign key runs
- Test all structures ok, no INVALID INDEXES
- tnsnames.ora updated
- applications restarted

After

- test access from Web **FORCDF:**AC++ and browser
- benchmark performance
- tested every 5-10 hours for 48 hours
- complete backup of database

4.2.8 Observations

- Tested on Sun SPARC 1000
- Three tests were done
- managed to complete in 2 hours
- Custom Standby Databases are expensive: would cut downtime though.

4.3 Chapter 9: Backup, Recovery and Archiving

Select two backup methods and ask

- need shutdown?
- what is the time window?
- SLA specific: parallel restore?
- Flexibility:
  - last committed
  - lost data allowed?
  - to a past point in time
  - loss of data file, redo log control flie
  - loss of table or index
  - hardare and media required
  - customization allowed
  - auto backup allowed **FORCDF:**problem in conflict of resources online: start of backup at begin/end run?
  - Able to use API (ie. OEM)
  - support scheduling
  - portable across machines (platforms)
  - easy to use
  - reporting functionality

Oracle supports:

- backup sets
- datafile copy
- OS hot
- OS cold (not 24×7)
- logical and export

4.3.1 Tips

Perform Hot Backups during low DML activity

- _LOG_BLOCKS_DURING_BACKUP is set true to overcome split or fractured blocks but this results in a significant redo log.
- **FORCDF: ACTION:DEVELOPMENT TASK:** test impact of backup running during various phases. How do we control this?

Do NOT Place all tablespaces in hot backup mode simultaneously

FORCDF: what do we do?

- a fast checkpoint is done to flush dirty buffers
- check point SCN determines what the redo "START" is (SCN= System Change Number)
- fast checkpoint: DBWR continously scans dirty buffers. **PARAMETER CONSTRAINT:** Otherwise just clears at DB_CHECKPOINT_WRITE DB_CHECKPOINT_BATCH
- **PARAMETER CONSTRAINT:MONITOR TASK:** LOG_CHECKPOINTS_TO_ALERT will allow a look at checkpointing

Advice on Tuning Backup Performance

- **SQL IMPACT:IMPACTS API DESIGN:** Fix the SQL to ensure efficient
- spread out redo log, data file on different controllers
- used EDU (Enterprise Backup Utitity) but RMAN is better
- get 15 hours down to 6 hours
- Need to consider
  - Outage time
  - Tolerable Down time
  - Backup timings
  - Capacity available
- Back up to disk then tape. Tape CAN do 12 MB/Sec **FORCDF:** – do we do this for offline?
- Most recent on disk reduces recovery time
- **FORCDF:NEEDED FOR REPLICATION:** with Fermi Redhat, we can ensure params match server distribtuion
- For disk backup, ensure

  - space (consider compression usage)
  - use different disks/controllers
  - **MONITOR TASK:** write to a well-striped set of disks: No I/O hotspots
  - write to tape before overwriting on disk
  - keep two days of backups on disk so if a disk fails you still have the other backup
  - only go to tape if not enough disk or if a tiny database

Do not back up online redo logs during hot backups

- redo logs only during initial recover: need current log. Have end of redo thread
- **SCALE Reference:** fairly small database: 24GB, 21 GB data
- **FORCDF:INT SAME SIZE AS PRD:** bring up instance on integration after a crash
- **FORCDF:** we drive people nuts on which database they should use at any given time.

Evaluate the use of triple mirrors

DEVELOPMENT TASK: check this

- Triple mirrors mirror each primary disk to two other drives
- Hardware or a combination of hardware and software
- Allow isolation during backup
  - All tablespaces are (can be) put into simultaneous backup
  - third mirror broken
  - all tablespaces taken out of hot backup
  - third mirror used to backup to tape
  - resynchronize 3rd backup
- Short time, all tablespaces can hit the redo log
- resilvering can eat some time - less so if hardware based
- can let 3rd mirror lag 1/2 hour behind to allow user error recovery
- during resilvering, there is no backup
- can go to a 4th set with two sets lagging by a few hours

Make sure hot backup commands are executed synchronously

- use cron
- check the are being done
- **FORCDF:** ace training and usage?

Check Backup Log Regularly

- **MONITOR TASK:**grep and email
- **ACTION:DEVELOPMENT TASK:**periodic review of backup tape to ensure it works

Avoid manual backups

- **DEVELOPMENT TASK:** use database to store inforamtion on backup
- **MONITOR TASK:**v$datafile v$dba_data_files
- **DEVELOPMENT TASK:**test impact of backup: what can we tolerate?

Ensure security is not compromised

- **FORCDF:** no passwords in scripts or use of default accounts from 3rd party

RMAN:resychronize recovery catalog from time to time

HUH?

- update with information in redo logs
- do as one of the last steps of backup

Make sure you get all Oracle file types

ACTION:

- make list
  - data files
  - online redo logs
  - control files
  - archived redo
  - init params
  - trace files
  - software
- provide backup at different points in time
- automate update to web
- list tablespaces and datafiles

4.3.2 Oracle File Types

- Tablespace/Datafiles
  - **FORCDF:** Back up files in one tablespace as logical unit
  - Prioritize tablespace:
    * system critical
    * may wish to mirror and not stripe due to critical nature
- Online Redo Logs
  - used in instance and media recovery
  - rollback as well as rolling forward
  - use multiple logs
    * **FORCDF:**each on separate disk/controller
    * **FORCDF:**stripping not sensible for writing sequential but can help reduce hot spots
  - **MONITOR TASK:** monitor for stale/invalid before, during, after backup
  - STALE: write pending to log but shodtown abort
  - INVALID: hardware error
  - **MONITOR TASK:**stale/invalid for backup in v$logfile
- Control Files
  - physics database structure information
  - crucial in recovery
  - keep $\geq$3 copies **FORCDF:**online/offline
  - use mirroring (stripe: no performance, too small, decreases MTBF)
- Archived Redo Logs
  - backup every four hours to tape
  - **HUH?** Don't these need to be tuned with backup interval. Otherwise need to restore from tape
  - **ACTION:DEVELOPMENT TASK:**need recovery after we have a valid data checker
  - The archive log space needs to be considered with the peak loads. **ACTION:**simulate loads
  - **NEEDED FOR REPLICATION:** replication can change our requirements
- Initialization and Configuration files
  - init.ora config.ora
  - **ACTION:**list and put into CVS

- Alert Log and relevent trace files (also Section 6.1)
  - cron to slice up and archive
  - keep 2 weeks on disk
- Oracle Software
  - at install
  - after upgrades/patches
  - **ACTION:** audit
- **ACTION:FORCDF:** To summarize: make table 9-2 for cdf

Backup control files after all database changes

- New table space, drop tablespace, recreate control file do:
  - ALTER DATABASE BACKUP CONTROLFILE TO TRACE
  - timestamp or check into CVS
  - Keep record of changes on web
- Take control file backup each morning for pilot error coverage
- Take a binary image:ALTER DATABASE BACKUP CONTROLFILE TO ...
- backup for each readonly to read/write switch and back **ACTION:FORCDF:** get a form for this action
- scenario where readonly but recovery REQUIRES a write!
- recovery via a control file is a last resort

Address export-related concerns before doing exports

- Export requires knowlege of inter-table dependencies
- Redo logs may not be large enough for consistency

Form export groups for intersegment consistency

ACTION: Need instructions by schema controller

- Identify key tables **FORCDF:** CALIBRUNLISTS
- identify interdependency
- **PARAMETER CONSTRAINT:**form export groups and export together with CONSISTENT=Y
- Export during low DML activity

- Import entire group
- Truncate tables about to import
- Enable large rollback segments during import

Plan space for export dumps

COST:FORCDF:ACTION: Estimate extra size due to this

- Keep latest export file on disk
- Keep date/timestamp in filename

Perform exports with direct path

ACTION:FORCDF:do we do this

- 70% faster
- avoids conversions

Ensure compatibility and usability

- downgrade issue

Perform backup after any system clock change

- allows point in time recovery
- **FORCDF:ACTION:** must schedule for cdf
- **FORCDF:** Don't do big development changes over daylight savings time changes

Check for corruptions during/after backups

- **MONITOR TASK:** RMAN populates v$backup_corruption and v$copy_corruption
- **MONITOR TASK:**need to use db_verify
- exports will see corruption because of full table scans

Setting Optimal I/O Size during backups

- **PARAMETER CONSTRAINT:**DISK_IO_SIZE
- **PARAMETER CONSTRAINT:**TAPE_IO_SIZE
- **PARAMETER CONSTRAINT:HUH?**set bs

make use of incremental backups

FORCDF:do we do this

- RMAN
- configure "backup sets"
- **ACTION:**set different levels and assign priorities
- cumulative vs non-cumulative
- **IMPACTS TABLE DESIGN:ACTION:FORCDF:**could be important for large database where many table entries are static.

4.4 Archiving

4.4.1 Evaluate whether ARCHIVELOG is necessary

COST:"In any 24×7 it is absolutely necessary" but this is part of the 90/10 rule.

- Evaluate advantages:
  - complete media recovery
  - can back up hot
  - allows inconsistent backup: smaller units
  - allows distributed recovery
  - **FORCDF:** non-SYSTEM tablespace can be taken offline IMMEDIATE mode: as the UPS will do for us
- Evaluate disadvantages:
  - administrative overload
  - work: get to tape and testing.
  - storage: redo records contain vital change vector:
    * dba
    * class
    * op
    * seq
    * scn
  - **FORCDF:MONITOR TASK:** if space fills, database freezes.
  - bookkeeping
  - ARCH must run adding to overhead
  - ARCH Must be posted (told to write)
    * log switch
    * timeout (3003)

    * archive start
    * database open
    * self check
    * LGWR
  - data loss and recovery time considedration
  - **FORCDF:**INFO messages in Control room that are "fail" but marked as INFO

4.4.2 Tips

Implement Auto Archive

- Use manual only in emergency (disk full)
- Use manual for redo log

Allocate enough space for ARCHIVE_LOG_DEST

- this would freeze the database
- use enough for 2 backup cycles

use ALTER DATABASE ACRHIVELOG for enabling/disabling

- **ACTION:DEVELOPMENT TASK:FORCDF:SCALE Reference:**Takes longer to create database if at create time. For example, 16 tablespaces, sun 4000, 1.5Gb, Raid S(EMC) 40 GB took 8 hr with ARCHIVELOG: hot spot in CREATE DATABASE and CREATE TABLE.took 6h 45m with no archive log, a 14% savings. The ALTER DATABASE ARCHIVELOG took a few seconds.
- **MONITOR TASK:**Check it is running with ps
- SVRMGR > ARCHIVE LOG LIST

Take a complete backup as soon as ARCHIVELOG mode is enabled

- prior to batch load, may disable
- disabling archive zeroes out SCN
- marks split in coherence of log numbers
- even in DSS, the assumptions of loadable or UNRECOVERABLE/NOLOGGING may be violated

Use ARCHIVE_LOG_CURRENT

- **FORCDF:**for manual logging

Do not create holes in the log sequence

- will be tempted to do this when facing a full ARCHIV_LOG_DESTINATION emergency.
- **FORCDF:ACTION:**add ARCHIV_LOG_DESTINATION emergency to emergency book
- options
  1. Redirect Archive: have some spare disk. Need space ALTER SYSTEM ARCHIVE LOG TO <> can CRON: partition logging or waring at 50%
  2. Move already archived logs: DBA disable scratch?
  3. Selectively delete older logs: ls -ltr

If Possible Keep Dual Copies

- **COST:PARAMETER CONSTRAINT:**LOG_ARHCIVE_DUPLEX_DEST
- Impact on performance

Create more online redo log groups when archiving to offline

- **COST:**if have to go to tape
- **FORCDF:**verify that here is disk archive

Be aware of special considerations for OPS

Unlikely to use this so must see book.

4.5 Recovery

4.5.1 Types of Recovery

Block Recovery(Automatic)

When server process terminates prematurely, rollback is done and locks released. Why terminate:

- Hardware
- Manual: kill
- statement failures:
  - tablespace quota
  - unavailable free space
  - privileges
  - constraint violation

- MAXEXTENT

- Distributed lock problem
- thread recover: automatic after an instance crash. Does shutdown abort and kills processes
- **INT SAME SIZE AS PRD:ACTION:**need an emergency book and emergency training

Media recovery

This requires a DBA

- file damage
- **FORCDF:ACTION:**evaluate potential points of failure

4.5.2 Factors affecting recovery time

COST:

- need to restore to disk from tape: this is the most time
- redo log reads (CPU intensive)
- datafile blocks read by DBWR in memory go to disk: uses recovery buffer
- OPS must merge threads
- **ACTION:**need to ensure architecture of redo log placement allows parallel controller use
- Datafiles can be put into separate groups
- size of buffer cache: if bigger, allows faster recovery
- DBWR throughput: **MONITOR TASK:** observe v$sysstat for
  - free buffers requested: how often server kicks DBWR
  - free buffers inspected: how busy DBWR is
  - dirty buffers requested: DBWR needed to write to disk
  - DBWR make free requests
  - etc. pages 464-465

Ensure recovery times consistent with SLA MTTR

- **ACTION:**time list on page 468
- **ACTION:INT SAME SIZE AS PRD:**drills to set those times and support SLA

Consider Alternative backup/recovery approaches for point in time recovery

- Export
- Parallel table creation
- SQL Loader and flat file **ACTION:**What about 8.1.6 for N years!
- **NEEDED FOR REPLICATION:**simple snapshots: for large tables with incremental changes
- **NEEDED FOR REPLICATION:**copy of table in failover instance: Advanced replication
- **IMPACTS TABLE DESIGN:FORCDF:**examine in light of our tables.

