

# **Überblick über die Architektur des Oracle Datenbanksystems und daraus abzuleitende Tuning Ansätze**

Vortrag SAGE Rhein-Main  
Darmstadt, 26.09.2002

M. Mann  
ONSYS GmbH, Ludwigshafen  
matthias.mann@onsys.de

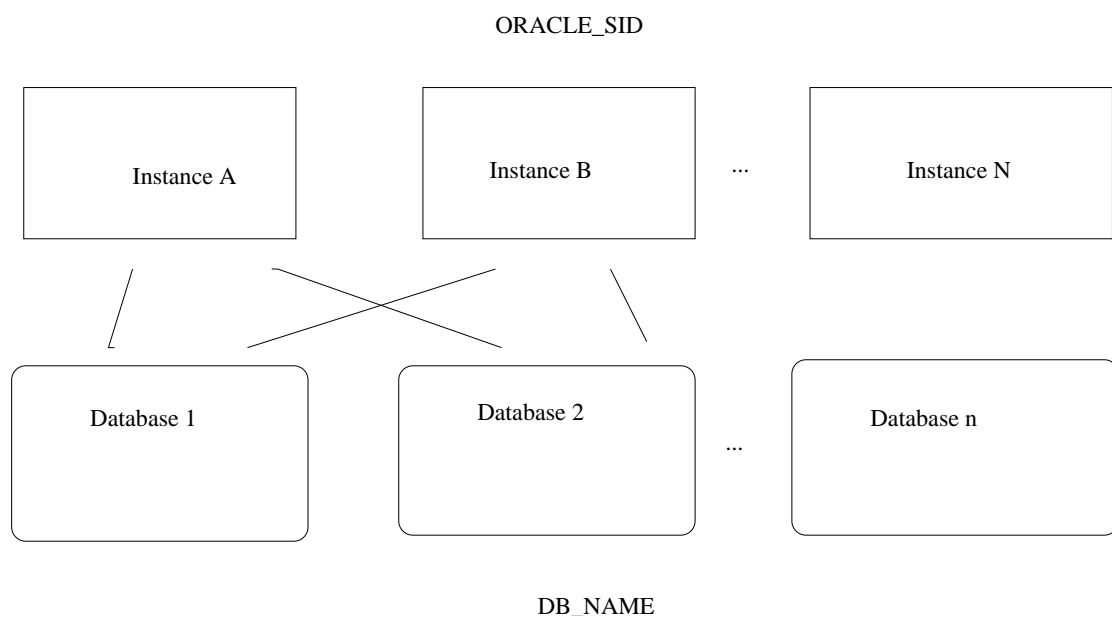
## Inhalt

- 1 Oracle Architektur im Überblick
  - 1.1 Komponenten des Datenbanksystems (DBMS)
  - 1.2 Clientprozesse und DBMS
  - 1.3 Verknüpfung von Instance und DB
  - 1.4 Struktur der DB
  
- 2 Tuningebenen eines Oracle DBMS
  - 2.1 Überblick
  - 2.2 Memory
  - 2.3 Input/Output und Filelayout
  - 2.4 CPU
  - 2.5 Weitere Oracle Spezifika
  - 2.6 OS Spezifika

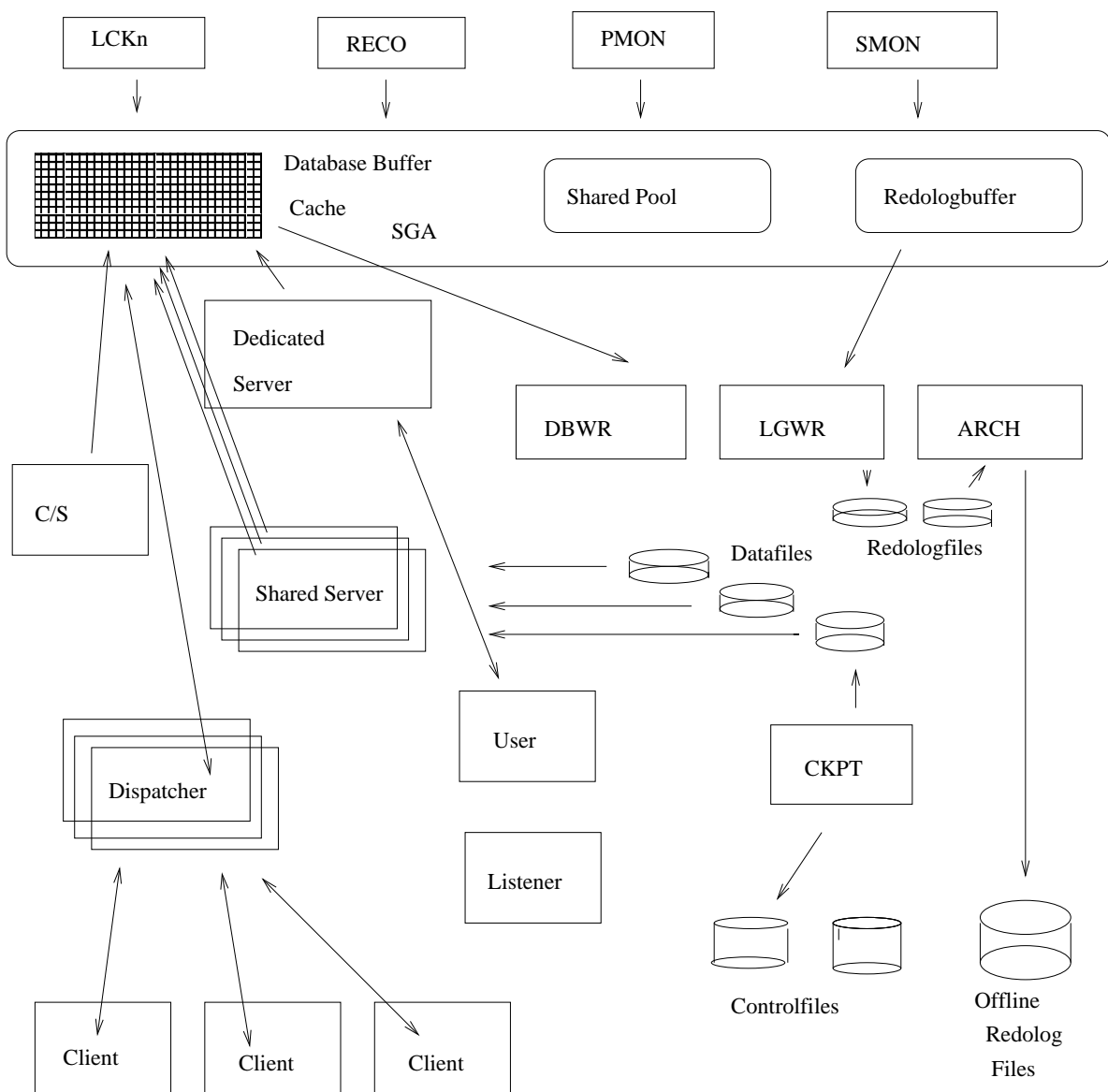
# 1 Oracle Architektur im Überblick

## 1.1 Komponenten des DBMS

DBMS = DB (passiv) + Instance (aktiv)



