V$ Views – Don’t Leave $HOME Without Them

Mike Bell
BMC Software
Dos and Don’ts

Will Do

- Look at some V$ View Basics
- Talk about the underlying X$ Tables
- Categorize The V$ Views
- Uncover the undocumented init.ora parameters
- Present some useful scripts that are based on the V$ Views
- Learn about a wild animal

Won’t Do

- Provide a complete reference of the V$ Views
- Show an explain tree (won’t even mention it)
- Turn this into a tuning class
- Bore the experienced DBA (hopefully not)
- Traumatize the new DBA (definitely not)
Presentation Objectives

- Review V$ View Fundamentals
- V$ View Creation and Access Methods
- Categorize the V$ Views
- Provide Useful V$ View Scripts
  - Memory Allocation
  - Problem Queries
  - Media Recovery
- Show the Value of Each Script
What Are The V$ Views?

- Unfiltered, unbiased looks into the heart of the Oracle database
- Supplement to the data dictionary
- Basis for database performance monitoring and tuning
- Key to moving from the average to the expert DBA
How Are V$ Views Created?

- V$ Views are created from the X$ tables when the database is created
- The CATALOG.SQL and CATLDR.SQL scripts are executed
- A V_$ view is created from the V$ view
- Old synonyms for the V$ views are dropped
- New V$ synonyms are created on the V_$ view
V$ View Fundamentals

How’s That Again?

The V$ Views that are accessed by SYSTEM are actually synonyms that point to the V_ $ Views that are views of the original V$ Views based on the X$ Tables!
Here’s an excerpt from the CATALOG.SQL Script

```sql
create or replace view v_$database as select * from v$database;
drop public synonym v$database;
create public synonym v$database for v_$database;
grant select on v_$database to select_catalog_role;
```
V$ View Fundamentals

How Are They Accessed?

- SELECT access only
- Can’t grant access to V$ views even as SYS
- You can grant access to the underlying V_$ views

*Tip – It’s better to grant access to the V_$ Views to a specific user than to give users the SYS or SYSTEM passwords.
What Are X$ Tables?

- X$ tables are fixed tables created in memory at database startup
- Store up-to-date information on database activity
- Cannot be dropped
- Cannot be updated (except by Oracle)
- Only accessible by SYS
V$ View Fundamentals

The X$ V$ Connection

- X$ tables are very cryptic
- V$ views created on X$ tables for readability
- Several X$ tables and columns are not referenced by V$ views
- V$ views are created on one or more X$ tables
The X$ V$ Connection (cont’d)

- X$ tables and V$ views are only accessible by SYS
- 183 V$ views and 265 X$ tables in Oracle 8.1.6.0.0
- 227 V$ views and 352 X$ tables in Oracle 9.0.1.1.1
Listing the X$ Tables and V$ Views

```
select type,name from v$fixed_table order by type,name;
```

<table>
<thead>
<tr>
<th>TYPE</th>
<th>NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>TABLE</td>
<td>X$ACTIVECKPT</td>
</tr>
<tr>
<td>TABLE</td>
<td>X$BH</td>
</tr>
<tr>
<td>VIEW</td>
<td>GV$ACCESS</td>
</tr>
<tr>
<td>VIEW</td>
<td>GV$ACTIVE_INSTANCES</td>
</tr>
<tr>
<td>VIEW</td>
<td>V$ACCESS</td>
</tr>
<tr>
<td>VIEW</td>
<td>V$ACTIVE_INSTANCES</td>
</tr>
</tbody>
</table>
## V$View Fundamentals

### X$ Tables That Make up The V$ Views

```sql
select * from v$fixed_view_definition where view_name = 'GV$FIXED_TABLE';
```

<table>
<thead>
<tr>
<th>VIEW_NAME</th>
<th>VIEW_DEFINITION</th>
</tr>
</thead>
<tbody>
<tr>
<td>GV$FIXED_TABLE</td>
<td>select inst_id,kqftanam, kqftaobj, 'TABLE', indx from x$kqfta union all select inst_id,kqfvinam, kqfviobj, 'VIEW', 65537 from x$kqfvi union all select inst_id,kqfdtnam, kqfdtobj, 'TABLE', 65537 from x$kqfdt</td>
</tr>
</tbody>
</table>
Cheetah Factoid

Cheetahs Can Run a Sustained Speed of?