Ensure check points often enough

"Performance vs availability"

- **PARAMETER CONSTRAINT:**LOG_CHECKPOINT_INTERVAL
- **PARAMETER CONSTRAINT:**LOG_CHECKPOINT_TIMEOUT: if disabled, checkpoint occurs at redo switch
- **PARAMETER CONSTRAINT:SCALE Reference:**every 30 minutes: **FORCDF:**after a calibration run or any run end **HUH?**Can we force a checkpoint?
- **DEVELOPMENT TASK:INT SAME SIZE AS PRD:**Test this on a different machine that is similar or less power but not too dissimilar
- **DEVELOPMENT TASK:ACTION:INT SAME SIZE AS PRD:**Evaluate what happens when the following drill is conducted
  - two online redo logs
  - cold backup
  - force redo switching: 1M rows pg 472
  - delete a data file
  - other drills: pg 471-473

Always consider lowest Unit of recovery

- levels are
  - database
  - tablespace (point in time)
  - data file.
- data file recovery can lead to logical corruption

Always examine v$ddatafile priory to recovery

- **MONITOR TASK:**see if online or offline
- when going offline NORMAL, check point is done
- if some inconsistent files, must do a TEMPORARY offline
- if files are inconsistent must do an IMMEDIATE offline
- A database crash after a repair is a disaster if the redo logs were used.

Maintain special init.ora for recovery

ACTION: FORCDF: Allows special parameter settings:

- DB_BLOCK_BUFFERS: high
- SHARED_POOL_SIZE:low
- Asynchronous I/O
- DBWR Params
- Parallel query option and related parameters: number of parallel servers = number of cpus
- MTS related parameters: disable:
  - SORT_AREA_SIZE
  - log buffers
- **TUNE:**need to optimize. Repeat when requirements change.

Always use RESETLOG as a last resort

- Except when using a hot backup: creates online redo logs
- Only if NORESETLOGS errors out
- standby databases need rebuild
- do backup if used: control file
- can use old backup and redo logs to get to point in time but error prone

Evaluate all UNRECOVERABLE/NOLOGGING

- index create
- NOLOGGING does no redo
- back up immediately
- only if a real performance gain

Categorize all segments for ercovery

FORCDF:IMPACTS TABLE DESIGN:ACTION:need to set this up

- may wish to export and back up
- categorize tablespaces and data files based on availability

Keep tablespaces without write activity READONLY

- **FORCDF:IMPACTS TABLE DESIGN:**evaluate the ones that are admin and really readonly
- expedites recovery
- Take backup if tablespace is made readonly
- Take backup if tablespace is made read/write

Set AUTORECOVERY on during recovery

4.5.3 Advice to managers

- Test backups
- Ensure staff training
- Support contract with senior dba
- Form database recovery team with
  - DBA
  - sysadmin
  - developers
- Document all database structural changes

4.6 Chapter 10: Startup and Shutdown Procedures

RANDOM NEURON FIRED: Do we consider standby for online is offline? **RANDOM NEURON FIRED:**How is this limited by kerberos?

4.6.1 Controlling Database Shutdown/Startup

Always Automate Shutdown and Startup

- Allows sysadmin to do the startup/shutdown
- SQLDBA or SeverManager
- sqldba svrmgr,dbstart,dbshut, pit in /etc/rc

- Oracle should be first to start and last to shut down
- listener should start after the database and shut down before
- OEM can be used
- DatabaseManager is a GUI for these operations
- InstanceManager

OEM Useful for Multiple Databases

RANDOM NEURON FIRED: How is this limited by kerberos **HUH?**The init.ora file is on the OEM machine: how does this impact backup

Urgent:Database SHUTDOWN ABORT

- **FORCDF:** user logged in but walked away stops a normal shutdown: we should consider timeouts (Section 5.1.9)
- SHUTDOWN IMMEDIATE is not quick: **MONITOR TASK:** must examine
  - v$session
  - v$lock
  - v$sqlarea
- **ACTION:FORCDF:**Need shutdown procedures describing what to do in what circumstances
- **ACTION:FORCDF:**Need to drill shutdowns:see test on page 503
- After shutdown abort, must perform normal shutdown after it is started up again

Just Prior to SHUTDOWN ABORT do ALTER SYSTEM CHECK POINT

- **FORCDF:**note that UPS is SHUTDOWN ABORT so add this to the procedure on UPS. **INT SAME SIZE AS PRD:**Need to test with simulation of full database
- This results in less online redo.
- **ACTION:FORCDF:**Exercise page 505
- **SCALE Reference:**Time for shutdowns:
  - Normal: 16 min
  - Abort: 6 min
  - with checkpoint: 40 s

Chapter 5

Maintenance and Disk Space management

5.1 Chapter 11: General Maintenance

- patch/version upgrades
- init param changes
- defragmentation
- index rebuild
- stats
- corruption check/repair

Downtime is inevitible: **FORCDF: DEVELOPMENT TASK:** need standby for true 24×7.

5.1.1 Types of Maintenance

Normal

- DB security
- **MONITOR TASK:**Freespace check and modify
- **MONITOR TASK:**monitor performance and change init parameters
- **FORCDF:**consider cost based optimizer use

Execeptional

- recovery
- defragmentation/reorganization

- corruption and repair
- upgrade

5.1.2 Implement Robust Security at all Levels

- Bad security fosters:
  - accidents: trip over cables, new users
  - malicious attack
- **FORCDF:**user error is the largest percentage: consider delayed entry of values (Section 4.5.2).
- **FORCDF:**do regular security audits: define, formalize, schedule
- security on OS critical files **FORCDF:**we restrict login accounts
- End user identification essential **FORCDF:**need to consider user column in CalibDBTables

Types of security are:

Type of security: Trust Relationships

- directly on server
- other nodes of cluster
- **DEVELOPMENT TASK:FORCDF:**subnets
- cryptography for snooping

Type of security: Password based

FORCDF:HUH?how much of this is implemented?

- length
- alphanumeric
- lockup if too many tries
- no old password repeats allowd
- can choose whom to enforce this on **FORCDF:**our specific general accounts
- **DEVELOPMENT TASK:FORCDF:**encryption: pg 516, is this in place and does it interact with kerberos

Type of security: Other

- External Authentification
- Advanced Authentification: Kerberos as well
- Oracle 8 Security Server

Security for Sessions filtered in a middle tier

- need to funnel many users to a few connections
- set up special circuit for these

Direct Privilege Management, Rule-Based Security Audit, Profile-Based Security

- Follow least privilege path
- Can audit roles by use **FORCDF:**We need not bother as we assign table to functionality and divide accounts according to functionality
- Can track suspicious acts **FORCDF:***This* is a security policy!
- Can get to OS users **FORCDF:**not useful for external authentification and accounts assigned by functionality
- **FORCDF:ACTION:**Disconnect after idle time (Section 5.1.9)
- **DEVELOPMENT TASK:**Can assign higher resources to users: Database Resource manager

5.1.3 Views, Stored procedures and triggers

- **IMPACTS TABLE DESIGN:**can restrict horizontal or vertical access: **FORCDF:**sort out in admin roles?
- **DEVELOPMENT TASK:IMPACTS TABLE DESIGN:**trigger on check user before write

5.1.4 Understand "ORA 1555 Snapshot too old"

- Database space management issue:
  - Sub optimal sizing of rollback segments
  - application management/scheduling
- Reads: "readers"
  - **SQL IMPACT:**SELECT
  - **SQL IMPACT:**SELECT FOR UPDATE

- **SQL IMPACT:**WHERE
  - **SQL IMPACT:**subqueries
- Writes "writers"
  - **SQL IMPACT:**INSERT
  - **SQL IMPACT:**UPDATE
  - **SQL IMPACT:**DELETE
- Transactions
  - readers and writers PLUS COMMIT
- Read consistency
  - when read starts, get the table at that time, even if 10 minutes later a change was made by another process
  - rollback segments used to keep multiple versions
- Snapshots
  - consistent across result set
  - nothing to do with replication
- Fetch across commits
  - Non Ansi: avoid
- Delayed Logging Block Cleanouts
  - Decide on when data go to disk
  - Gets marked in segment
  - But this will change and free the segment so a read will miss the cut
  - therefore overwrites the read and destroys the consistent snapshot
  - **FORCDF:IMPACTS TABLE DESIGN:**Which applications are likely to do this: CALIBRUNLISTS, data file catalog

5.1.5 Scenarios of Trouble

ACTION:FORCDF: Audit

1. Large number of transactions, inadequate rollback
2. Inadequate sized rollback segments
3. Most common: Intermingled large and small transactions **FORCDF:**offline
4. Delayed logging block cleanouts

5. **SQL IMPACT:IMPACTS API DESIGN:**Fetch across commits, especially if combined with delayed logging block cleanouts. The same session causes trouble. Frequent changes and commits in a loop can cause trouble.
6. **SQL IMPACT:**Inappropriate OPTIMAL clause: shrinks the rollback segment
7. Buffer cache too small (no need to go to rollback if big enough)
8. **HUH?**Discrete transactions being used
9. corruption
10. **HUH?**OS MAC Mode

5.1.6 Scenarios of Solutions

- Can increase the number of rollback segments
- Increase the size of each segment
- **SQL IMPACT:IMPACTS API DESIGN:**Reduce frequency of commits (requires larger rollback segments)
- most common : faulty architecture: OLTP and DSS mix **FORCDF:NEEDED FOR REPLICATION:**offline: datawarehouse and implication that we need replication
- OLTP and DSS should not live together
- scheduling is critical **FORCDF:**how to run online, offline scenarios, simluate **INT SAME SIZE AS PRD:**Need machine to check with same setup
- Alternatives to ensure data are not changed
  - **IMPACTS TABLE DESIGN:**lock tables in SHARED or EXCLUSIVE mode
  - **IMPACTS API DESIGN:**isolation level: serializable or read only. stops transactions that would change the same data
  - **IMPACTS API DESIGN:**reduce ranges of rows being accessed **RANDOM NEURON FIRED:**Partition tables for CID
  - **IMPACTS TABLE DESIGN:IMPACTS API DESIGN:SQL IMPACT:**for multiple users of a table use ORDER BY
- **IMPACTS API DESIGN:SQL IMPACT:**Force a cleanout before long transactions **FORCDF:**Need to consider this for the browser
  - **ACTION:IMPACTS API DESIGN:SQL IMPACT:**for index blocks: SELECT f FROM x WHERE gind > 0; page 532
  - **IMPACTS API DESIGN:**can disable and suffer performance

- **IMPACTS API DESIGN:SQL IMPACT:**Avoid fetch across commits – but need larger rollback segments. If it must be done:
  - ORDER (see 5.1.6)
  - set commits in multiples of "n" (see 533)
  - **ACTION:IMPACTS API DESIGN:**Another reason to add estimate of rows to API
- **ACTION:DEVELOPMENT TASK:IMPACTS TABLE DESIGN:**Study growth patterns and adjust optimal clause **MONITOR TASK:**monitor rollback segments extensively
- Oversize
- **HUH?**Don't use discrete transactions
- "See chapter 14"
- **PARAMETER CONSTRAINT:**set LOG_CHECKPOINT_INTERVAL lower

5.1.7 Understand, Prevent and Resolve Latch and Enqueue Contention

FORCDF:LINUX VS SUN for latches

Latches

- serialization of access to SGA done at a low level
- PMON cleans up latch usage
- latches are mutually exclusive
- Modes:
  - immediate: no wait; process continues if no latch available
  - willing to wait: sleeps/ retries/ polls/ sleeps ... polling set by **PARAMETER CONSTRAINT:**SPIN_COUNT
  - noqueues: only for multi cpu

5.1.8 Enqueues

- allow/manage concurrent access with sharing
- FIFO used
- modes: immediate and willing to wait
- **MONITOR TASK:**v$latchname
- **MONITOR TASK:**latch contention eats CPU

- **MONITOR TASK:**see list pg 537: want report to reflect IMGET, MIS provided on redo allocation
- **INT SAME SIZE AS PRD:ACTION:DEVELOPMENT TASK:**Want to force trouble to test
- **PARAMETER CONSTRAINT:**Parameters of interest
  - SPIN_COUNT
  - ENQUEUE_RESOURCES (number of locks)
  - DB_BLOCK_LRU_LATCHES
- **MONITOR TASK:DEVELOPMENT TASK:INT SAME SIZE AS PRD:**How much missing do we need before we lock cpu
- **INT SAME SIZE AS PRD:**One CPU is useless for testing
- Spin/Sleep cycling is used to reduce contention in
  - redo allocation and redo copy (Chapter 4.1)
  - **PARAMETER CONSTRAINT:**raw cache objects latch: increase SHARED_POOL
  - **PARAMETER CONSTRAINT:**Change buffers LUR chain latch: DB_BLOCK_BUFFERS, DB_BLOCK_LRU_LATCHES
  - **PARAMETER CONSTRAINT:SCALE Reference:**use one latch per 50 datablock buffers
  - **PARAMETER CONSTRAINT:**cache buffers chain latch: DB_BLOCK_BUFFERS