# 1.1 Komponenten des DBMS (2)



## 1.1 Komponenten des DBMS (3)

- Clientprozesse kommen auf verschiedenen Wegen an die DB
- Database:
  - Datafiles (Benutzerdaten und Data Dictionary)
  - Redologfiles (Protokollierung)
  - Control Files (Datenbank Struktur)
- Instance: System Global Area (SGA) + Background Prozesse

## 1.1 Komponenten des DBMS (4)

### Clientzugriff auf DB

- lesend: Daten (Blocks) werden in SGA (Buffer cache) übertragen
- schreibend:
  - zentrale Designentscheidung: Clients schreiben nicht selbst in Datafiles
  - warum? Glättung von Aktivitätsspitzen, kein synchrones Schreiben
  - wer schreibt? Database Writer (DBWR)
  - wie? zyklisch, LRU Prinzip, Huckepack, Deferred Write

## 1.1 Komponenten des DBMS (5)

### Redologmechanismus

- Problem bei Systemabsturz (Inkonsistenz der DB):
  - geänderte Daten (COMMIT) sind noch nicht in DB
  - geänderte Daten (kein COMMIT) wegen Huckepack bereits in DB
- deshalb Erweiterung der Funktionalität: Redolog Buffer, Logwriter (LGWR), Redologfiles
- Client DML ändert Daten in SGA
- geänderte Bytes kommen in Rollback Segmente (ebenfalls in SGA vorgehalten)
- geänderte Bytes kommen ebenfalls in Redologbuffer (Ringbuffer)
- LGWR schreibt periodisch (bzw. bei COMMIT, dann synchron) Redologbuffer in Redologfiles
- es entsteht ein Protokoll aller Änderungen, dann Löschung im RBS

Abgeschlossene Transaktionen stehen immer auf Platte (Redologfiles) und sind somit vor Systemabsturz geschützt.

Der Aufwand für den Redologmechanismus ist weit geringer als der Gewinn durch das asynchrone Schreiben des DBWR.

## 1.1 Komponenten des DBMS (6)

### Process- und Instance - Recovery (1)

Unterbrechungen des regulären Datenbankbetriebs:

- (1) irreguläre Beendigung eines Clientprozesses (Process Failure)
- (2) irreguläre Beendigung eines oder mehrerer Hintergrundprozesse (Instance Failure)
- (3) Zerstörung externer Speichermedien (Media Failure)

Wie und durch wen erfolgt Recovery?

- (1) Process Monitor (PMON): Freigabe von blockierten Ressourcen, Rollback unfertiger Transaktionen
- (2) System Monitor (SMON): Überwachung der DB Konsistenz, Instance Recovery

## 1.1 Komponenten des DBMS (7)

### Process- und Instance - Recovery (2)

#### Redolog Mech. während des DB Betriebs:

- Online Redolog Files (RL1, RL2)
- LGWR beschreibt RL1
- RL1 ist voll
- Log File Switch und Checkpoint
- LGWR schreibt in RL2 und DBWR schreibt alle geänderten aber noch nicht in DB befindlichen Blöcke aus RL1 in DB (also auch die oft benötigten Blöcke (LRU) aus der SGA)

Alle Aktionen werden in die DB übertragen, bevor der LGWR das Protokoll überschreibt.

#### Phasen des Instance Recovery:

- Roll Forward: alle Aktionen in den RL werden nachgefahren
- Roll Backward: unvollständige Aktionen (Roll-back Segmente) werden zurückgerollt

DB ist dann konsistent. Oracle garantiert die Permanenz der Transaktionen (COMMIT), nicht aller durchgeführten Aktionen.

## 1.1 Komponenten des DBMS (8)

### Archivierung und Media Recovery (1)

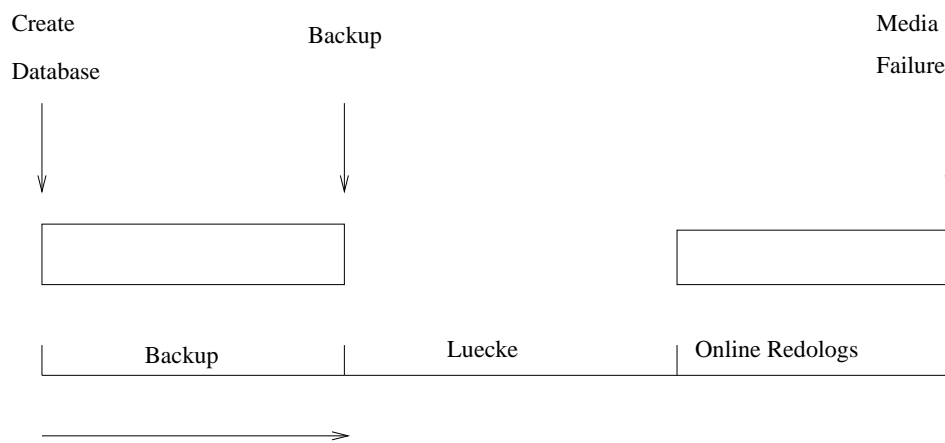
Media Recovery liefert konsistente DB bei:

- Einspielen eines konsistenten (aber nicht aktuellen) Backups
- Nachfahren der Offline Redologs
  - Archiver (ARCH) im optionalen Archivierungsmodus kopiert die Online Redologs in Offline Redologs (permanent)
- Nachfahren der Online Redologs
- Rollback der unvollständigen Transaktionen

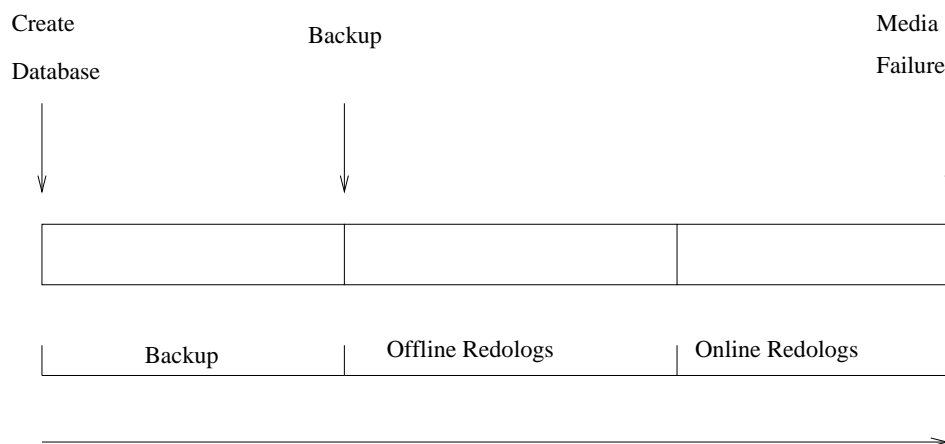
# 1.1 Komponenten des DBMS (9)