- 25 miles-an-hour
- 45 miles-an-hour
- 65 miles-an-hour
- 100 miles-an-hour

65 miles-an-hour
V$View Categories

- **Backups, Archiving, And Recovery**
  - V$BACKUP, V$ARCHIVE, V$RECOVER_FILE, etc.

- **Caches**
  - V$ROWCACHE, V$LIBRARYCACHE, etc.

- **Control Files**
  - V$CONTROLFILE, V$CONTROLFILE_RECORD_SECTION

- **SQL Statements and Cursors**
  - V$SQL, V$SQLAREA, V$SQLTEXT, V$OPEN_CURSOR, etc.

- **Database/Instance**
  - V$DATABASE, V$INSTANCE, V$VERSION, etc.

- **SQL*Loader Direct Path Load Option**
  - V$LOADCSTAT, V$LOADPSTAT, V$LOADSTAT
V$View Categories

- **Fixed Views**
  - V$FIXED_TABLE, V$FIXED_VIEW_DEFINITION

- **General**
  - V$TIMER, V$TYPE_SIZE, V$_SEQUENCES

- **I/O**
  - V$FILESTAT, V$WAITSTAT

- **Latches/Locks**
  - V$BUFFER_POOL, V$LATCH_MISSES, V$LOCK, etc.

- **MTS/OPS**
  - V$CIRCUIT, V$DISPATCHER, V$QUEUE, etc.

- **Overall System Performance**
  - V$SYSTAT, V$SORT_USAGE, etc.
V$View Categories

- **Parallel Query**
  - V$EXECUTION, V$PQ_SESSTAT, etc.

- **Oracle Parameters**
  - V$PARAMETER, V$NLS_PARAMETERS, etc.

- **Redo Logs**
  - V$LOG, V$LOGFILE, V$LOGHIST

- **Rollback Segments**
  - V$ROLLSTAT, V$TRANSACTION

- **Security/Privileges**
  - V$ENABLED_PRIVS, V$PWFILE_USERS

- **Sessions**
  - V$SESSION, V$PROCESS, etc.


### V$View Fundamentals

#### 282 Undocumented Parameters in 8.1.6.0.0

#### 432 Undocumented Parameters in 9.0.1.1.1

```sql
select indx, ksppinm from x$ksppi where substr(ksppinm, 1, 1) = '_';
```

<table>
<thead>
<tr>
<th>INDEX</th>
<th>KSPPINM</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>_trace_files_public</td>
</tr>
<tr>
<td>2</td>
<td>_latch_miss_stat_sid</td>
</tr>
<tr>
<td>3</td>
<td>_max_sleep_holding_latch</td>
</tr>
<tr>
<td>4</td>
<td>_max_exponential_sleep</td>
</tr>
<tr>
<td>190</td>
<td>_corrupted_rollback_segments</td>
</tr>
</tbody>
</table>
Warning

Oracle’s Warning

Warning: Information about the dynamic performance views is presented for completeness only; this information does not imply a commitment to support these views in the future.

Murphy’s Warning

Warning: All undocumented parameters should be extensively tested before and after use.
Goal #1: Memory Allocation

The Goal is to Ensure That The Correct Amount of Memory is Allocated to Oracle.
Oracle SGA

The RDBMS buffer and working storage area
Contains buffers for:

- Data Buffers
- Redo Buffers
- Data Dictionary / SQL / Procedures / Packages information

Created at instance startup
If `DB_BLOCK_BUFFERS` is low, users will not have enough memory to operate efficiently.