IMPACTS API DESIGN:SQL IMPACT: These are all symptoms: need to tune the API

5.1.9 Be Familiar with Different Wait Events

ACTION:MONITOR TASK: Order this analysis by ways to fix or parameters that need tuning

- **MONITOR TASK:**Can see events with:
  - v$session_wait: low level
  - v$session_event: session level
  - v$system_event: system level

  Need to see which sessions are waiting: **MONITOR TASK:**for control room **MONITOR TASK:**look at:

  - v$event_name: gives wait events
  - v$waitstat

- v$kcbfwait: gives per file waiting

- Wait Event:SQLNET Message from Client
  - most prominent
  - wait for client
  - not servers
  - reason:network clog

- Wait Event: SQLNET Mesage to client
  - converse of last
  - network clog

- Wait Event: WMON goes to sleep
  - just wakes up processes, no worry

- Wait Event: Buffer Busy Waits
  - **MONITOR TASK:**page 547 shows how to look at block wait
  - **IMPACTS TABLE DESIGN:ACTION:MONITOR TASK:**Identify files prone to wait
  - FREELISTS, FREELIST GROP, INITRANS, PCTUSED, PCTFREE, DB_BLOCK_SIZE

- Wait Event:Buffer Dead Lock
  - not often, hard to detect
  - related to Buffer Busy Waits 5.1.9
  - solution: see 5.1.9

- Wait Event: Checkpoint Completed
  - takes 5s
  - make checkpoints occur less frequently
  - **PARAMETER CONSTRAINT:**LOG_CHECKPOINT_INTERVAL
  - **PARAMETER CONSTRAINT:**LOG_CHECKPOINT_TIMEOUT
  - **PARAMETER CONSTRAINT:**DB_BLOCK_CHECKPOINT_BATCH: units of time limited by OS
  - **PARAMETER CONSTRAINT:SCALE Reference:**DB_BLOCK_CHECKPOINT_BATCH = 1/4 of DB_BLOCK_BUFFERS
  - **PARAMETER CONSTRAINT:SCALE Reference:**DB_BLOCK_CHECKPOINT_BATCH < _DB_BLOCK_WRITE_BATCH

- Wait Event: Control file related wait events
  - control file parallel write, sequential read, single write
  - **ACTION:MONITOR TASK:**evaluate i/o loads on disk where control files are
- Wait Event: Data file related wait events
  -
DB File parallel write	DBWR write
DB File scattered read	full segment scan: **PARAMETER CONSTRAINT:** DB_FILE_MULTIBLOCK_READ_COUNT
DB File sequential read	REGULAR READ
DB File single write	DBWR write
  - examine data file keys and i/o efficiency
- Wait Event: Enqueue
  - wait on a specific enqueue
- Wait Event: Free Buffer Waits
  - Not enough buffer in buffer cache
  - DBWR not keeping up
  - hit FREE BUFFERS INSPECTED limit
  - refrain from chaging dsta file/tablespace status
  - increase DBWR throughput
  - **SQL IMPACT:IMPACTS API DESIGN:**tune application SQL:
    * reduce index scans and full table scans
    * DISTINCT
    * GROUP BY
    * ORDER BY
    * use materialized views
  - **PARAMETER CONSTRAINT:**increase DB_BLOCK_BUFFERS
- Wait Event: Index Block Split
  - strains response time
  - rebuild indexes
  - increase PCTFREE
- Wait Event: LatchFree
  - latches not available
- Wait Event: Log buffer space

  - **PARAMETER CONSTRAINT:**increase LOG_BUFFER
  - **PARAMETER CONSTRAINT:**increase LGWR throughput: put redo logs on fast disk/controller

- Wait Event: Log File related wait events
  - log file parallel write, sequential read, single write
  - fast disk/controller
  - streamline LGWR

- Wait Event: Log File Space Switch
  - Wait for log buffer to be flushed to disk
  - can take 10 seconds
  - **SQL IMPACT:IMPACTS API DESIGN:**understand DML usage of commits
  - understand redo size: **MONITOR TASK:**v$sysstat where name ='redo size' then name='user commits'; divide out to get average size.
  - increase size, number of redo logs
  - DBWR and ARCH throughput increase

- Wait Event: Log File Switch (checkpoint incomplete)
  - same solution as above

- Wait Event: Log File Sync
  - time to flush log buffer to online redo logs when a COMMIT is issued
  - **RANDOM NEURON FIRED:NEEDED FOR REPLICATION:**can we forget redo in replication?
  - **NEEDED FOR REPLICATION:**different server config file in linux ("oracle lite")
  - **ACTION:TUNE:**tune LGWR
  - **MONITOR TASK:ACTION:SQL IMPACT:IMPACTS API DESIGN:**evaluate usage of commits

- Wait Event: Parallel Query Create Server
  - set up parallel query server processs
  - **COST:**can be expensive
  - **PARAMETER CONSTRAINT:**can prestart parallel query servers: PARALLEL_MIN_SERVERS

- Wait Event: PMON timer
  - idleness of PMON: ignore

- Wait Event: RBBMS PC messages and RDBMS PC reply
  - interprocess communication
  - ignore
- Wait Event: SMON timer
  - idleness of SMON
  - ignimore
- Wait Event: Undo Segment Extension
  - rollback segments subected to cycles of extend/shrink/extend ...
  - stops access to the database for 1/100 s
  - **MONITOR TASK:PARAMETER CONSTRAINT:**size rollbacks properly
  - **SQL IMPACT:**OPTIMAL has to be set so that there are not too many shrinks
- Wait Event: Undo Segment Recovery
  - aborted transaction being recovered
  - takes 3 s
  - make more rollback segments
- Wait Event: Undo Segment TX Slot
  - unavailable transaction slots
  - takes 1 s
  - multiple transactions on one rollback
  - increase number of rollback segments
- Wait Event: Write Complete Waits
  - also interprocess pings if using OPS
  - 1 s
  - increase DBWR throughput

5.1.10 Periodically Watch for and resolve locking conflict

- conflict rare but severe
- **MONITOR TASK:**DBA_WAITERS view can be used

How to look

- **DEVELOPMENT TASK:**Third party software for locking conflict checks
- **MONITOR TASK:SQL IMPACT:**v$lock, v$session: need to "calibrate" baseline with normal running. Need to simulate
- catblock.sql defines views (run as SYS)
- utlblock.sql
  - run after catblock.sql
  - creates and drops tables in SYS tablespace so it can fragment this space
  - run in non-sys tablespace
- DBA_LOCKS
- DBA_WAITERS
  - currently waiting sessions
  - session having the resource wanted
  - takes 2 minutes to run
  - can eat all the temporary tablespace
  - pages 563-564
- DBA_BLOCKERS: who is holding things up

Debugging Locking Conditions

- Chapter 13 6.1
- use oradbx, oradebug, truss
- ALTER SYSTEM KILL SESSION on batch jobs; if this fails, us OS kill
- **SQL IMPACT:IMPACTS API DESIGN:**omit commit/rollback: common lock
- **SQL IMPACT:**Do not use LOCK
- **SQL IMPACT:**Do not use CREATE INDEX ANALYZE TABLE VALIDATE STRUCTURE
- **SQL IMPACT:IMPACTS API DESIGN:**Applications should change tables in the same order
- Create indexes on foreign keys

5.1.11 Monitor Shared Pool effectiveness and Tune

MONITOR TASK: This is our highest priority on monitoring

- must bounce database to change
- estimate usage patterns of applications: Oracle Administrator's Guide
- Designer: needs 18Mb

Shared Pool and its components

- part of SGA
- server code
- metadata
- more important that datablock buffers, redo buffers, etc.
- four primary portions:
  - fixed buffers
  - shared pools
  - database block buffer cache
  - redo log buffers
- **FORCDF:**SGA used more in MTS
- time frame 20 processes: instances, process session roll if call PGA: local variables, cursor state
- shared application code goes in shared pool. lib cache, meta data is in ddl cache: "raw" cache
- shared pool management:
  - KGH heap manager: allocates memory
  - Library CACHE: KGL uses KGH
    * doubly linked list
    * size: 251, 509, 1021 etc
    * hash growth requires 2s and rehashing
  - Objects in buckets
    * names, procedure
    * status flag
    * dependency
    * child table
    * authorization table
    * KGL data blocks

- Concurrency control
    - Kinds of locks

NULL	allow lock to be made exclusive
SHARE	read
EXCLUSIVE	**SQL IMPACT:**CREATE, ALTER, DROP

    - PIN: is share or exclusive
    - latches: on code
    - **MONITOR TASK:**library cache latch needs monitoring
- Data Dictionary Cache (row cache)
    - allows simultaneous access to table structure
    - **FORCDF:**CALIBRUNLISTS
    - static admin tables should be there **ACTION:**how to preload

Evaluating and tuning Shared Pool Efficiency

TUNE:ACTION:MONITOR TASK:

- check cache hit ratios
    - **SCALE Reference:**> 98% ok: pg 572
    - Reloads: number of times aged out
    - Invalidations: recreated or recomputed
    - check for developers in production
    - Find out all large code routines and use DBMS_SHARED_POOL_KEEP on > 8KB code
    - **IMPACTS API DESIGN:SQL IMPACT:**large PL/SQL: convert to stored procedures
    - **MONITOR TASK:SCALE Reference:**cache too big if free memory in v$sgastat > 20%
- check parse frequency
    - **MONITOR TASK:**need to set a baseline
    - statement execution $< 2 \times$ number of parses is excessive
    - **SQL IMPACT:**lack of sharing between SQL statements increases load
    - GETHITRATIO < 85% is bad: pg. 574
    - To corrrect excessive parses
        - **IMPACTS API DESIGN:**Audit application code
            - **SQL IMPACT:IMPACTS API DESIGN:**check for upper/lowercase, hard carriage return, etc
            - procedures: **FORCDF:**HWDB and JDBC

        * **SQL IMPACT:IMPACTS API DESIGN:**bind statements: SELECT NAME, AGE FROM NAMES WHER ID=32: should use param for ID or declare epxlicit cursor
        * Enable **PARAMETER CONSTRAINT:**
            · CURSOR_SPACE_FOR_TIME
            · SESSION_CACHED_CURSORS
        * Keep lots of cursors in cache:Need bigger cache

- check for libary cache latch contention
    - **MONITOR TASK:**check in v$latch
    - **SCALE Reference:**if misses > 5%: contention
    - reduce fragmentation
    - reduce parsed calls
    - **FORCDF:**don't use synonyms: pg 515
- check for fragmentation
    - ORA 4031
    - **MONITOR TASK:**can look for this error
    - when contiguous memory is not enough
    - extreme fragmentation
    - need to flush the shared pool
    - pg 576: ALTER SESSION to look for code module doing it
    - Pinning objects in the shared pool combats fragmentation
    - load/reaload is the probable cause
    - check modules over 8K
    - **MONITOR TASK:**look at x$ksmlru
    - **RANDOM NEURON FIRED:IMPACTS API DESIGN:**Can we put modules in via api to C++ and monitor

Evaluating and tuning Shared Pool Efficiency

- **PARAMETER CONSTRAINT:**SHARED_POOL_RESERVED_SIZE : size of reserved area
- **PARAMETER CONSTRAINT:SCALE Reference:**SHARED_POOL_RESERVED_MIN_ALLOC : size of modules to load here: 5-8 Kb
- reserve space for large modules
- **MONITOR TASK:**v$shared_pool_reserved: **SCALE Reference:**5-10% in normal enviroment, 30% is a lot

5.1.12 Other areas to address

- Deadlocks
- **SQL IMPACT:**inefficient SQL
- use of resources by application
- **ACTION:MONITOR TASK:FORCDF:**test each application and monitor: maintain? validate?
- **MONITOR TASK:**monitor for crises like MAXEXTENTS, inadequate free space

5.2 Chapter 12: Space and Growth Management

- ORA 1653: growth errors
- require
  - Growth Analysis
  - Capacity planning
  - Space Configuration
  - Monitoring

5.2.1 Understand the business drivers influencing segment growth

FORCDF: At CDF we commission the database with the detector. This allows experimentalists to acclimatize themselves to the usage of the database. At the same time it allows us to understand the tools needed by the experimental groups dedicated to commissioning and operating the various detector components. It allows us to study usage patterns and growth rates and permits tuning of the database.