## Media Recovery (2)

### 1. ohne Archivierung

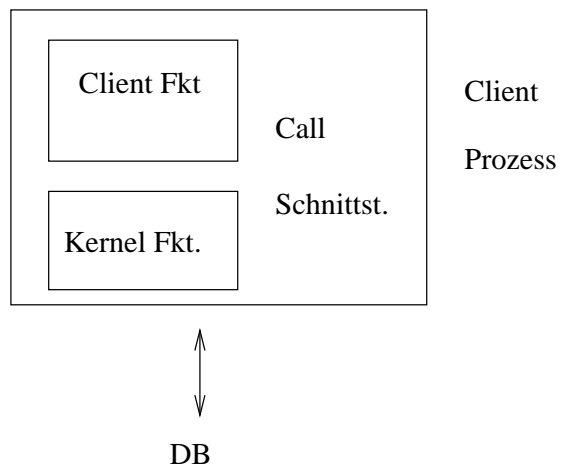


### 2. mit Archivierung

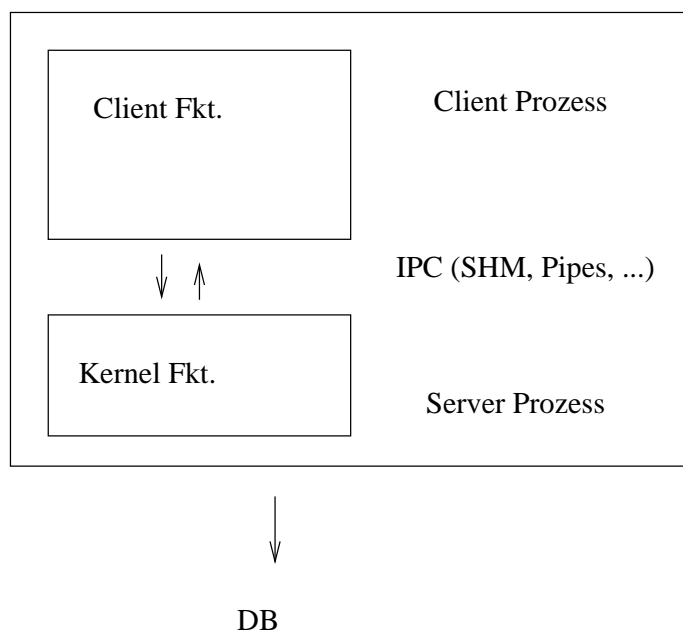


## 1.2 Clientprozesse und DBMS

(1) Single Task: schnell, unsicher

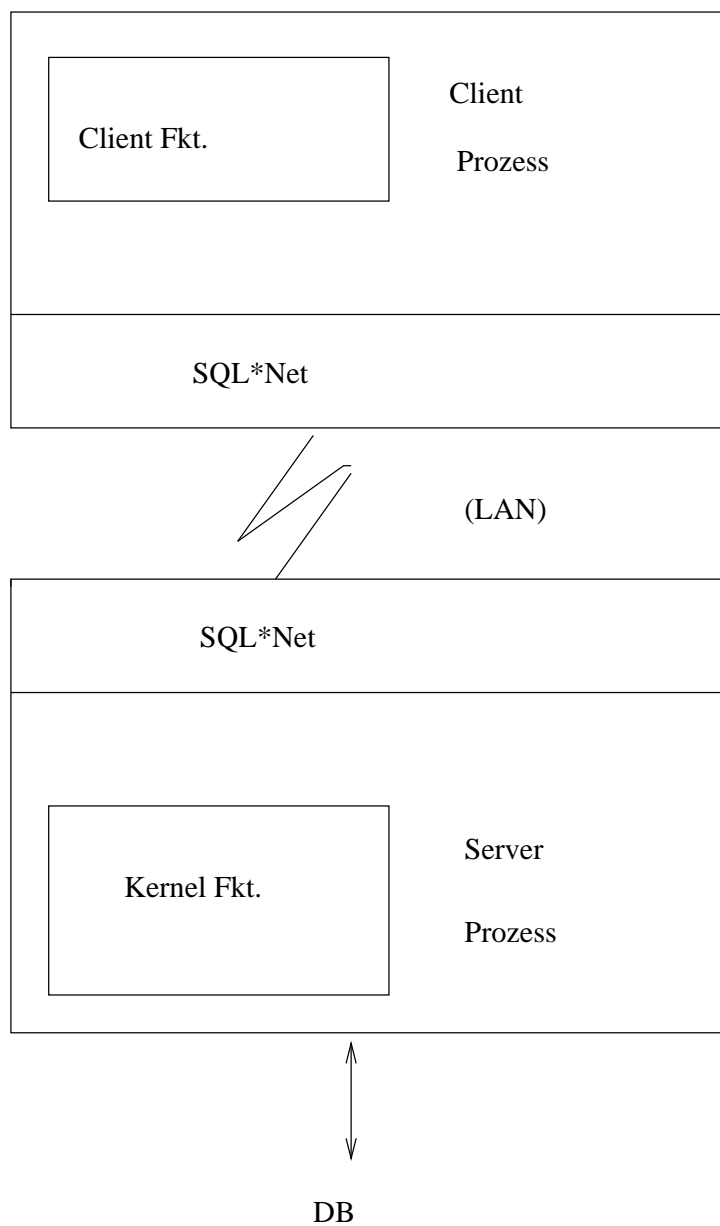


(2) Two Task (langsamer, sicherer)