If `DB_BLOCK_BUFFERS` is high, your system may begin to swap and may come to a halt.
Determine if DB_BLOCK_BUFFERS Is Set Too Low:

```
select  
1 - (sum(decode(name,'physical reads',value,0)) / 
 (sum(decode(name,'db block gets',value,0)) + 
 (sum(decode(name, 'consistent gets',value,0))))) 
* 100 "Read Hit Ratio"

from   v$sysstat;

Read Hit Ratio
98.415926
```
Use X$bh to Get State of SGA

```sql
select state, count(*)
from x$bh
group by state;
```

<table>
<thead>
<tr>
<th>STATE</th>
<th>COUNT(*)</th>
<th>Status (fyi)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>920</td>
<td>free</td>
</tr>
<tr>
<td>1</td>
<td>1553</td>
<td>available</td>
</tr>
<tr>
<td>3</td>
<td>27</td>
<td>being used</td>
</tr>
</tbody>
</table>

DB_BLOCK_BUFFERS available = 1553
DB_BLOCK_BUFFERS being used = 27
DB_BLOCK_BUFFERS never used = 920
Hold On A Minute!
There’s Some Misleading Information Here!

`select state, count(*)
from x$bh
where state = 1
and lrba_seq <> 0
Group by state;

COUNT 38`

Some blocks listed at state = 1 can still be in use by an existing query!
Select
   decode(state,0, 'FREE',
      1, decode(lrba_seq,0,'AVAILABLE','BEING USED'),
      3, 'BEING USED', state) "BLOCK STATUS", count(*)
From x$bh
group by decode(state,0,'FREE',
   1,decode(lrba_seq,0, 'AVAILABLE','BEING USED'),
   3, 'BEING USED', state);

<table>
<thead>
<tr>
<th>BLOCK STATUS</th>
<th>COUNT(*)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVAILABLE</td>
<td>1515</td>
</tr>
<tr>
<td>BEING USED</td>
<td>65</td>
</tr>
<tr>
<td>FREE</td>
<td>920</td>
</tr>
</tbody>
</table>
Shared SQL Area/Shared Pool

Contains:

- Pre-parsed database procedures
- Pre-parsed database triggers
- Recently parsed SQL
- Recently parsed PL/SQL requests
The **SHARED_POOL_SIZE**: 

This is the memory allocated for the library and data dictionary cache.

If the **SHARED_POOL_SIZE** is set too low then you will not get the full advantage of your **DB_BLOCK_BUFFERS**.
Determine library cache hit ratio.

```sql
select sum(pins) Executions, sum(pinhits) "Execution Hits",
((sum(pinhits) / sum(pins)) * 100) phitrat,
sum(reloads) Misses,
((sum(pins) / (sum(pins) + sum(reloads))) * 100) hitrat
from v$librarycache;
```

<table>
<thead>
<tr>
<th>Executions</th>
<th>Execution Hits</th>
<th>PHITRAT</th>
<th>Misses</th>
<th>HITRAT</th>
</tr>
</thead>
<tbody>
<tr>
<td>12810</td>
<td>11651</td>
<td>90.952381</td>
<td>48</td>
<td>99.6266</td>
</tr>
</tbody>
</table>
Dictionary Buffers

Contain:

- Table definitions
- Column definitions
- View definitions
- Sequence definitions
- Role definitions
- Authorizations & Permissions
Data dictionary cache miss ratio:

```
select sum(gets) "Gets",
       sum(getmisses) "Misses",
       (1 - (sum(getmisses) / (sum(gets) + sum(getmisses))))*100 "HitRate"
from  v$rowcache;
```

<table>
<thead>
<tr>
<th>Gets</th>
<th>Misses</th>
<th>HitRate</th>
</tr>
</thead>
<tbody>
<tr>
<td>10233</td>
<td>508</td>
<td>95.270459</td>
</tr>
</tbody>
</table>

This would be a good Ratio and would probably not require action in this area.
Determine the Memory Left in the \texttt{SHARED\_POOL\_SIZE}:

- \texttt{col value for 999,999,999,999 heading “Shared Pool Size”}
- \texttt{col bytes for 999,999,999,999 heading “Free Bytes”}
- \texttt{select to_number(v$parameter.value) value, v$sgastat.bytes,}
  \hspace{1cm} \texttt{(v$sgastat.bytes/v$parameter.value)*100 “Percent Free”}
- \texttt{from v$sgastat, v$parameter}
- \texttt{where v$sgastat.name = ‘free memory’}
- \texttt{and v$parameter.name = ‘shared_pool_size’}
- \texttt{and v$sgastat.pool = ‘shared pool’;}

<table>
<thead>
<tr>
<th>Shared Pool Size</th>
<th>Free Bytes</th>
<th>Percent Free</th>
</tr>
</thead>
<tbody>
<tr>
<td>20,480,000</td>
<td>12,778,732</td>
<td>62.3961523</td>
</tr>
</tbody>
</table>
Determine the Memory Left in the SHARED_POOL_SIZE:

```sql
select *
from v$sgastat
where name = 'free memory'
  and pool = 'shared pool';
```

<table>
<thead>
<tr>
<th>POOL</th>
<th>NAME</th>
<th>Free Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>shared pool free memory</td>
<td></td>
<td>12,759,956</td>
</tr>
</tbody>
</table>

If there is free memory on a consistent basis, then there is no need to increase this parameter.
Memory Left - SHARED POOL:

```
select sum(ksmchsiz) Bytes, ksmchcls Status
from  x$ksmsp
group by ksmchcls;
```

<table>
<thead>
<tr>
<th>Free Bytes</th>
<th>STATUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,024,000</td>
<td>R-free</td>
</tr>
<tr>
<td>40</td>
<td>R-freea</td>
</tr>
<tr>
<td>3,176,368</td>
<td>free</td>
</tr>
<tr>
<td>6,130,576</td>
<td>freeabl</td>
</tr>
<tr>
<td>11,631,584</td>
<td>perm</td>
</tr>
<tr>
<td>1,297,996</td>
<td>recr</td>
</tr>
</tbody>
</table>
Oracle does not state what the STATUS values mean. Here is a table of possible descriptions courtesy of Rich Niemiec.

<table>
<thead>
<tr>
<th>Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>r-free</td>
<td>This is SHARED_POOL_RESERVER_SIZE</td>
</tr>
<tr>
<td>r-freea</td>
<td>This is probably reserved memory that has been used but freeable</td>
</tr>
<tr>
<td>free</td>
<td>This is the amount of contiguous free memory available</td>
</tr>
<tr>
<td>freeabl</td>
<td>This is probably memory that has been used but is freeable</td>
</tr>
<tr>
<td>perm</td>
<td>This is free memory not yet moved to the free area for use</td>
</tr>
<tr>
<td>recr</td>
<td>Not quite sure – possibly reserved memory for Oracle</td>
</tr>
</tbody>
</table>
The Oldest Fossils of The Cheetah (from about 10,000 years ago) were found where?

- North America
- Asia
- Africa
- Europe

*MIRACINONYX INEXPECTATUS*  
prehistoric cheetah-like cat of North America
Goal #2: Identify Problem Queries

The Goal is to Identify SQL Statements That Are Causing Database Performance Degradation.
Identifying Which Statements Are The Top Disk Readers (Physical)

Select `disk_reads`, `sql_text` from `v$sqlarea` Where `disk_reads` > 10000 order by `disk_reads` desc;

<table>
<thead>
<tr>
<th>DISK READS</th>
<th>SQL TEXT</th>
</tr>
</thead>
<tbody>
<tr>
<td>12,987</td>
<td>select order#, columns, types from orders where substr(orderid, 1, 2) = :1</td>
</tr>
<tr>
<td>11,131</td>
<td>select custid, city from customer where city = ‘DETROIT’</td>
</tr>
</tbody>
</table>
Identifying Which Statements Are The Top Buffer Readers (Logical)

```sql
select buffer_gets, sql_text
from v$sqlarea
Where buffer_gets > 200000
order by buffer_gets desc;
```