5.2.2 Understanding the units of space consumption

Logical storage Units

- Tablespaces
  - system
  - temporary
  - rbs
  - tools
  - application data
  - application indexes
- system holds
  - data dictionary

- system rollback segment
- also, if rollback IMMEDIATE is used: consider mod to sql.bsq to avoid storing procedures and triggers in the system space – but must consult oracle support for this. It is hard to migrate these changes

- segments:entities that reside in tablespaces
- extents
  - initial storage unit
  - unit for growth: **IMPACTS TABLE DESIGN:**INITIAL, NEXT
  - MAXEXTENTS: ora-1628 to 1632
  - MAXEXTENTS to infinity forces DBA to watch growth patterns
- data blocks
  - Data segment
  - various rollback types

Physical storage Units

- files
  - file number is part of row id
  - a segment can span files
  - an extent must be contained within a file
  - every data file has a header with
    * kernel information,
    * checkpoint data,
    * scn and
    * backup info
- blocks
  - building blocks of each file
  - DB_BLOCK_SIZE
  - rowid also contains block id
  - each block has a data block address: dba
  - 84-107 bytes overhead
  - data and free space:**PARAMETER CONSTRAINT:**PCTFREE
  - part of freespace used for DML
  - **PARAMETER CONSTRAINT:**max transactions: INITRANS,MAXTRANS
  - Transaction entry: 23 bytes

5.2.3 Understand the various segments that comprise your database

- tables
- indexes
- rollback segments
- temporary (sort) segments
- bootstrap (cache) segments

Tables

IMPACTS TABLE DESIGN:

- **IMPACTS TABLE DESIGN:**classify by size and growth
- **IMPACTS TABLE DESIGN:IMPACTS API DESIGN:ACTION:FORCDF:**use standard storage parameters
- **IMPACTS TABLE DESIGN:**keep INITIAL, NEXT consistent:

$$\begin{aligned} \text{INITIAL} = \text{NEXT} &= n\times \text{ DB\_BLOCK\_SIZE} \\ &= m\times \text{ OS block size} \\ &= k\times \text{ strip size} \\ \text{Filesize} &= l\times \text{ DB\_BLOCK\_SIZE} \end{aligned}$$

- Allow growth (month, year)
- **PARAMETER CONSTRAINT:**for export, watch out for COMPRESS=Y
- Partition tables that will grow to very large size
- monitor usage
- **SCALE Reference:**100 extents is large: impacts SYS space needed
- **MONITOR TASK:**monitor harmful fragmentation

Indexes (B*Tree)

IMPACTS TABLE DESIGN:

- Restrict to columns not frequently updated – or partition the index
- Restrict number of tables with large insert/update
- **FORCDF:**Probably a rolloff requires an index rebuild
- **FORCDF:DEVELOPMENT TASK:ACTION:**Simulation:study index usage
- Only certain columns help performance initially:
  - primary

  - foreign
  - partitioning lags

- **IMPACTS TABLE DESIGN:IMPACTS API DESIGN:** check that indexes are used correctly

- allocate space in memory to create indexes

  - ensure adequate temporary tablespace
  - **PARAMETER CONSTRAINT:** increase SORT_AREA_SIZE (use ALTER SESSION)
  - **PARAMETER CONSTRAINT:SCALE Reference:** set CREATE_BITMAP_AREA_SIZE:
    (column cardinality $\times$ (DB_BLOCK_SIZE $\times$ .5)) $\times$ 1.15

- How a bitmapped tree is made: **INT SAME SIZE AS PRD:** puts requirements on memory

  1. Scan table by one or more processes (parallel builds)
  2. rowid's and values into bitmap engine:
     PARAMETER CONSTRAINT: CREATE_BITMAP_AREA_SIZE give memory to the engine (needs temp tablespace otherwise)
  3. bitmaps may be split or merged
  4. sort and save in tree

- **SQL IMPACT:** DISTINCT is fast (no reference to data)

- **SQL IMPACT:** GROUP BY can be fast

- **FORCDF:** need to check for hwdb and triggerdb

- Range scans can be a problem: need merge **PARAMETER CONSTRAINT:** BITMAP_MERGE_AREA_SIZE

- **MONITOR TASK:** Monitor actual cardinality: need size of largest bitmapped index

- if query takes 60% of table, can get BYTTES from USER segments

- if OPS, hot blocks, consider reversed indexes

- **FORCDF:**: CID will lopside the tree as in stock numbers, page 243

- high DML activity on a B*Tree causes nonoptimal structure:

  - indexes populate large number of sparsely populated blocks
  - tall tree
  - flat due to low cardinality

Consider an index rebuild

MONITOR TASK:DEVELOPMENT TASK:

- **MONITOR TASK:**use ANALYZE_INDEX .. VALIDATE STRUCTURE
- **FORCDF:**establish methods, consider impact on performance
- **FORCDF:**maintain reports
- use SYS.INDEX_STATS (page 600)
- if delete leaf big compared to all leafs, rebuild
- **SCALE Reference:**20% is a rogue: if delete leaf > 20% of total, rebuild
- **SCALE Reference:**if height $\geq$ 4 rebuild
- dump structure to see number of keys spread across leaves

Dumping An index tree structure

MONITOR TASK:DEVELOPMENT TASK:

- ALTER SESSION SET EVENTS IMMEDIATE TACE NAME TREE DUMP LEVEL <NUMBER>
- <NUMBER> = OBJECT_ID FROM DBA_OBJECTS: page 601
- File goes to USER_DUMP_DEST
- **MONITOR TASK:FORCDF:**set up Auto parse and study - auto monitor as well?
- **PARAMETER CONSTRAINT:**may need to increase MAX_DUMP_FILE_SIZE
- Can analyze depth and balance
- **FORCDF:**CID: Sorts of numbers cause lopsidedness
- **INT SAME SIZE AS PRD:FORCDF:**need tests and simulation

The rebuild option

- ALTER INDEX ... REBUILD but does not fix problems
- ANALYZE TABLE ... VALIDATE STRUCTURE CASCADE detects index corruption

5.2.4 Rollback Segments

- provide undo
- provide read consistency
- need space, or get ora 1628

considerations of space management

- physically like any other segment ie. tables
- **PARAMETER CONSTRAINT:**INITIAL, NEXT depend on transaction volume: same for each segment
- OLTP and Rollback: can have larger segments around but offline:
  - for rebuild
  - good to have just in case
  - cannot mix with smaller segments: because of the round-robin approach to using the segments, this makes no sense since a job will fail if it gets the wrong one
- Segments grow due to information put into them:
  - the information in them becomes stale **HUH?**
  - duration of rollback usage: see ch 11
  - to reclaim space, use **PARAMETER CONSTRAINT:**OPTIMAL: set to "ideal" size. Should shrink automatically.
  - rollback segments that are too small lead to read consistency problems
  - when they are too large, they waste space
  - **FORCDF:**error on the side of too large if it is just space: then monitor usage and readjust. Our space demands upfront will be smaller so we can afford this to be a bit large and then resize with experience.
  - **MONITOR TASK:**ALTER ROLLBACK SEGMENT SHRINK does the trick
- Have dedicated tablespace: RBS
  - may have separate sized in separate spaces for performance reasons **FORCDF:**at start, may wish to set up several sizes so that we are ready for anything
  - one space for all at least allows apace to be available
  - **PARAMETER CONSTRAINT:**INITIAL = $n \times$DB_BLOCK_SIZE
  - Applications cannot choose rollback segments
  - **PARAMETER CONSTRAINT:**Ensure RBS tablespace has 25 extents
    * **DEVELOPMENT TASK:MONITOR TASK:**need to evaluate current transactions in undo
    * prevent undo header waits
    * **SQL IMPACT:**not only INSERT, UPDATE, DELETE
    * can get peaks in demand
  - **FORCDF:HUH?**Is a suite of database performance tools not commericially available or something we want to budget for. Search Web? As a DOE-wide thing? There are scripts in chapter 19.
- Consider two sets for doing coalescing

5.2.5 Temporary (sort) Segments

- **PARAMETER CONSTRAINT:**Add segments when SORT_AREA_SIZE is inadequate
- **ORACLE OS Parallel:**"ORACLE virtual memory"
- maintain dedicated temporary tablespaces
- Explicitly specify tablespaces for a user:**FORCDF:**The web browser is a good example of this: CREATE TABLEPSACE ... TEMPORARY
- For a large sort, need two tablespaces of equal size (may wish to rebuild, so switch users to the other one). Call them temp1 and temp2:

  1. ensure that there is space
  2. script to alter all users to temp2
  3. check users are altered
  4. mark temp1 offline
  5. note physical attributes of temp1
  6. drop temp 1
  7. recreate temp1 using information stored in 2
  8. leave ready until temp2 needs to be switched in

  Could use RBS, but this may fragment it

- Can do this for RBS as well
- **NEEDED FOR REPLICATION:HUH?**Is this needed for remote sites?
- **FORCDF:**keep standard EXTENT sizes for temporary segments
  - **PARAMETER CONSTRAINT:**INITIAL,NEXT multiples of DB_BLOCK_SIZE
  - **PARAMETER CONSTRAINT:**PCTINCREASE=0
  - **PARAMETER CONSTRAINT:**set INIT = NEXT = $n\times$SORT_AREA_SIZE + DB_BLOCK_SIZE where $n \geq 1$.
  - **PARAMETER CONSTRAINT:**set INIT = NEXT = $m\times$DB_MULTIBLOCK_READ_COUNT $m \geq 1$.
  - **PARAMETER CONSTRAINT:**
- **FORCDF:**may wish to stripe/maintain on dedicated disks
- **SQL IMPACT:IMPACTS API DESIGN:**recursive calls: SQL generated by Oracle to ensure user SQL is OK: checks that columns actually exist, etc.
- **MONITOR TASK:SQL IMPACT:IMPACTS API DESIGN:**excessive recursive calls degrade performance

- **FORCDF:**in mirrored environment, may want temp on unmirrored one just looses a transaction but no real data
- **FORCDF:**high speed non-striped, plain disk with a separate controller
- **MONITOR TASK:**monitor use of (scripts in chapter 19)
  - recursive calls
  - sorting

5.2.6 Bootstrap (cache) Segments

SYSTEM: Ignore them

5.2.7 Be familiar with errors and effects that result from inadequate space management

ORA-1628 to 1632	max number of extents
ORA-1650 to 1655	Unable to extend
ORA-1555	Snapshot too old

- Fragmentation is a root cause (ORA 1650-55) for:
  - row level internal fragmentation
  - block level internal fragmentation
  - segment level internal fragmentation
  - tablespace level external fragmentation

Row Level Fragmentation

- row chaining (during insert)
- row migration (during update)
- row in different data blocks: multiple i/o read
- row too large to fit in ONE block
  - **PARAMETER CONSTRAINT:IMPACTS TABLE DESIGN:**rowsize is limited by DB_BLOCK_SIZE
  - **FORCDF:HUH?**need DB_BLOCK_SIZE in designer
  - **FORCDF:**DB_BLOCK_SIZE must be same in all instances
  - **FORCDF:**must take care with LOB
- a migrated row still gets referred to by its old rowid, so this cuts performance
- A row has
  - row header info $\geq$ 3 bytes

- LOB $<$ 4 kb stored inline **FORCDF:IMPACTS API DESIGN:**implies codegen needs to know DB_BLOCK_SIZE and a need for rebuild to read the database!!!!!
- Colomn head 1 byte if data $\leq$ 250 bytes
- Colomn head 3 bytes if data $>$ 250 bytes
- CREATE/ALTER TABLE: disable storage in row forces out of line storage **FORCDF:**check trigger, hwdb

- **IMPACTS TABLE DESIGN:FORCDF:**Use VARCHAR2 not CLOB of data $<$ 4K
- **IMPACTS TABLE DESIGN:FORCDF:**Use NVARCHAR2 not NCLOB of data $<$ 4K (nonchar data)
- **IMPACTS TABLE DESIGN:IMPACTS API DESIGN:**NULL columns at end of row don't have a header: put NOT NULLS prior to NULL
- A row also takes 2 bytes in block header
- **PARAMETER CONSTRAINT:**Row migration controlled with PCTFREE, PCTUSED: govern space constraints in a block
- **PARAMETER CONSTRAINT:**Freelists keep track of free blocks
- **IMPACTS TABLE DESIGN:IMPACTS API DESIGN:PARAMETER CONSTRAINT:**Blocks are filled up to PCTFREE: this leaves room for UPDATES
- **MONITOR TASK:**Need average and maximum row sizes
- **FORCDF:HUH?**Consider CDF databases and updates:
  - HWDB uses updates.
  - RUNDB uses updates.
  - **HUH?**DFC?
  - **HUH?**Trigdb
- **MONITOR TASK:**Monitor but during low database use
- **MONITOR TASK:FORCDF:**need to specify a monitoring schedule
- **PARAMETER CONSTRAINT:**PCTUSED keeps too much free space from being wasted. Allows block to have a row inserted "between" other rows
- **MONITOR TASK:**monitor table fetch continued row in v$sysstat
- **ACTION:**Solve fragmentation problem by a table rebuild OR
  1. Prepare a list of tables needing repair
  2. Create table to capture row id's: pg 621 and use TOOLS tablespace not SYSTEM tablespace

3. Analyze each table: ANALYZE TABLE <table> LIST CHAINED ROWS
4. copy rows
5. delete from original
6. reinsert into original
7. delete temporary table

Block Level Fragmentation

- High influx and removal of rows
- **PARAMETER CONSTRAINT:**Control with PCTUSED
- packing blocks with rows reduces i/o needs
- need familiarity with DML
- **PARAMETER CONSTRAINT:SCALE Reference:**PCTUSED=40 by default
- **MONITOR TASK:**To identify block fragmentation
    1. list popular tables for medium and large segments
    2. compute average row size (AVG())
    3. use 20% error if not able to use SQL to do this
    4. divide by DB_BLOCK_SIZE to get number of blocks
    5. compute blocks held by subtracting 1 from header
    6. compare 4 with 5
- delayed logging block cleanout causes fragmentation:
    - mark for delete but not written
    - rows are added
    - this leaves holes

  later on they are populated but not immediately. Can hurt if doing large inserts and near the edge of the extent. **FORCDF:IMPACTS API DESIGN:SQL IMPACT:**Do a full table scan after a large delete: this ensures a flush to disk was done.