## 1.2 Clientprozesse und DBMS (2)

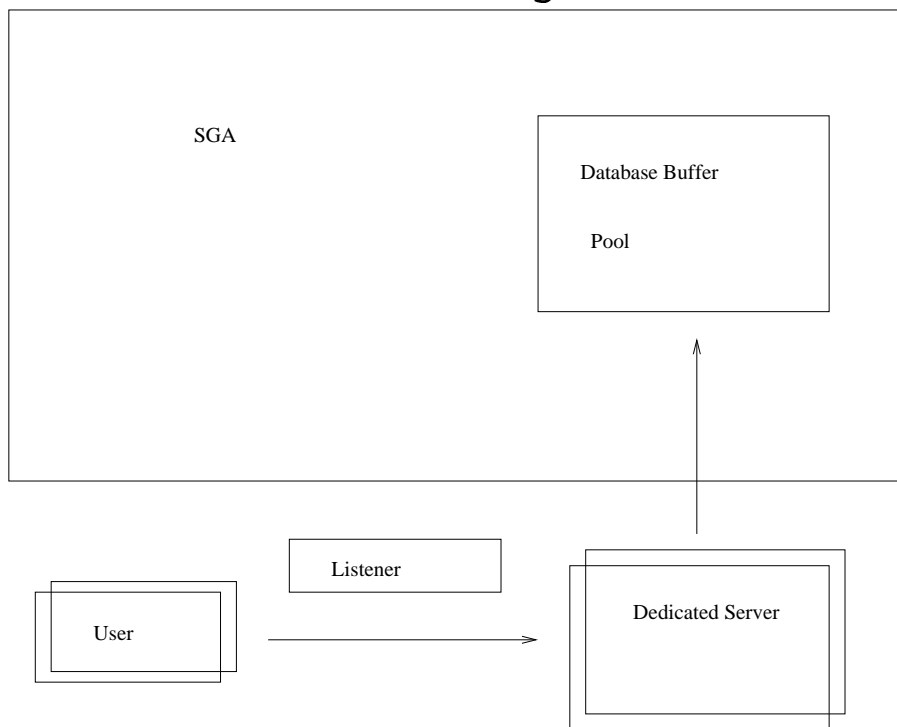
### (3) Client/Server Architektur



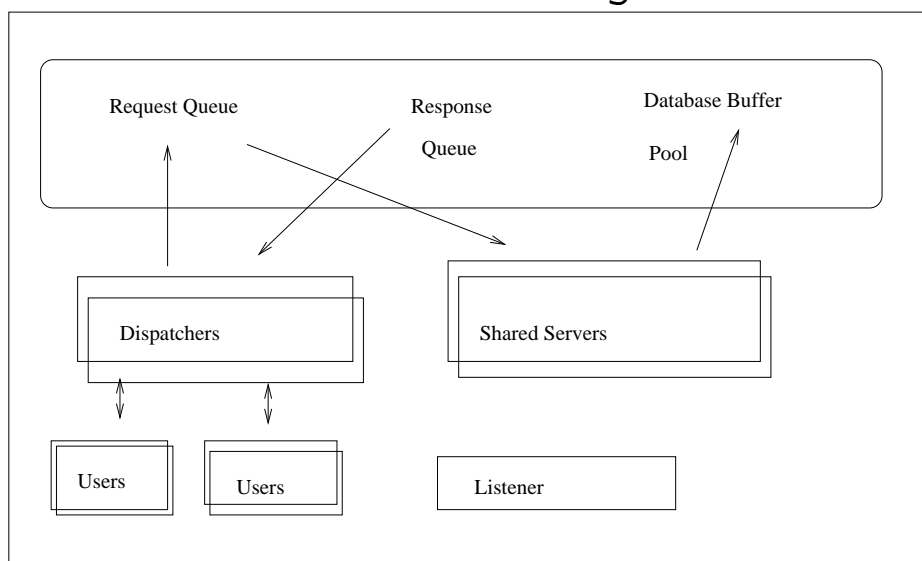
## 1.2 Clientprozesse und DBMS (3)

### Verwaltung der Serverprozesse

#### Dedicated Server Konfiguration



#### Multithreaded Server Konfiguration



## 1.2 Clientprozesse und DBMS (4)

### Speicherstrukturen für Clientprozesse (1)

- auf Instance Ebene
- auf Prozessebene
- auf Statementebene

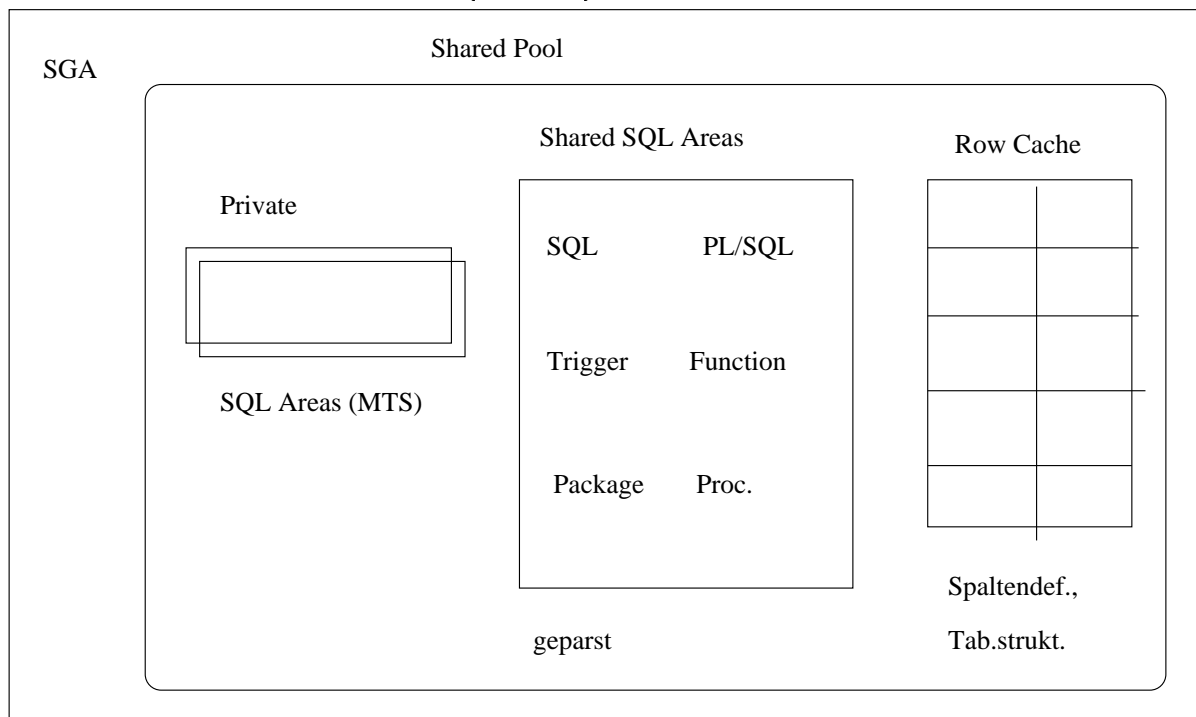
Ausführung von SQL Statements:

- beim erstmaligen Ausf. innerhalb eines Client Proz.: Allokation einer Private SQL Area
- Prüfung auf Shared SQL Area im Shared Pool
- evtl. Ablegen des geparsten Statements in Shared SQL Area

## 1.2 Clientprozesse und DBMS (5)

### Speicherstrukturen für Clientprozesse (2)

#### auf Instance Ebene (SGA)



shared\_pool\_size

Private SQL Areas:

- persistent area
- runtime area

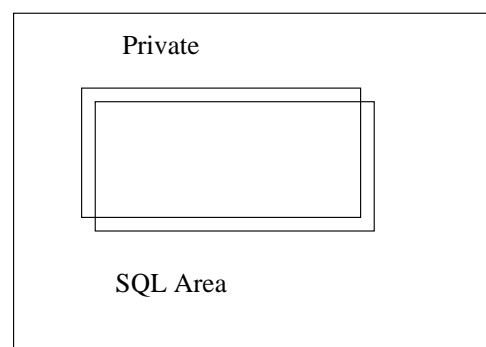
Shared SQL Area:

- SQL
- program units

V\\$\$SQLAREA, V\\$\$ROWCACHE, V\\$\$LIBRARYCACHE

PGA

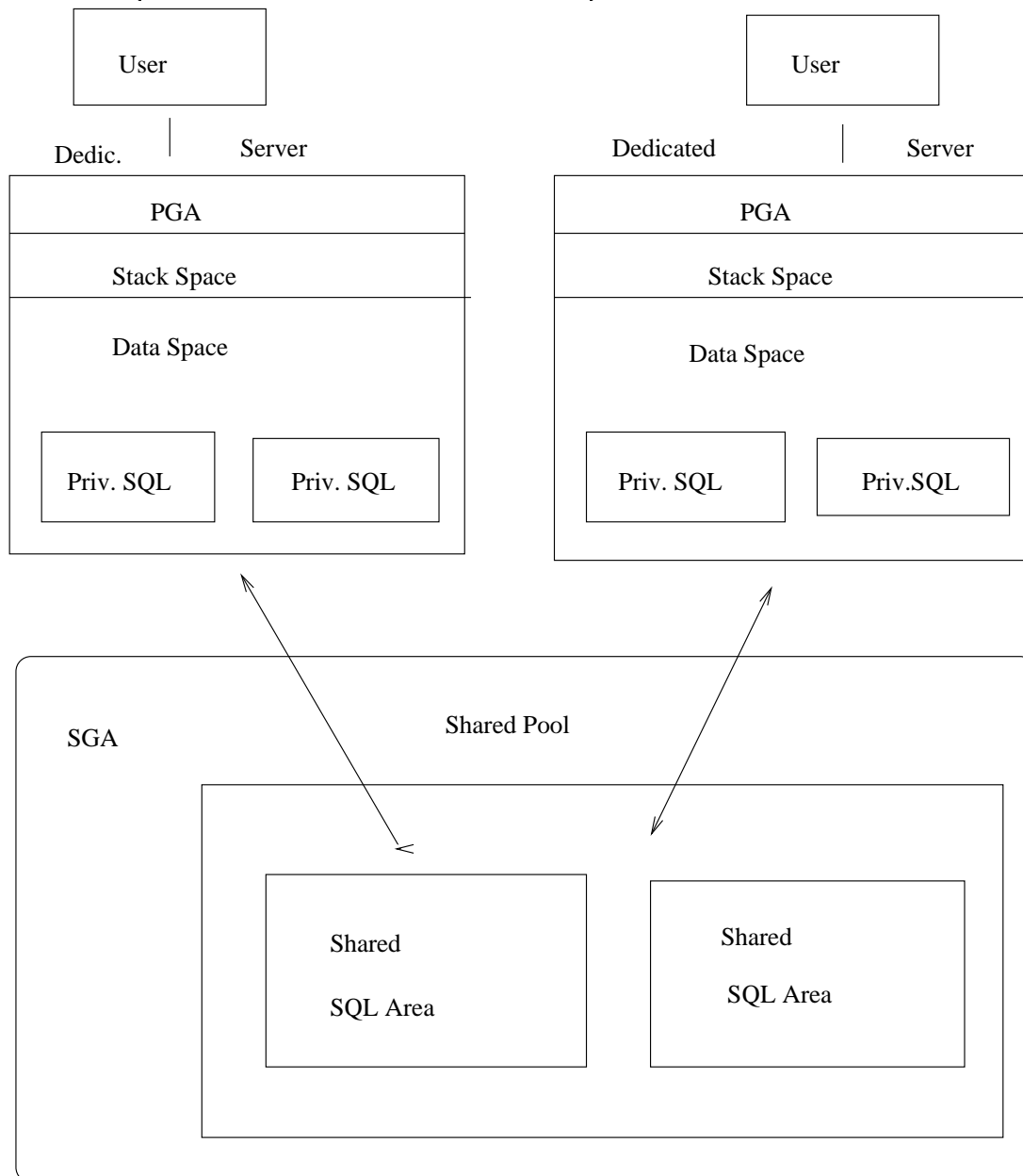
Mod. LRU Algor.



## 1.2 Clientprozesse und DBMS (6)

### Speicherstrukturen für Clientprozesse (3)

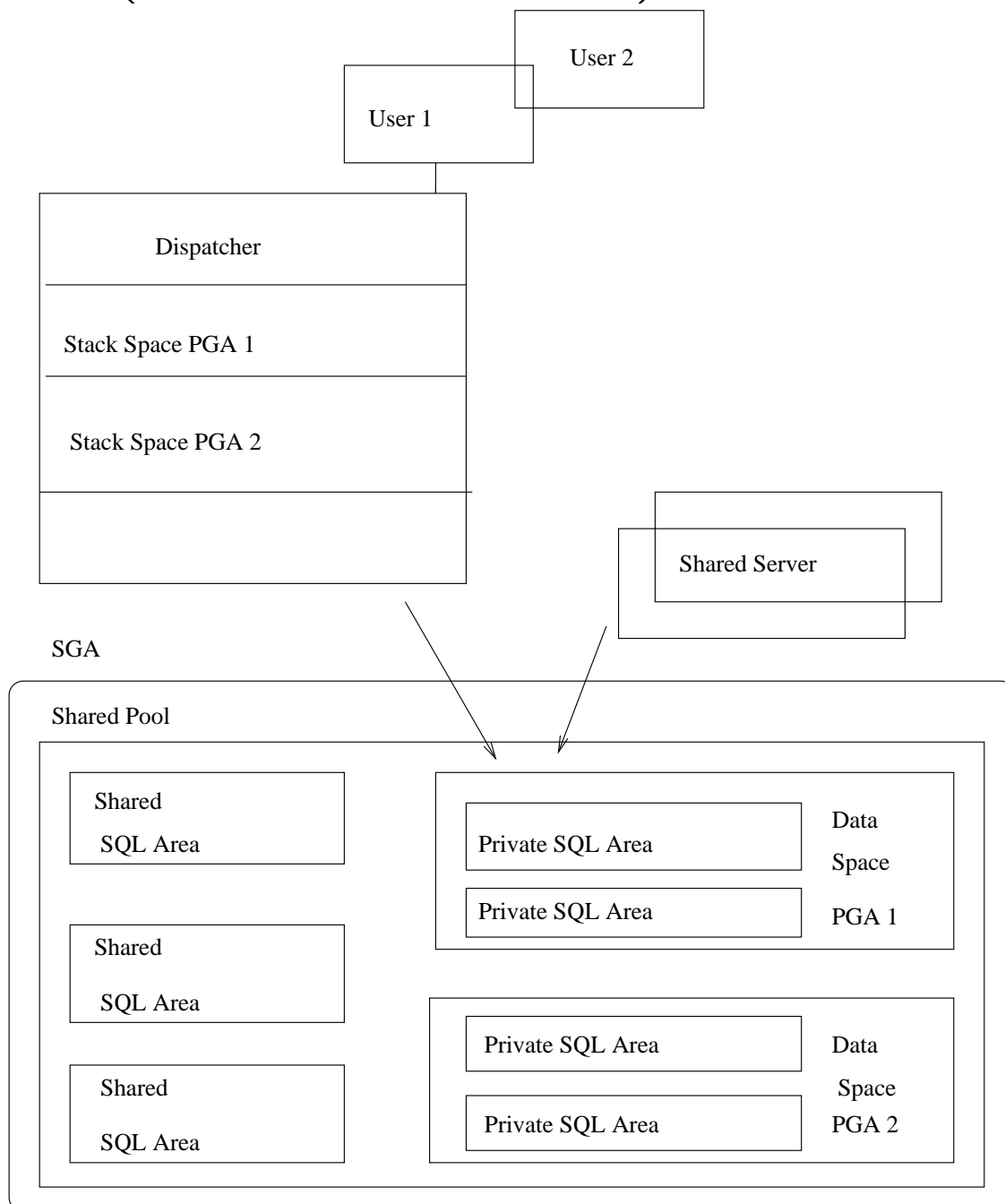
#### PGA (bei Dedicated Server)