<table>
<thead>
<tr>
<th>Buffer_gets</th>
<th>SQL TEXT</th>
</tr>
</thead>
<tbody>
<tr>
<td>300,219</td>
<td>select order#,cust_no, from orders where division = '1'</td>
</tr>
</tbody>
</table>
1. Set Up The Report Headings

```
col PTYP format a15 heading 'PL/SQL Type'
col OBJ format a40 wrap heading 'Name'
col EXES format 999,990 heading 'Execs'
```
2. Select From The V$ Views

```sql
select PTYP, OBJ, 0 - EXEM EXES
from (select distinct EXEM, PTYP, OBJ
    from ( select O.TYPE PTYP, O.OWNER || '.' || O.NAME OBJ,
            0 - O.EXECUTIONS EXEM
    from V$DB_OBJECT_CACHE O
```
3. Filter The Results

where O.TYPE in ( 'FUNCTION', 'PACKAGE',
 'PACKAGE BODY', 'PROCEDURE', 'TRIGGER')

where ROWNUM <= &1
4. Execute The Query

@hotprocs.sql  5

<table>
<thead>
<tr>
<th>PL/SQL</th>
<th>Type</th>
<th>Name</th>
<th>Execs</th>
</tr>
</thead>
<tbody>
<tr>
<td>PACKAGE</td>
<td></td>
<td>SYS.DBMS_APPLICATION_INFO</td>
<td>267</td>
</tr>
<tr>
<td>PACKAGE</td>
<td>BODY</td>
<td>SYS.DBMS_APPLICATION_INFO</td>
<td>266</td>
</tr>
<tr>
<td>PACKAGE</td>
<td></td>
<td>SYS.DBMS_OUTPUT</td>
<td>39</td>
</tr>
<tr>
<td>PACKAGE</td>
<td>BODY</td>
<td>SYS.DBMS_OUTPUT</td>
<td>38</td>
</tr>
<tr>
<td>PACKAGE</td>
<td></td>
<td>SYS.DBMS_SPACE</td>
<td>1</td>
</tr>
</tbody>
</table>
1. Set Up The Report Headings

```
col CTYP heading 'Command Type'
col OBJ format a32 wrap heading 'Table'
col EXES format 999,990 heading 'Execs'
col GETS format 99,999,990 heading 'Buff Gets'
col ROWP format 99,999,990 heading 'Rows Proc'
```
Finding the Hot Tables (cont’d)

2. Select From The V$ Views

```sql
select CTYP, OBJ, 0 - EXEM EXES, GETS, ROWP
from (select distinct EXEM, CTYP, OBJ, GETS, ROWP
from ( select decode (S.COMMAND_TYPE
    , 2, 'Insert into ', 3, 'Select from '
    , 6, 'Update  of ', 7, 'Delete from '
    , 26, 'Lock    of ') CTYP
    , O.OWNER || '.' || O.NAME OBJ
    , sum(0 - S.EXECUTIONS) EXEM
    , sum(S.BUFFER_GETS) GETS
    , sum(S.ROWS_PROCESSED) ROWP
from V$SQL S, V$OBJECT_DEPENDENCY D
    , V$DB_OBJECT_CACHE O
).distinct
```

Finding the Hot Tables (cont’d)

3. Filter The Results

where S.COMMAND_TYPE in (2,3,6,7,26)
    and D.FROM_ADDRESS = S.ADDRESS
    and D.TO_OWNER     = O.OWNER
    and D.TO_NAME      = O.NAME
    and O.TYPE         = 'TABLE'
group by S.COMMAND_TYPE
    , O.OWNER
    , O.NAME )
where ROWNUM <= &1
/

//
### Finding the Hot Tables (cont’d)