Segment Level Fragmentation

- when segment spans multiple extents
- not necessarily a performance problem
- allows spread over multiple controllers

- single extent does not guarantee contiguous data: no chance for contiguous data (by design) on RAID
- segment extents should not be frequent: recursive calls get made.
- **MONITOR TASK:**check v$sysstate for high recursive calls **FORCDF:**QIEPED a good test
- single extent may expedite full table scans **PARAMETER CONSTRAINT:**DB_FILE_MULTIBLOCK_READ_COUNT indexed lockups do not matter **HUH?**
- **SCALE Reference:**hunderds of extents cause segment extent map to span multiple db blocks.
- multiple extents allow disks to grow with time, without up front costs.

Tablespace Level Fragmentation

Two definitions:

- bubble: chunk of free space
- honeycomb: series of chunks adjacent to each other
- fragmentation results in a waste of space
- not necessarily performance problem
- **SQL IMPACT:**CREATE, DROP, TRUNCATE may suffer to find bubbles
- SMON coalesces bubbles (but don't use it)
- first, a lesson: what Oracle does to insert a row. if successful at any step, it stops.
  - Freelist examined
  - look for free space to extend
  - look in bubbles
  - use rounded up to mod 5
  - look for bigger bubble
  - if no bubble look for honeycomb to coalesce
  - try auto extend
  - abort
- Cause for tablespace fragmentation: **PARAMETER CONSTRAINT:IMPACTS TABLE DESIGN:**storage parameters not multiples of one another
- **PARAMETER CONSTRAINT:IMPACTS TABLE DESIGN:**steps needed to prevent

  - segregate segments based on size and growth rates
  - ensure reuse of free bubbles
  - all segments in a tablespace should have the same storage parameters
  - ... or at least keep as multiples
  - specify default sotrage parameters
  - specify PCT_INTCREASE = 0 or 100 but nothing in between
  - periodically coalesce tablespaces
  - do not use SMON
  - use parallel query option (PQO) judiciously for creating indexes and tables. each parallel server gets an extent, these are combined and this leads to odd sizes.
  - avoid higher parallelism than number of data files**CHECK BOOK:**

- partitioning allows rebuilds
- to identify tablespace fragmentation and reorganize specific files
  - identify bubbles (script pg 635)
  - identify segments for rebuild
  - use a low usage time
  - ... or use a backup table
- to rebuild
  - Save data: export or CREATE TABLE AS SELECT
  - Drop segments
  - Coalesce tablespace: ALTER TABLESPACE COALESCE
  - recreate segments
  - rerun query to check defragmentation worked

5.2.8 Enable AUTOEXTEND for all application tablespaces and SYSTEM

- in CREATE/ALTER TABLESPACE
- **PARAMETER CONSTRAINT:**up to MAXSIZE
- up to using disk
- not for temporary or rollback segments

5.2.9 Use Standard filesizes

- 32 bit max is 2GB
- **PARAMETER CONSTRAINT:**filesize = $n \times$DB_BLOCK_SIZE
- size of control file and archived redo logs controlled by oracle
- **PARAMETER CONSTRAINT:**multiple of OS_STRIPE_SIZE
- **PARAMETER CONSTRAINT:**may need to adjust: DB_FILES
- requires a bounce

5.2.10 Accommodate data file header block in standard file sizes

- A 1001 MB not 1000MB because the block must be accommodated

5.2.11 Use standardized storage clauses for each segment type

IMPACTS API DESIGN:IMPACTS TABLE DESIGN:

- **FORCDF:**Implementation in code gen must be carefully done
- see chapter 6 for some standard
- **PARAMETER CONSTRAINT:SCALE Reference:**storage
  - INITIAL, NEXT (with PCTINCREASE), PCTICREASE "influence size"
  - PCTFREE: space to leave for inserts
  - PCTUSED: when block goes into freelist
  - FREELISTS: set so no latch contention, depends on concurrent inserts: if 10 concurrent, set to 5.
  - INITRANS: 23 byte slots/block for entries allocated at creation time. set at least to freelists, < 10
  - MAXTRANS
  - MINEXTENTS
  - MAXEXTENTS unlimited
  - OPTIMAL for rollback
- Algorithms used by oracle to allocate extents
  - NEXT and PRECENT_INCREASE/DB_BLOCKSIZE rounded up to next multiple of blocksize
  - < 5 block before extents: use up blocks!

  Hence, some extra blocks can be tacked on, even if multiples were carefull chosen: parameters only "influence" the size of extents

5.2.12 Set INITIAL and NEXT keeping DB_BLOCK_SIZE in mind

- reduces free space fragmentation
- **PARAMETER CONSTRAINT:**aids in multiblock reads: DB_FILE_MULTIBLOCK_READ_COUNT
- multiblock reads cannot span extents

HUH?but why bother if oracle is tacking on bubbles uncontrollably

5.2.13 Set PCTINCREASE optimally to reduce fragmentation

- next extent done by roudning up to nearest DB_BLOCK_SIZE multiple
- PCTINCREASE: previous size is used to increase so one gets fractions of DB_BLOCK_SIZE
- beware the neighboring bubbles in free blocks
- PCTINCREASE can create fragmentation problems
- used if growth rate is not known
- always set to 0, but if stuck set to 100 to preserve multiple of DB_BLOCK_SIZE

5.2.14 Understand FREELISTS and set them optimally to reduce latch contention

- freelist tracks space in each block
- as block populated to below PCTFREE, it is removed from freelist
- as deletes occur, blocks put back into freelist according to PCTUSED
- Structure of freelist block:
  - used status flag
  - dba at beginning of list
  - dba at end of list
- Freelist types:
  - master:
    * tracks free space within a segment
    * processes inserting into one segment will contend
    * this is a table and rollback contention
    * “eventually” updated to synchronize with others
  - process:
    * additional freelists

    - different processes use different blocks
    - used to seew what is available
  - transaction:
    - > 16 transaction freelists/seg
    - gets marked when process freelist is modified
    - commit done only if processs freelist is updated

- if an extent is called for, a freelist merge is done: hence extents reduce performance
- the blocks in the master are added to the processs freelists
- if the master is empty, it looks at other process freelists

5.2.15 Periodically coalesce your application tablespaces

- combine honeycombs into bubbles
- to coalesce
  - ALTER TABLE ... COALESCE
  - PCTINCREASE not 0 fires SMON
  - PCTINCREASE=1 creates uneven segments
- don't coalesce Rollback and Temporary: too dynamic
- **FORCDF:INT SAME SIZE AS PRD:**measure size of effects with simulation
- **IMPACTS TABLE DESIGN:**
  FORCDF:
  PARAMETER CONSTRAINT:
  best to keep in control with INITIAL, NEXT, OPTIMAL

5.2.16 Understand the effect of an erroneous high water mark

- HWM: what was the largest size of a table between two DDL statements
- at CREATE TABLE HWM =0
- at CREATE TABLE AS SELECT HWM = number of blocks filled
- only blocks with data in HWM
- computed in units of 5 blocks
- **SQL IMPACT:**used in full segment scan to choose how far to read
- **SQL IMPACT:IMPACTS API DESIGN:**a huge delete will not decrease the HWM: result in big, inefficient scans
- must use TRUNCATE: ALTER TABLE DEALLOCATE UNUSED is not sufficient

- Action on TRUNCATE:
  - exclusive lock or request TX lock
  - flush block from buffer cache to disk
  - release block beyond MINEXTENTS
  - update freelists
  - reset HWM
  - BUT removes valid rows!
- to save rows:
  - save with export or pCTAS (index scan saves time)
  - TRUNCATE
  - INSERT INTO ... SELECT FROM
- **HUH?**can one take a lock?
- **MONITOR TASK:**only tables with lots of deletes
- **IMPACTS API DESIGN:**need table plan:
- **MONITOR TASK:**check tables with large init/extent
- **MONITOR TASK:SCALE Reference:**total vs used pg 659: of 40% rebuild but consider table size and how often a full table scan is actually done
- **MONITOR TASK:SQL IMPACT:IMPACTS API DESIGN:**use EXPLAIN PLAN in monitoring

5.2.17 Periodically Deallocate Unused Blocks

- must truncate with and DROP STORAGE
- OR - ALTER TABLE DEALLOCATE UNUSED
- only deallocates things above the HWM
- supports a KEEP to allow some extent
- KEEP also allows MINEXTENTS to be overridden and then it is reset in the data dictionary
- Also: LOBs, indexes and overflow segments
- **DATABASE RECOVERY HINT:**can also use to recover disk space in an emergency
- **HUH?RANDOM NEURON FIRED:DEVELOPMENT TASK:**compare a 100G database with 100 1G BLOBS the same as a 100G with 10^6 rows and 50 tables: scan our parameter space – are we at a stable performance maximum?
- **MONITOR TASK:**DBMS_SPACE.UNUSED_SPACE page 663, also useful for HWM

5.2.18 Avoid dropping data files explicitly

- no offline drop: reason is hardware: must rebuild tablespace
- no place to store file status in SYS.FET$ so cannot just delete file
- if in ARCHIVELOG mode, use offline option of ALTER DATABASE
- **FORCDF:**this is a reason to use ARCHIVELOG mode even in a warehouse
- **FORCDF:NEEDED FOR REPLICATION:**what are the implications in replication

5.2.19 Learn the tablespace shuffle for emergency unanticipated space requirements

ACTION: Prepare this as a canned procedure to be checklisted through with tools annotated

- take USERS offline
- reduce size of TEMPORARY
- need to backup a dormant tablespace
- restore dormant when done
- may move datafiles with operating system backup
- **MONITOR TASK:**need to keep an up to date list of files and tables in them
- **DEVELOPMENT TASK:**why not use offline and delete as an archive method?
- make sure dormant space is enough **RANDOM NEURON FIRED:**classify extra space according to performance and safety
- **MONITOR TASK:**make sure temp space ok for applications
- **MONITOR TASK:**ensure no users attached
- check referential integrity **FORCDF:**there is a rolloff parameter in the calibrations
- check for indexes
- no online rollback segment can be handled this way
- advise users
- back up control file before and after doing this: **DEVELOPMENT TASK:**see transportable tablespaces (Section 8.2.1)

5.2.20 In DSS don't blindly accept unrestricted segment growth

- **FORCDF:**Purge point: rolloff in calib db
- **FORCDF:**they compare to full table scans of 50 MR
- **FORCDF:**do a "trend analysis" of database activity: for DSS, this is offline
- **DEVELOPMENT TASK:**consider the rolling window option

RANDOM NEURON FIRED:NEEDED FOR REPLICATION: What about BLOB being fetched using oracle client? What does this do to performance?

Chapter 6

Troubleshooting

6.1 Chapter 13: Interpreting the Alert log and trace files

Factors in running without disruption and at high performance:

- administrators that are capable and talented
- detailed requirements analysis and capacity planning
- careful selection of hardware and software architecture
- optimal installation, configuration, tuning
- proactive monitoring: see page 672

RANDOM NEURON FIRED:DEVELOPMENT TASK: Must parse the error log: can we do this and report to online monitoring system

6.1.1 The alert log

Items noted in log:

- redo log switch
- check point
- critical error:internal or ORA 600
- startup/shutdown
- tablespace: drop/add, on/offline, readonly/read-write
- add segment
- status change
- recovery

Goes to BACKGROUND_DUMP_DEST

continually tail the alert log

- **MONITOR TASK:FORCDF:**in control room **HUH?**does oem do this or provide?
- look at first thing (shift)
- "control room" on 8 when no shift?
- shifts for DBA's at start: just like any detector
- a 24h operation requires monitoring and development to automation just as any other detector element

Be familiar with events/messages in the alert log

- **ACTION:FORCDF:**document response/action to
  - Routine info
  - non-routine info
  - non-critical warning, errror
  - critical errors
- Use oerr ora 1578 to get printout of error: **FORCDF:**to the error log screen
- **HUH?MONITOR TASK:FORCDF:**license: high water mark is in database shutdown
- Serious Problems (check oerr on the numbers, page 683-697)
  - segment unable to extend due to inadequate space
  - license **MONITOR TASK:**v$license
  - **TREAT AS HV TRIP:**Space management timeout
  - cannot alloate new log checkpoint complete
  - archive stuck
  - **TREAT AS HV TRIP:**database corruption
  - application deadlock detected
  - **MONITOR TASK:**memory leak ORA 0600

    $$SPACE\_LEAK$$

  - background processes being terminated (what is rate?)
  - internal errors
    - **ACTION:MONITOR TASK:**put together automation of checklist on pg 697
    - **ACTION:MONITOR TASK:**monitor automatically in the control room

- Edit repetitive messages to be listed once
  * **ACTION:**Use filtering, perhaps something like MCS
  * **ACTION:**Space for backup logs to allow trend analysis
- archive and trim the alert logs every 2 weeks to 3 months
  * part of the CDF operations
  * what to do when not running: again 8th floor control room?
  * tool? to look at
  * trim but don't DELETE the file: oracle keeps file pointer and all you can do is helplessly watch the disk fill until you can bounce the database!
- trace files: accompany log file: abnormal events
  * has memory dump , call stack, the DBA affected
  * **MONITOR TASK:**need tool: look for a data file segment
  * should we keep a log of segments in case the DBA is too sick?
  * use automated error report to send severity based on what is actually affected
  * need to simulate the problem: perhaps any machine OK
  * can crash instance