## 1.2 Clientprozesse und DBMS (7)

### Speicherstrukturen für Clientprozesse (3)

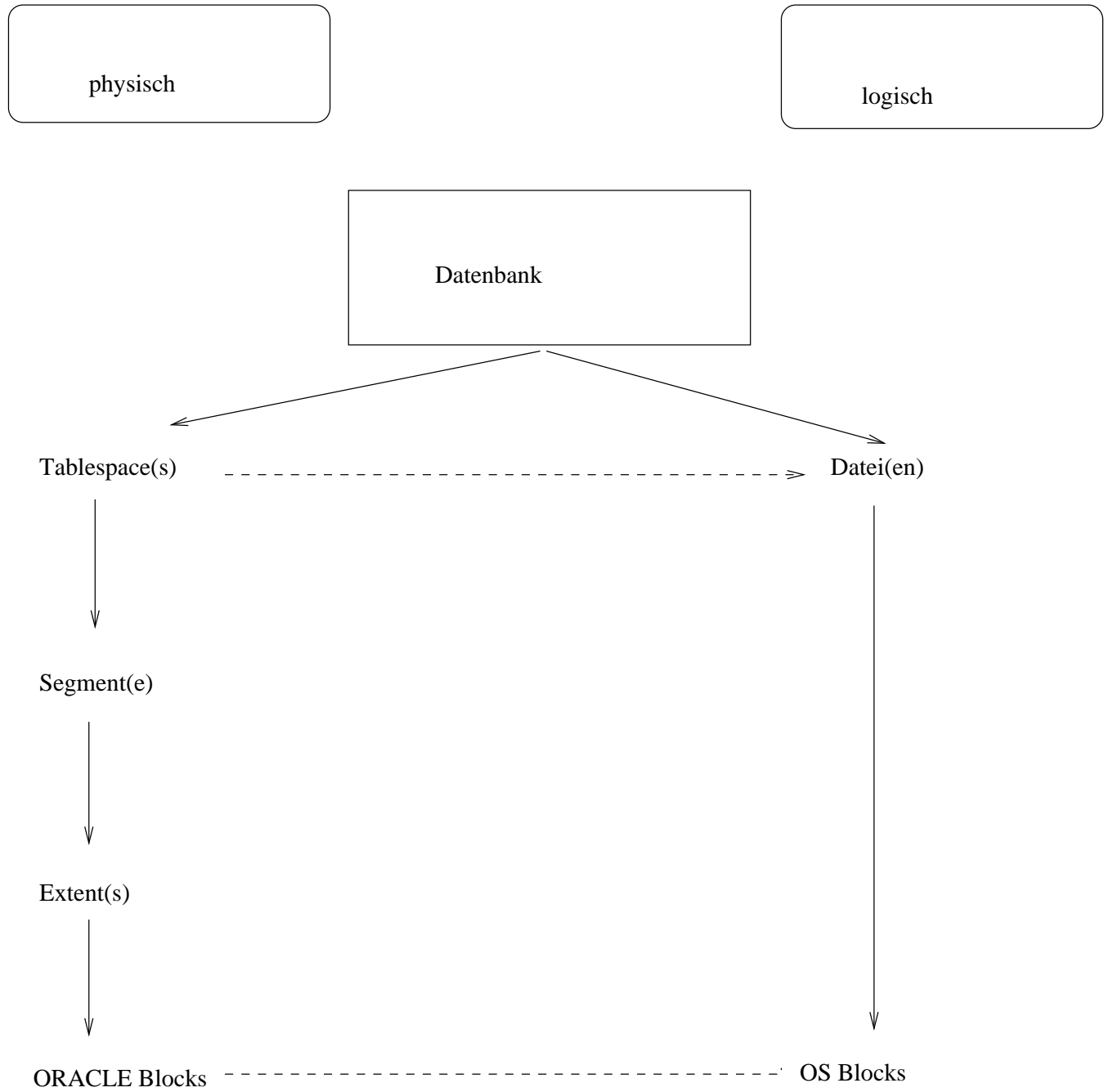
#### PGA (bei Multithreaded Server)





## 1.4 Struktur der Datenbank

### Physische und logische Datenbankstruktur



## 1.4 Struktur der DB (2)

Der Tablespace:

- mehrere verschiedene Segmente (Tabellen, Indexe, ...)
- ein oder mehrere TS pro DB (System-TS)
- ein oder mehrere Dateien pro TS (Zuordnung veränd.)
- Defaulteinstellungen für Storageparameter

Möglich ist eine Zuordnung von Oracle Schemata (User Accounts) zu Tablespaces (Erleichterung der Rechtevergabe).