6.1.2 know how to identify trace files that belong to a specific event/session

MONITOR TASK:ACTION: Automate: Use ps -ef to get the UNIX ID and:

- for background process: <BKG PROC NAME>_<SID>_<UNIX ID>
- for foreground process: ORA_<SID>_<UNIX ID>.trc

6.1.3 Know what causes trace files to be generated

- database/session crash
- database/session hanging
- NFS mounted volumes are unsupported and cause hangups pg 708
- **ACTION:**make Database hang checksheet pg 708
- **HUH?**How to dump state if database is hung?
- with oracle support: ALTER SESSION SET EVENTS pg 708: can cause crash!
- **MONITOR TASK:**use par, truss, glance etc (pg 712)
- **MONITOR TASK:**use ps -ef
- **MONITOR TASK:**oradbx, oracledebug: goes to USER_DUMP_DEST
- **FORCDF:**ACE training? What level of debug do they do?
- **FORCDF:MONITOR TASK:**check archive log directory

6.1.4 To diagnose a Hang

ACTION: This requires tests and drills

- **ACTION:**what is capacity of components: test/simulate
- resources gobbled up
- reproduceability
- duration
- i/o, cpu, utilization
- **SQL IMPACT:**options used: can we dump SQL
- Data Dictionary corrupt
- is there non-default locking
- check waits: v$session_wait etc pg 714
- check oracle priorities matched with original **MONITOR TASK:**need to keep a set of original values

6.1.5 Tracing

ACTION: Need to drill/think:Can be used for debugging but enable with forethought **RANDOM NEURON FIRED:**We can develope our expertise and experience with drills

- SQLNET/NET8
  - tiers to consider: client, application, databases
  - client to server debug
  - levels: 0, 4, 10, 16 = off, user, admin support
  - degrades performance
  - same components need a bounce afterwards
  - again, the log files cannot be removed since the file pointer is kept
  - page 716 for how to do this
- Server or client packet tracing
  - **FORCDF:ACTION:**Write instructions
  - needs oracle support ("underscore" params)
- Oracle names to trace name server: requires bounce of nameserver
- Trace Assistent: **HUH?COST:**do we want this
- Oracle Trace **HUH?**oem can do this: do we have this with kerberos
- PL/SQL: needs trace files, dump buffer. requires an event (oracle support)
- ODBC tracing

6.1.6 Core Dumps

- in CORE_DUMP_DEST
- can check in USER/BACKGROUND_DUMP_DEST for trace files
- dbx, adb, gdb **HUH?**totalview?

6.1.7 oracle debugging tools

- undocumented (perhaps fear of reverse engineering?)
- only with oracle support (ditto)
- Oradbx
  - Asynchronous messages with oracle debugger
  - can take an event based dump
- Oradebug replaces oradbx, can attach, or dump SGA, PGA
- SQL_TRACE: DBMS_SYSTEM wrapped to discourage use
- "must have oracle support"
- **COST:ACTION:DEVELOPMENT TASK:**training for dba interaction with oracle on this

6.1.8 Events and Setting Events using a debugger

- dump
- conditional dump
- chage behaviour of oracle
- in $ORACLE_HOME/rdbms/mesg/oraus.msg
- set with ALTER SESSION EVENT (instance level)
- use DBMS_SYSTEM.EV procedure
- use with debugger
- some events enabled via bounce
- Can, with help of Oracle support
  - dump trace files
  - crash the process
  - invoke the debugger

6.1.9 Make tracefiles readable by developers

- This is an underscore parameter so need help from oracle.
- Trace files normally not visible because of UNIX privilege setting.

6.2 Chapter 14: Identifying and Repairing Database Corruption

ACTION: This chapter contains a number of drills that can generally be tried out and then put into nightly validation on the database. This means they are ready to use and working.

6.2.1 Corruption and the ways it can manifest itself

- hardware: i/o, memory, network
- software:a bug
- tools to battle hardware corruption
  - mirroring
  - RAID
  - parity check and reconstruction
  - redundant hardware
- tools to battle software corruption
  - OS Kernel
  - database
  - middlelayer (www)
  - application
  - client

6.2.2 Classes of corruption

ACTION:DEVELOPMENT TASK:tools to narrow down on these

physical

- media or data block
- memory
- logical, data
- damage to memory/data structure

- cause:
  - hardware: disk
  - software: OS or administrative (pilot error)
- effect: cannot read database blocks

Media corruption

- bug in hardware
- affects disk
- areas affected:
  - Data
  - Redo
  - Archived log
  - control file
- Effect on Data
  - unreadable
  - severity depends: header or body-
    * header can wipe file
    * body is "only" records
  - Restore from Backup
  - blocks affected:
    * table
    * index
    * rollback
    * tempoary
  - corrupt segment can crash database
- Data dictionary and system require backup restore
- **MONITOR TASK:**ORA 1578
- **MONITOR TASK:**Determine segments with script, pg 747

6.2.3 Reparing Corruption to Tables

- retrieve data as best you can
- identify the bad row
- recreate the bad rows

- **IMPACTS TABLE DESIGN:**for each table, deterimine what it means to have lost the data
- **FORCDF:**if reloadable, this should be in validation
- 10231 is a database event: can do with oradebug, must not necessarily bounce the database
- repair via indexes can be done:
  - can use to retrieve rows
  - **ACTION:**can practice this
  - need EXPLAIN PLAN to check only indexes used
  - can put into validation testing
  - can get pruning key into a temporary table and do a join
  - **RANDOM NEURON FIRED:**practice, then into validation

6.2.4 Repairing Corruption using rawid, to Indexes, and Temporary Segments

- RowID hint
  - Range scan
  - for tables without indexes
  - **ACTION:**pages 754-755:try this: classify tables
- repair index: drop them and recreate
- Temporary segment corruption
  - causes transaction failure
  - relocate tablespace according to pg 756

6.2.5 Repairing Corruption to a Rollback Segment

- hard to do
- destroys transaction consistency
- could be held for read consistency or delayed block cleanout
- for offline segment, gives error when brought online: just drop and recreate
- if rollback corrupt: take offline; may drop/recreate
- BUT may have transactions trapped
- possible failures:
  - undo header block

- segment body: undo blocks

- Action
  - point in time recovery *if* you know *when* the corruption occurred
  - recovery using initialization parameters
    * manual resolution
    * need DBA with knowledge
    * need into segments – can use an event
  - to fix:
    * Drop, recreate segments involved
    * **ACTION:**Need scripts ahead of time: grant privileges
    * manage internal consistency
    * examine impact
    * need oracle support to force the database open to determine damage

6.2.6 Media Corruption in Online Redo Logs

- If archived logs have all the current information, then open database with RESETLOGS
- RESETLOGS is not easy to deal with
- mirrored group may be ok: **HUH?**how to tell?
- **FORCDF:**may know from hardware if disk is ill
- corruption can be created by incorrectly applying redo logs when rolling forward
- need to understand cause
- if one of the mirrored set works, ok
- if the group that is corrupt is not current, use ALTER DAMAGE CLEAR LOGFILE
  - Especially important before an ARCHIVE
  - log can freeze if corrupt and it is not full
- if current is corrupt, you have to restore the backups prior to the corruption
- there is an underscore parameter for when the previous fails.
- The DBA has to make the database consistent

6.2.7 Media Corruption in Archived Redo logs

- Database will not crash
- create holes in sequence: will show up in media recovery attempt
- causes:
  - incorrect backup procedure
  - back up while being created: cron job moves before done! Need to check status when moving
  - FTP in ASCII Mode
  - **COST:DEVELOPMENT TASK:FORCDF:**LOG_ARCHIVE_DUPLEX_DEST does multiple redo logs

6.2.8 Media Corruption in Control Files

- **PARAMETER CONSTRAINT:**maintain 3 copies: set with CONTROL_FILES parameter
- one breaks and the database crashes
- keep 3 backups
  - binary backup
  - ascii backup
  - ascii dump
- document all changes to the database so the control file can be reconstructed
- to recover
  - Edit config<SID>.ora to eliminate suspect file
  - If not sure which to do, try one at a time until DB does not crash (**ACTION:**drill)
  - copy valid files over all others and restore
  - if all are corrupt
    - use a script with CREATE CONTROL FILE
    - **ACTION:**MUST have all data files listed and their sizes
    - if no script, need to dig out info: costs a lot of time

6.2.9 Proactively check for and detect media corruption

MONITOR TASK:a general shopping list to detect media corruption; there are issues of performance vs security

- run db_verify

- ALTER SYSTEM CHECK DATAFILES GLOBAL;
- VALIDATE STRUCTURE ... CASCADE
- DB_BLOCK_CHECKSUM
- LOG_BLOCK_CHECKSUM
- monitor v$recover_file, v$datafile v$rollsrt pg 764
- check database alert and trace files
- oracle support: check events if suspect corruption

6.2.10 Understand what you can do to prevent media corruption

- Hardware evaluation and redundancy
  - **FORCDF:**Evaluate
  - mirrors
  - parity
  - hot swappable
- **COST:ORACLE OS Parallel:**redundancy within oracle
  - online redo mirror
  - control file mirror
  - archive redo log
- **INT SAME SIZE AS PRD:**robust test environment (pg 765) that is similar to production prior to deployment
- security
- **ACTION:**valid backup with periodic check for corruption

6.2.11 Techniques to Deal with Memory Corruption

- Memory corruption is due to:
  - hardware
  - oracle bugs
  - application code
  - user action
- In the SGA: cause for alarm
- In the PGA: PMON rolls back. can just reconstruct
- Data loss potential:

  - data block buffer cache: loose data
  - shared pool (dictionary, library): no data loss

- good chance of a crash if in the SGA
- redo log corruption can prevent recovery
- buffer cache corruption usually originates in memory itself since foreground processes do checks
- Detecting Memory Corruption
  - when PMON is cleaning up aborted session
  - when DBWR is writing a dirty block
  - when a server process is reading/chaning a block
- **PARAMETER CONSTRAINT:**if suspect memory corruption, set DB_BLOCK_CHECKSUM but it slows things down.
- **PARAMETER CONSTRAINT:**with oracle support can check log files with _LOG_BLOCK_CHECKSUM but this needs a database bounce and it writes to the redo log
- to prevent:
  - **INT SAME SIZE AS PRD:**can test with a production like enviroment
  - **INT SAME SIZE AS PRD:**test new releases on a similar system to production
  - don't use single task enviroment except for upgrade/migration

6.2.12 Techniques to deal with logical corruption

- manifest within data: violates business rules
- deadly
- orphaned rows
- due to
  - bugs
  - admin error
  - incomplete recovery
  - corrupt segments
  - oracle bugs
- detection
  - familiarity with applications
  - check table sequence load

Preventing/repairing logical corruption

- **INT SAME SIZE AS PRD:**test on an integration database
- security
- **INT SAME SIZE AS PRD:**standby database so you can recreate and test after recovery
- VALIDATE STRUCTURE ... CASCADE (pg 771) takes time!
  - do twice before a rebuild
  - if same error (ora 1498) then internal corruption in segment
  - if different file and block number, media error: disk controller or SCSI chain
- run an EXPLAIN PLAN to see which segments are being used
  - check for corrupt indexes
  - on joins, may be the driving table that is corrupt
  - only need VALIDATE STRUCTURE on the segments that are corrupt
- rebuild index with CREATE INDEX builds from the table but is safer: need to check indexes are ok if you want to use ALTER INDEX

6.2.13 List components susceptible to corruption

Players:

- system admininistration
- network administration
- database administration
- application designer
- system/datatables developer

Classify potential corruption:

- hardware
- network, database, etc
- **ACTION:**document corruption as it occurs as in pg 776
- procedures and practice reduce subjectivity

Reduce/prevent corruption

- design and development
- version control (CVS)
- test
- security
- $<$ 2GB data files

To detect corruption early

- **FORCDF:**External check: hardware check
- database physical corruption
- **FORCDF:MONITOR TASK:**monitor system and database **RANDOM NEURON FIRED:**l3, farm?
- **MONITOR TASK:**parse syslog, alert log, trace log: control room/ace
- leave an audit trail
- application log, validation on production
- analysis with VALIDATE_STRUCTURE
- DB_VERIFY must specify block size (pg 778) if blocks bad, repeat to see if the same block
- DB_BLOCK_CHECKSUM, LOG_BLOCK_CHECKSUM,DB_BLOCK_COMPLETE_CHECKSUMS, etc.
- EVENTS (oracle support): 10015, 10210, 10212
- **FORCDF:**OS tools
  - Sum for a file, validate backups
  - chksum consistent implementation

6.2.14 indications of corruption

- ORA 1578, ORA 600 on user screen or alert log
- inconsistent results
- core dump, trace file generation

6.2.15 Measure to prevent corruption from affected availablility

- hardware redundancy
- hot swappable components
- standby table/database
- fast detection and repair
- mirrored switching

6.2.16 Method and tools to repair

- fix component
- database event like 10231 to skip blocks
- raw retrieval based on existent id
- rowid
- reentry of values

6.2.17 Retain relevant information regarding the corruption

ACTION: we need a form for this: see table 14-3 and need to train for handling corruption

- alert log and errors
- trace files
- heap dump (see page 781)
- events and circumstances:
  - new patch
  - programs running
  - non-database activity
  - database events
  - adhoc statement
  - init param file
  - reproducible test case

Chapter 7

High Availability – getting from 30 days to 3 hours of downtime per year

7.1 Chapter 15: Standby Options

7.1.1 Using a Standby Database for Disaster Recovery

- "conventional approach"
- uses archived redo logs from primary
- both primary and secondary are in ARCHIVELOG mode
- standby mounted, not open
- both db need backup
- at failure, switch and try to apply online redo
- second can have lower resources
- second in a state of constant recovery
  - can open for read over day
  - close at night for recovery
  - means MTTR is higher during business hours
- there are complex issues

Choosing a server

- remote location, not same machine
- **COST:**identical resources to ensure there is no performance degradation
- disk space is critical

- same OS version and patches
- set up network: check bandwidth

Creating the standby database

- same Oracle version
- **PARAMETER CONSTRAINT:**COMPATIBLE same
- DB_NAME DB_FILES DB_BLOCKSIZE
- if resources smaller, tune down DB_BLOCK_BUFFERS, SHARED_POOL, SORT_AREA_SIZE
- init.ora has different params:
  - LOG_ARCHIVE_DEST_<N>
  - LOG_ARCHIVE_DEST_STATE_<N>
  - LOG_ARCHIVE_TRACE (to write second copy of logs)
- **COST:**second copy doubles log space
- primary needs
  - STANDBY_ARCHIVE_DEST
  - DB_FILE_STANDBY_NAME_CONVERT converts paths that are different on two instances)
- same physical structure on primary and secondary: number and location of files, etc.

Implementing the standby configuration

- control file from primary generated
- backups used
- automate via scripts
- ARCH writes to both

Transaction resolution and maintenance

- NOLOG activities are not transferred: need to audit these
- mount of new file not transferred
- need to question commands that don't propagate automatically either manually or by rebuilding standby from the control file
- need to monitor UNRECOVERABLE/NOLOG
- need to monitor structural change from alert log: can analyze old alert logs from the database to see what has changed.

error propagation

- keep 1/2 hr behind
- disable LOG_ARCHIVE_DEST_STATE_<N> using ALTER SYSTEM when the failure is detected
- need monitors that page dba to keep within the 1/2 hr window.
- errors to check for:
  - primary and secondary up and running
  - number of files the same
  - ensure redo log status NOT stale or invalid
  - list critical tables and check
  - ensure enough disks, directories not full
  - network up

switchover

- switch over and going back
- NOLOG activity
- dba training
- well scripted
- apply online redo and determine if it is ok
- implement mechanism to track write to database during switchover (LogMiner product can be bought)
- Rebuild Primary
- RESETLOGS implicit in ALTER DATAGASE ACTIVATE STANDBY DATABASE original cannot be reinstated quickly: an unsupported method of doing this without RESETLOGS exists.
- application failover can be oracle or hardware based

read only access

- writes due to temporary/sort segments **NEEDED FOR REPLICATION:**need locally managed temporary tablespace "EXTENT MANAGEMENT LOCAL"
- **NEEDED FOR REPLICATION:**need to evaluate DBA load with multiple "identical" databases.
- create user on primary database

- grant privileges for queries
- archive redo logs and apply to the database
- startup the standby database in read only
- add tempfile and locally managed temporary tablespace
- log on as user and query
- can swtich to standby

7.1.2 Standby when 24×7not mandatory

- HA in Sun's Sun Clusters
- Oracle configuration pack
- Veritas FirstWatch and ClusterServer
- Database on shared disks of cluster
- Failover kills off session on primary , get on database for 3 minutes, then automatically on the failover
- with OPS, failover instantaneous, needs license, raw devices
- standby has lower cost, maintenance
- but cluster-based is susceptible to environmental problems
- ibm: ping to see if there, auto move and restart manual
- microsoft has a fail safe

7.1.3 architectural alternatives

- primary /standby: node redundancy only
- primary/standby - primary/standby: each node acts as well as standby **FORCDF:**interesting because of peaks, performance can degrade for these short periods: don't have to have 50% extra capacity.

7.1.4 Multiple solutions

- Oracle Parallel Server
- Advanced Replication
- 3rd party

7.1.5 Multiple databases for standby

- one with most recent redo to optimize MTBF
- one that is 4 hours behind **HUH?**what is the usual time before a user error is detected?
- **FORCDF:**for data file catalog ?
- **FORCDF:**emergency at b0 or fcc would be safe so we can analyze if we use replication to some remote machine
- **FORCDF:**what are the licensing issues

7.1.6 OPS and standby

- OPS has enviromental susceptibility
- **FORCDF:**where to place the data warehouse

7.1.7 Availability during routine maintenance and emergencies

- Standby does not address
  - availability during maintenance and upgrade: Patch invalidates standby
  - user errors
- Use a custom solution
  - basic replication
  - triggers and journals help avoid the higher overhead normally associated with Advanced Replication (pg 815)

7.1.8 Custom Standby Database (CSD)

FORCDF:much like our replication

- complete db on another system
- push and pull with journal and trigger
- no need to duplicate system, schema
- no need for perfect match of db's
- uses a replay of the transaction so no version dependence
- code journalize into API
- employ triggers to write to a second table

steps to create a CSD

- create table on secondary database
- data copied
- allows critical segments to be identical and duplicated
- let db lag for user error protection
- ALTER/TRONCATE things not propagated
- DDL changes are not propagated
- physical structural changes need not be propagated
- must create a separate journal table
  - in a special administrative schema
  - write trigger to insert into journal table
  - DBMS_JOB runs on a second database to play back recorded transactions

Steps to switch over to a CSD

- disable DBMS_JOB
- manually apply final transactions
- route clients to secondary
- disable primary
- problem is there are still users on primary
- problem with last-second transactions
- either turn off for a few seconds or pop in the last few transactions
  - best to take outage instead of risking corruption
- Upgrade pattern of two machines, A, B running version 7 and wanting to go to version 8 goes like:
  - A has 7, B has 7
  - switch to B
  - A has 7 to 8, B has 7
  - switch to A
  - A has 8, B has 7
  - A has 8, B has 7 to 8
  - reset the the play

  - A has 8, B has 8

- failover unavailable during migration or upgrade
- performance loss in this method is 5 to 10%
- database links are inefficient **FORCDF:**need to check performance when we are playing with imports/exports from production to integration or development.
- may use as a temporary measure for a migration
- availability vs. performance so need to increase the capability of the system to handle this
- we can use basic replications both ways, aside from the table/view naming conflict.

7.1.9 Case Study on Implementing CSD

Scenario

- 3 DB's, one is 24×7: "PT05"
- "PT05" has three schemas
- clear 24/7 but in a specific period
- evaluate: disk failure, segment corruption, user error, maintenance

Problem

- cannot upgrade
- many users make mistakes

CSD

- used backup to create database to avoid performance strain
- logical delta to get journal table in place. took 15 min, allowed 3h
- applied archive redo to secondary server
- put on trigger but disabled on secondary server
- DBMS_JOB created on secondary database

Incident

- Data files became inaccessible: deleted by accident from ARCHIVELOG
- **FORCDF:**we can test this
- Did a switch over: took 6 minutes downtime
- Result: lots of practice needed for this!

7.1.10 Standby Tables to cater to segment level failures

FORCDF: This is a possible solution for the common tables in the data file catalog and for the calibrunlists tables. It provides security against corruption

- want certain tables to be changed
- keep copies of critical tables
- to determine which tables
  - criticality: core tables getting most action
  - size: some are tiny (**SCALE Reference:**large $=10^5$ rows, small$=10^3$ rows)
  - activity

How standby tables work

- core tables
- separate schema, separate disk/controller
- table mirrors in admin schema
  - same name, indexes, constraints
  - must evaluate triggers
- create journal tables
  - keeps all transactions
  - same schema as standby table
- privileges are same as original, not more
- public synonym handling depends on usage:
  1. all log on as the user **FORCDF:**we don't
  2. private synonym **FORCDF:**we don't
  3. direct reference
  4. public synonym
- scheme works fine for 1 and 2
- for 3, need the same schema and names like table_name_STDBY, but cannot have automatic switchover
- for 4 have a conflict in names, so need to make all public synonyms private**FORCDF:**can control at API

Segment Failover

- rename to table_name_OLD, and the mirror is taken instantaneously
- if dropped by accident, the mirror takes over automatically
- uses fewer physical resources and can mirror with time lags to handle user errors

Keep Original and standby tables synchronized

- via triggers on both
- copy all writes, dormant on standby
- trigger on standby copies write to a journal to allow the original to be brought up to date later

DDL changes

must be applied manually

Example

- Foreign key points to original if no copy or to a copy if there is one
- use journal table to switch back, but must be during no activity so that new entries are not made:
  - take a small outage: recommended - clean, no logical corruption
  - make tables unavaliable
- advise users
- disable trigger on standby table
- truncate journal table
- enable primary table trigger
- make tables available

Requirements of Standby Tables

- for warehouse, no journal, just do bulk writes to both tables
- standby must have robust security so nobody else write to it
- backup mirror schema
- recover standby tables last: need to document them for an out of house DBA to deal with.**FORCDF:**a good idea to document to this level

7.1.11 Case Study for Stand By Tables

Setup

- 4 databases: 2 DSS, 1 OLTP with 3 schema, 1 standby (15 min update)
- 24 tables
- 6 used a lot and they had fragmentation and corruption

Intermediate Solution and need for better

- 30 min on redo copy to allow thinking
- control over propagation
- insert, update, delete, select privileges to schema owner: ORDER_ENTRY
- dropped public synonyms
- made journal tables
- test by manual recreation of problems
- use API to page when there is a failure so switchover can be done quickly **FORCDF:**: use ACE's as first line of defense
- on call DBA sent mail: drill to make sure that the persons take responsibility **FORCDF:**need such drills and lists
- **FORCDF:**must set up a shift rota
- **FORCDF:**HWDB susceptible as well as CALIBRUNLISTS

What the Responding DA does

1. Check if problem merits switchover and is not intermittent
2. Run switchover script targetted at table if:
    - Enables trigger on the standby table to populate the journal
    - Renames the affected table to table_name_OLD
    - Disable the trigger on table_name_OLD that populates standby
3. Ensures application runs smoothly
4. Diagnoses cause/effect and extent of damage to primary table
5. Repairs Damage
6. Tests tables
7. On next chance, switch back

Switching Back

- manual
- 2 DBA's to check one another
- Replay transaction from journal and delete those replayed
- stop application and run a script to
  - apply rest of transactions
  - restore standby to table_name_STDBY
  - disable trigger to update to journal
  - rename orignal to correct name
- restart schema
- test/validate

Result and Lesson

- 10% impact on performance until tuning was done
- few minutes downtime
- allows database to be open during corruption repair
- allows migration
- physical architecture repair
- enhances design
- change ownership
- change physical storage characteristics

Caveats

- rely on journal table: if it is corrupt, you are sunk
- may need to relax constraints

7.2 Chapter 16: Oracle Parallel Server

Note:requires raw devices: implications in Section 3.3 Also, seems rather more like hype than experienced advice.

7.2.1 What OPS is and how it enhances availability

- multiple instances that refer to the same physical data files
- all nodes (instances reside on nodes) share sibling relation
- synchronize nodes with a distributed lock manager (DLM)
- one down node is ok, rest still run
- PMON, DBWR, SMON CGWR, ARCH run on 1 local disk for each node
- data on shared disk

Primary advantages of OPS

- automatic recovery
- load balancing
- high capacity and scalability **FORCDF:**think about offline

Caveats

- disk failure susceptibility
- performance degradation if applications not OPS-aware
- **COST:**increased maintenance
- more system complexity
- **COST:**cost

7.2.2 OPS architectural components and operations

- open SHARED PARALLEL mode
- own init.ora on each node
- shared params in config.ora

Unique components

- DLM: oracle product
- adds structures to include messages, locks , cache management to SGA: they are the ones the multiple nodes are likely to request
- datablocks must be informed amongst nodes
- system resources available
- interprocess communication

- failure detection
- uses locks: Queues are GRANTED (ie. obtained) and CONVERT (wait)
- "DLM database" contains information on locks and other structures:
  - lock Daemon (LD) : "keeper" of the DLM
  - lock Monitor (LM) creates DLM database, opens channels, creates structures, collects dead processes and sleeps **HUH?**sounds like the job for me
  - connection manager (CM): tracks node status and sends heartbeat messages

DLM database

- Locks
- who wants what
- nodes
- Attempts to establish node affinity with lock database
- **SCALE Reference:**two to four nodes usually

Directory Tree

- information on resource and nodes that carry them

LCKN, LMON, LMD0

- Lock process LCKn manages locks: can start up more if much activity
- LMON: reconfigures global lock
- LMD0: redirects requests

DML locks

- Parallel cache management, PCM or non-PCM.
- PCM allows one node to access data buffer of another
- non-PCM, global Enqueues, share data files to objects

PCM locks

- null at startup
- DLM makes structures, masters the resources
- GC_* sets lock on
- GC_* rounded up to nearest prime number

- **MONITOR TASK:**Can examine lock allocation: pg 864
- Has to get from DBlock to GC lock

Non-PCM

- dynamic

7.2.3 Automatic Failure Recovery

Node or network:

Node

- CM checks nodes
- if timeout, failure noted
- connection marked broken, cluster reorganized
- if repaired CM detects the node is back picks it up