## 1.4 Struktur der DB (3)

Das Segment:

- Gesamtheit aller Daten eines Objektes (z.B. Tabelle)
- ein oder mehrere Extents
- gehört zu genau einem TS, wohl aber zu mehreren Dateien

Segmenttypen:

- Daten Segmente (Tabellen)
- Index Segmente (muss nicht im gleichen TS liegen wie indizierte Tabelle)
- Rollback Segmente (Datenzwischenspeicherung, Lesekonsistenz)
- temporäre Segmente (Sortiervorgänge, Verwaltung durch System)
- Bootstrap Segment (Def. der Data Dictionary Tabellen, create Database, beim Startup in SGA)

## 1.4 Struktur der DB (4)

Das Extent:

- immer in genau einer Datei
- vollständig einem Segment zugeordnet
- kann nicht freigegeben werden, solange Segment besteht (Schweizer Käse)
- Extent Verwaltung eines Segmentes durch Storage Klauseln

Der Oracle Block:

- kleinste Oracle spezifische Einheit
- OS abhängig
- unabhängig von OS Blockgrösse (gleich ist am besten)

## 2. Die Tuningebenen eines Oracle DBMS

### 2.1 Überblick

Allgemeines:

- grösste Tuningerfolge beim Einhalten der Reihenfolge Applikation - DBMS - Betriebssystem
- Tuning ist immer iterativ !
- klare Zielvorgaben sind wichtig

Tuningebenen:

#### (1) Business Rules

- klare Funktionsbeschreibung der Applikation, keine Implementationsdetails
- OLTP oder DSS (MTS oder nicht)

#### (2) Data Design

- Def. der Daten, Beziehungen und Attribute,
- Normalisierung (Redundanzen eliminieren)
- möglich aber auch tlw. Denormalisierung wegen Performance
- Primary, Foreign Keys

#### (3) Application Design

- Implementation von Stufe (1), (Bsp. Caching oft benötigter, statischer Daten in der Appl.)

#### (4) Logische Struktur der Datenbank

- Index Finetuning

## 2.1 Tuning Überblick (2)

- (5) SQL
  - Nutzung aller passenden Möglichkeiten von SQL
- (6) Zugriffswege auf die Daten
  - Verfeinerung der Schritte 2-4
  - Tabellencluster, B-tree indexe, Bitmapmed Indexe, ...)
- (7) Speicherbelegung
  - Sizing der SGA, PGA
- (8) I/O und physische Struktur der DB
  - Verteilung der DB auf Plattendevices
  - RAID
  - OS Spiegelung oder Oracle Spiegelung
  - SAME
  - RAW Devices oder Filesysteme
  - passende Storage Parameter
- (9) Ressourcenengpässe
  - Multi-User Umgebung
  - Locks, Checkpoints, Redologgröße, Rollbacksegmente
- (10) Plattformspezifika
  - OS Buffer Cache, LVM, Kernel, Memory

## 2.1 Tuning Überblick (3)

- Vorstellung von Ansätzen zum Tuning bzgl.
  - Memory,
  - I/O,
  - CPU

aus der Sicht des DBA bzw. OS Administrators

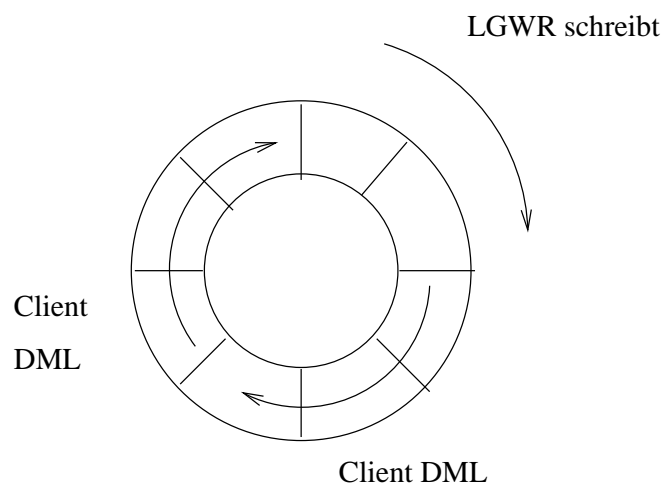
- Applikations- und Datenbankdesign werden nicht betrachtet.
- Tuningmittel:
  - dynamische Performancetabellen (V\$)
  - Oracle Enterprise Manager (OEM)
  - OS spezifische Performancemonitore

## 2.2 Memory Tuning

### Steuerung der Grösse der SGA

- Ziel: vollständig im Hauptspeicher
- init.ora in vordefinierten Versionen ist guter Startpunkt

#### (1) Tuning des Redolog Buffers



- init.ora: log\_buffer

## 2.2 Memory Tuning (2)

### (2) Tuning des Shared Pools

- init.ora: shared\_pool\_size
- V\$ROWCACHE (Data Dictionary Cache) (Misses/Hits)
- V\$LIBRARYCACHE (Cache für geparsete SQL Statements (shared) (Misses/Hits)
- V\$SGASTAT (free memory sollte immer da sein)

Tuning der Private und Shared SQL Areas:

- Applikationsentwicklung
- Identifikation unnötiger Parse Operationen und Reduzierung dieser
- SQL Trace Facility
- V\$SQLAREA, V\$STATNAME
- bei MTS auch V\$SESSTAT, da auch Session Inf. inkl. Priv. SQL Areas im Shared Pool

## 2.2 Memory Tuning (3)

### (3) Tuning des Buffer Cache

- init.ora: db\_block\_buffers
- V\$SYSSTAT (misses/hits)

### (4) Tuning von Sort Areas

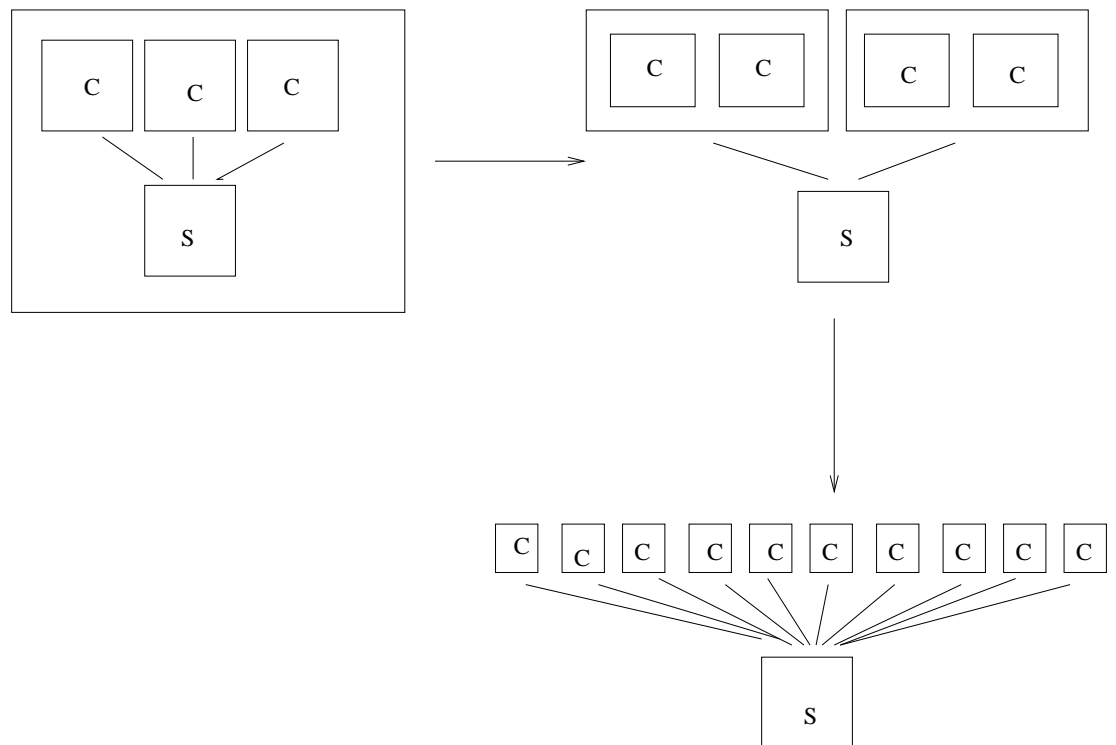
- sorts nach Möglichkeit immer im Memory
- init.ora: sort\_area\_size, sort\_area\_retained\_size (session bezogen)

## 2.3 Input/Output-Tuning und File-Layout

- Grundregel: so viel wie möglich parallelisieren (Channels, Devices)
- Datafiles, Online Redologs, Offline Redologs, Rollback-TS, Temp-TS trennen
- Tabellen und Indexe trennen
- OS Block Size  $\geq$  Database Block Size
- Row Chaining vermeiden (ANALYZE)
- SAME Prinzip (Stripe All, Mirror Everything)
- Oracle bietet Möglichkeit des Online Redolog Mirroring (prüfen!)
- Partitioned Tables , ...
- Auswertung mit OS spezifischen Performance Monitoren
- V\$FILESTAT (Datafiles)
- V\$SYSTEM\_EVENT, V\$SESSIONEVENT (Redologs, Archivelogs, Controlfiles)
- Typ der Applikation: DSS - large sequential reads, OLTP - small random R/W

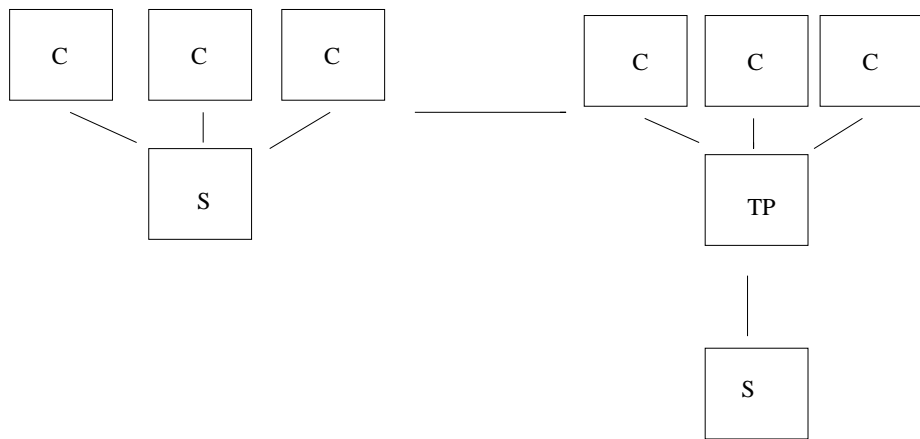
## 2.4 CPU Tuning

- Reanalyse des SQL Codes bei Performanceproblemen (95 %)
- Überdenken der Architektur (dann ist es aber meistens zu spät)
  - Single Tier zu Two Tier, kleinere Clients

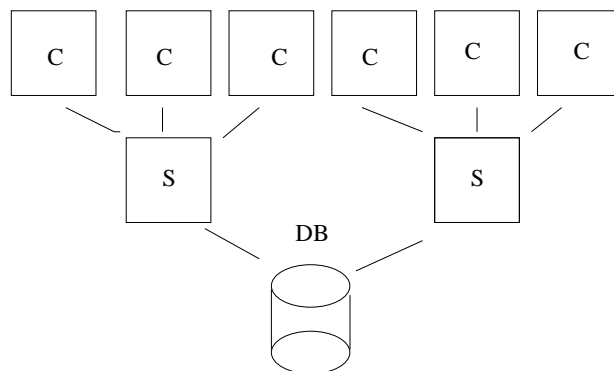


## 2.4 CPU Tuning (2)

- Two Tier zu Three Tier (TP Monitor)



- Oracle Parallel Server (?!?)



## 2.5 Oracle Spezifika

### (1) Tuning Rollback - Segmente

- Grösse und Zahl applikationsangepasst
- DSS: wenige, grosse RBS (set transaction use ...)
- OLTP: viele, kleine RBS
- dynamisches Space Management vermeiden
- ab 9i: Undo Tablespace soll vieles vereinfachen

### (2) Tuning Checkpoints

- CKPT Prozess bei vielen Datenfiles entlastet LGWR
- Reduzieren der Checkpointfrequenz durch (init.ora):
  - log\_checkpoint\_interval > grösster Redologfile
  - log\_checkpoint\_timeout = 0 (eliminiert zeitbasierte Checkpoints)

## 2.5 Oracle Spezifika (2)

### (3) Tuning Redologs

- wenn möglich mehr und grössere als weniger und kleinere (Erfahrung)

### (4) Optimizer

- 2 Optimizer: CBO (cost based), RBO (rule based)
- RBO: hat eigene Regeln, welche das SQL verwendet, um einen execution plan zu erstellen
- RBO: ab 10i desupported
- deshalb stets den CBO verwenden (init.ora)
- CBO erfordert Statistiken über alle relevanten Objekte
- `ANALYZE TABLE <table> ESTIMATE STATISTICS;`

## 2.6 OS Spezifika

### (1) Kernel

- Kernel sollte stets speziell angepasst sein
- Semaphoren
- Shared Memory (Grösse, Anzahl der Segmente)
- siehe Installation Guide
- OS Buffer Cache statisch und so klein als möglich

### (2) Filesystem- oder RAW-Device ?

#### RAW Devices

- umgehen das UNIX File Buffering (+)
- erfordern eine weitaus kompliziertere Konfigurationsplanung (—)
- bedingen ein erheblich schwereres Konfigurationstuning (—)
- DBA ist insgesamt erheblich unflexibler

## 2.6 OS Spezifika (2)

Tip: Von RAW Device basierenden DB ist abzusehen, diese sollten nur verwendet werden, wenn

- Direct I/O nicht verfügbar und
- bei extrem hohen Transaktionsraten

Allerdings: OPS benötigt RAW Devices in manchen Implementationen.

### (3) List I/O und Async I/O

- Auf manchen Unix Systemen ermöglichen derartige Kernelparameter mehrfache Schreiboperationen in parallel, jedoch meistens nur mit RAW Devices.

### (4) Prozess Scheduler

- Es ist essentiell wichtig, dass alle Oracle Prozesse die gleiche Priorität besitzen.

## Literatur

- (1) Ahmed Alomari: Oracle & Unix Performance Tuning, Prentice Hall
- (2) alle Bücher aus der O'Reilly Reihe zum Thema Oracle
- (3) Dokumentationen der Fa. Oracle