Reubilding the DLM Database

- Rebuilt after cluster reorganization
- LM initiates this
- locks frozen, PCM locks invalidate, survivors reacquire locks

7.2.4 Instance Recovery

- Like crash recovery, but performed by another surviving instance
- restore cache: cache recovery
- restore transaction: transaction recovery
  - write dirty buffers
  - switch over users/application from dead to survivor
  - SMON posted to do instance recovery
- in cache recovery PCM locks are checked:
  - redo logs applied to dubious locks on data
  - dirty bufffers written
  - uncommitted writes rolled back

7.2.5 Database Availability during Auto Fail Recovery

Mostly unaffected except:

- DLM database rebuilt
- cache recovery
- locks from failed database

7.2.6 Understanding, detecting and eliminating false Pinging

- Keep lock on local buffer until another requests; since likely that local node will ask for the same data again
- Another node requests and the one with the locks writes out the blocks: internode pinging results.
- CacheFusion allows shipment directly from one machine to the next without the disk
- if two nodes are doing a Read/Read operation, this is ok since they can use shared locks
- problem is that if one does read/write and the other write/write. The locked, request made, written to disk so no all reads or read write.
- multiple or soft pinging: when multiple blocks to one lock, those are written as well.
- **MONITOR TASK:**:
  - $<$ 100% false may occur
  - $=$ 100% certain that there is false pinging
  - $>$ 100% false pinging occurring extensively
- to eliminate:
  - inadequate number of PCM locks: each lock gets lots of blocks
  - allocate as many locks as dirty data blocks
  - .. so you need to know application usage.
  - locks consumer system resources
  - inadequate partitioning

7.2.7 Partition Application Prior to Deployment in OPS

- segregate like-usage segment between instances
- partition by application
- partition by user
- by hours of usage

- have lots of tablespaces
- segments with heavy read/write get separate tablespace
- group segments used by one instance together
- stripe and spread out large segments with multiple users
- row range partition or tables
- lump readonly together
- define strange params for a table to keep rows in a block: small for popular tables: **PARAMETER CONSTRAINT:**PCTFREE 99
- reverse indexes
- multiple free list groups
- different sequences for different instances

7.3 Chapter 17: Advanced Replication

The old versions seemed quite clumsy and resource intensive. The author seems to be selling the party line and is possibly not convinced himself.

- sychronization: multi-master, peer to peer
- some row changed by different users can cause conflict
- conflict resolution options:
  - time (first)
  - average
  - size
  - min or max value
- retain transaction in a local deferred queue if target is unavailable
- DML, DDL propagated
- uses triggers and journal tables
- propagates DDL

7.3.1 Example

- user makes change
- before, after image made.
- before image used to determine if there is a conflict
- events pushed (depends on if immediate or event-based)
- SNP process uses 2 phase commit and RPC
- apply to second master
- conflict found: before image mismatch
- error logger entry
- conflict algorithm applied
- propagates according to decision

7.3.2 The old Version

- serial
- whole row
- all groups or none
- updatable snapshot not possible
- LONG not replicated
- configuration complex

7.3.3 8i Version

HUH?please pass the snake oil

- enhance performance and management
- lower overhead
- mimimize elements

Init of trigger and packages

- c code
- tamper-proof

reduce data element propagation

- Use a key
- new column added
- values for conflict detection: time stamp

Parallel data element propagated

- multiple streams
- uses same mechanism as PQO
- **PARAMETER CONSTRAINT:SCALE Reference:**PARALLEL_MIN_SERVERS (=number of processes)
- **PARAMETER CONSTRAINT:SCALE Reference:**PARALLEL_MAX_SERVERS

Fine grained replication group quiescing

- single group offline allowed
- need to give forethought to how to group

Replication of partitioned segments

- Partitioned subset not directly supported
- use updatable snapshots
- global indexes are not supported

LOB support

- BFILE not supported
- **PARAMETER CONSTRAINT:**MIN_COMMUNICATION parameter

Replication Manager

- sales pitch=: GUI tool in OEM

Security

- Net8 encryption and security **HUH?ACTION:**

7.3.4 Options for HA AR implementations

Active/Passive

- source active, passive copy readonly
- can fail over easily

Active/Active

- both running: scalability and load balancing
- remote process communication overhead will cut performance

7.3.5 Be Aware of Possible AR architectural Alternatives

Master/Master

- entire database or mission critical
- Active/Passive or Active/Active
- **NEEDED FOR REPLICATION:**for high transaction enviroment not recommended to have more than two masters

Updatable Snapshots

- for case with only certain tables or subsets
- updatable at either site

Data Partitioning via snapshots

- horizontal and vertical
- H: certain rows (where clause) **FORCDF:**calibration status
- V: limit columns: protect against structural change

Other features

- subquery subsetting: avoid script writing at remote sites
- primary key snapshots: not rowid, snapshots preserved during reorganization
- automatic snapshot registration
- constraints on snapshots
- hybrid configuration: master-master and updatable
- synchronous replication
  - real time

  - difficult to implement
  - need large bandwidth
  - network down, causes write to fail
  - must have a LAN

7.3.6 Procedural Replication

- replicate in batch at master not prudent
- procedure invoked at remote site

7.3.7 Mass Deployment

FORCDF:

- if number of users is immense and geographically spread out
- usual manual snapshot cumbersome
- "deployment template"
- "configure once, install many times"

7.4 Chapter 18: Third Party HA

Come in flavors of :

- Hardware/OS
- Database-based

7.4.1 Hardware/OS Based

- EMC SRDF for disaster recovery purposes
- SRDF = Symmetric Remote Data Facility
- geographic mirroring
- external to Oracle
- uses Fiber optic connection
- primary disks in read/write, remote are readonly
- no impact on host: does not use server
- utilization of write back caches: write to cache, not disk: fast
- concurrent support for multiple platforms: UNIX, NT

Replication Options

- Synchronous i/o to both systems. If secondary fails, write goes only to primary
- synchrounous w/domino attribute: i/o freezes if secondary fails
- semi-synchronous
  - first write primary, then second done
  - no hold of primary if second fails
  - no wait, usually
- adaptive copy
  - No wait for acknowledgement
  - high degree of interdependence
  - can lead to logical inconsistency
- Mix: use required io for two nodes, adaptive for third.
- Recommendation: synchronous

Failover during Disaster Recovery

- Fallback after normalcy: secondary, now primary, exposed to failure
- need to resychronize with SRDF routines
- time consuming
- original primary may fail, leading to a rolling disaster

Caveats

- user and admin errors are copied
- 2 to 10% overhead (even up to 15%)

7.4.2 EMC Timefinder for high retrun on investment

FORCDF:this needs serious investigation as an entirely different alternative to oracle replication.

- BCV: Business Communication Volume added: additional mirror
- can also use for backup
- pair a disk with BCV
- can use BCV for
  - Real time DSS then recharge after use

  - backups
  - load warehouse
  - as a test system
- rapid synchronization: done at bit/byte level

Caveats

- BCV establish during low use
- aid external maintenance (backup disk reorg) but not index rebuild or defragmentation
- need to be sure they are a consistent copy if used for restore

7.4.3 Database Based: Quest Shareplex

- log based replication
- reconstruct SQL from redo logs
- order kept

Architecture and Operation

- controller, daemons, queue files
- tables on oracle database
- replication stops if a daemon dies
- "capturer" and "reader" on source: done incrementally for to lessen bursting
- set of files do queueing
  - DML captured and recoreded
  - SQL compressed
  - Queues up writes
- transfer
  - use import/export of ShorePlex
  - use proprietary high speed transport
  - no reliance on SQL NET
  - network pooled if a failure
- SQL writen to files on source system: need disk space for queue
- "post" process reads files

Characteristics

- Secondary database concurrently available
- query data without impact on production
- temporary segment restriction not there
- custom procedures can denormalize if desired

Support for multiple concurrent OS/Oracle Versions

- different versions at two sites
- solaris

Support for multiple replication scenarios

- peer to peer: secondary in readonly
- no writes in secondary but not blocked
- require manual resolution of conflict

Instantaneous Failover

- Application/user failover: how much of the latest data are you willing to loose?
- consideration the following failure scenarios
  1. writes to online redo logs not captured
  2. write captured and in queue but not propagated
  3. writes propagated but not applied to target
- Item 3 gets done, but the other two only after restoration

Planned Outages

- defragmentaion, migration, OS upgrade, database patch
- smooth switchover: redirects users to secondary

Other Features

- Broadcast replication: single database replicates to multiple
- Consolidation replication: multiple primary to single bigger secondary
- Multitier: middle tier need not be a database. Replicate from primary to middle and middle to rest
- master-master: both are primary and need conflict resolution

- heirarchical: grandfather-father-grandson
- choose certain tables
- LOBS
- allows referential integrity to be held
- add new segment dynamically

caveats

- ready made secondary database required during intial configuration needs
  - import/export
  - -or- "reconciliation utility" and hot backup
  - -or- use EMC TimeFinder
- Table locks during intial configuration: shareplex runs analyze when a new table is added
- inability to read archive redo: if capture process dies, can miss data
- NOLOGGING/UNRECOVERABLE not propagated: need to carefully evaluate use of NOLOG
- Disable Triggers on Target: script does this
- Disable Data Dictionary Replication: an application-related segment can be replicated
- performance: not instrusive

Chapter 8

The software toolkit to monitor

8.1 Chapter 19: the 24×7toolkit

MONITOR TASK: This chapter has lots of goodies that just need to be coded up

- Monitor specific failures with polling scripts: logfile organization
- check MAXEXTENT, NEXT extent, TEMP extent
- need to add other events
- check books, magazines, web
- lock, whatlocks, findalllocspace
- find freespace, sessions, cur sql etc...
- Techniques for application failover when calling PL/SQL: page 940

8.2 Chapter 20: HA in 8i

Beware the Bleading Edge

8.2.1 Maintenance Oriented Features

- migrating
- troubleshooting
- segment rebuild

Transportable segments for quick movement of data subsets

- archive prior to purge
- migration to OPS or Datawarehouse
- production to test

- warehouse to downstream
- **FORCDF:DEVELOPMENT TASK:ACTION:**may be a good way to populate dev/int
- better than export with SQL Loader: perhaps why SQL Loader is broken
- uses cp
- **PARAMETER CONSTRAINT:**export has TRANSPORT_TABLESPACE, DATAFILES, TTS_OWNERS
- same blocksize and character set at source and target
- functional and physical dependency required: "self-contained" tablespace
- DBMS_TTS_TRANSPORT_SET_CHECK can be used

Locally Managed Tablespaces to prevent fragmentation

- Dictionary managed tablespaces
- **HUH?**
- locally managed, extent management done by table
- **FORCDF:PARAMETER CONSTRAINT:**interesting, but what is this?

Create, Rebuild defragment indexes online

DEVELOPMENT TASK:interesting test

- online to CREATE/ALTER INDEX
- DDL not allowed
- parallel DMA not allowed
- rebuild vs. coalesce
  - rebuild redoes indexes, gets rid of corrupt indexes and is preferred
  - coalesce uses existing index and won't remove corruption

Non-Partitioned Table Reorganization

- ALTER TABLE ... MOVE
- allows movement to a new segment
- IOT are prime candidates
- no DML

Column Dropping

- no import/export
- ALTER TABLE SET UNUSED or DROP UNUSED
- don't get back the disk space but faster

Partitioning Enhancement

- Merge: drop or exchange needed and requires import/export: ALTER TABLE ... MERGE PARTITIONS
- Updatable partition keys allow migration across partitions but can cause high i/o
- **HUH?**Composite Partitioning
- Partitioning IOT's and Tables with LOB:
  - sqlloader use DIRECT
  - partition key cannot be a LOB
  - LOB can be partitioned

Other

- **MONITOR TASK:**Change Management Features: can collect stats on CPU

8.2.2 Troubleshooting Features: Repair corruption

DBMS_REPAIR detects and repairs

- indexes
- makes repair table
- not usable fo LOB, bitmapped index, function-based indexes

8.2.3 Backup/Recover

RMAN

- Proxy Copy
  - move to tape
  - drive via db management console
- database cleaning
  - done with duplication
  - can give a different name
- Duplex backup sets produce and execute copies

Control and Expedite Recover Time

- fast checkpoint
- rollback phase
- move checkpoint to limit dirty buffers

fast start on-demand rollback

- defers rollback phase
- allows aborted transactions to be done on demand

Miscellaneous

- fast start parallel rollback:SMON, only if beneficial
- use newer listener failover and load-balancing services
  - net8i
  - listener finds least burdened database
- **MONITOR TASK:FORCDF:**LogMiner – data file catalog
  - tool for reading redo logs
  - log corruption detection
  - can get SQL back
  - can get SQL UNDO statement
- process to elimimate "fat" OS overhead
  - ok on intel
  - micro kernel from sun
  - OEM pack integrated: predefined events and jobs
  - can use CD to restore oracle from failover
  - NEW AND UNPROVEN

Bibliography

[1] "Oracle 24×7", V.S. Devraj, Osborne/McGraw-Hill, 2000.

[2] Cindy Wike, FNAL, Private communication.

[3] Chris Green, Purdue University, Private communication.

[4] "Making the Decision to USE UNIX Raw Devices", Cory Millsap, Oracle support document.

[5] Loney, Kevin "Oracle8 DBA Handbook" Berkeley, CA: Oracle Press/Osborne/McGraw-Hill, 1998 Chapter 4.