<table>
<thead>
<tr>
<th>Command Type</th>
<th>Table</th>
<th>Execs</th>
<th>Buff Gets</th>
<th>Rows Proc</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select from</td>
<td>SYS.USER$</td>
<td>2,122</td>
<td>895,172</td>
<td>25,485</td>
</tr>
<tr>
<td>Select from</td>
<td>SYS.OBJ$</td>
<td>2,005</td>
<td>730,363</td>
<td>21,538</td>
</tr>
<tr>
<td>Insert into</td>
<td>BMC.MIG_ROWS</td>
<td>1,000</td>
<td>1,140</td>
<td>1,000</td>
</tr>
<tr>
<td>Update of</td>
<td>BMC.MIG_ROWS</td>
<td>1,000</td>
<td>53,260</td>
<td>1,000</td>
</tr>
<tr>
<td>Insert into</td>
<td>BMC.ITEM</td>
<td>880</td>
<td>1,878</td>
<td>880</td>
</tr>
<tr>
<td>Insert into</td>
<td>BMC.EMPLOYEE</td>
<td>608</td>
<td>1,312</td>
<td>608</td>
</tr>
<tr>
<td>Select from</td>
<td>SYS.CDEF$</td>
<td>584</td>
<td>1,771</td>
<td>232</td>
</tr>
<tr>
<td>Select from</td>
<td>SYS.IND$</td>
<td>558</td>
<td>180,488</td>
<td>4,756</td>
</tr>
<tr>
<td>Select from</td>
<td>SYS.JOB$</td>
<td>516</td>
<td>572</td>
<td>0</td>
</tr>
<tr>
<td>Select from</td>
<td>SYS.TAB$</td>
<td>510</td>
<td>275,021</td>
<td>8,010</td>
</tr>
</tbody>
</table>
Finding Miscellaneous Stats

Hot Stats STATS.SQL

col NAME  format a55 heading 'Statistic'
col VALUE format 999,999,990 heading 'Value'
select NAME, VALUE  from V$SYSSTAT
where NAME like '%%&1%'
order by NAME
## Finding Miscellaneous Stats (cont’d)

### STATS.SQL

```sql
@stats.sql redo
```

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>redo blocks written</td>
<td>82</td>
</tr>
<tr>
<td>redo buffer allocation retries</td>
<td>0</td>
</tr>
<tr>
<td>redo entries</td>
<td>147</td>
</tr>
<tr>
<td>redo entries linearized</td>
<td>0</td>
</tr>
<tr>
<td>redo log space requests</td>
<td>0</td>
</tr>
<tr>
<td>redo log space wait time</td>
<td>0</td>
</tr>
</tbody>
</table>
Finding Miscellaneous Stats (cont’d)

STATS.SQL

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>sorts (disk)</td>
<td>0</td>
</tr>
<tr>
<td>sorts (memory)</td>
<td>120</td>
</tr>
<tr>
<td>sorts (rows)</td>
<td>739</td>
</tr>
</tbody>
</table>
Cheetah Factoid

Of The 12,400 Cheetahs Remaining in The World, How Many Remain In The Wild?

- 1,000
- 2,400
- 5,000
- 12,400

2,400
Goal #3 - Media Recovery

Have You Ever Gotten This Error?

```
SQL> startup
ORACLE instance started.

Total System Global Area  65004812 bytes
Fixed Size             70924 bytes
Variable Size          44376064 bytes
Database Buffers      20480000 bytes
Redo Buffers           77824 bytes
Database mounted.
ORA-01113: file 11 needs media recovery
ORA-01110: data file 11: D:\ORACLE\ORADATA\B816\SFI_DEBUG_1.DBF'
```
Some people restore the database multiple times after receiving this error and still cannot resolve the problem.

What has happened is the one or more of the tablespaces were in backup mode when the database was shutdown.

Since the tablespace was in BACKUP mode, the datafile SCN number is not consistent with the last SCN of the database.
1. Query The V$BACKUP & V$DATAFILE Views to Identify The Datafiles.

```
SQL> SELECT NAME, V$BACKUP.STATUS
2  FROM V$BACKUP, V$DATAFILE
3  WHERE V$BACKUP.FILE# = V$DATAFILE.FILE#
4  and V$BACKUP.STATUS = 'ACTIVE';

NAME_________________________ STATUS
D:\ORACLE\ORADATA\B816\SFI_DEBUG_1.DBF   ACTIVE
```
2. Generate The Alter Database Statement

For Multiple Files:

```sql
select 'ALTER DATABASE DATAFILE '' || name || '' END BACKUP;'
from v$backup, v$datafile
where v$backup.file# = v$datafile.file#
and v$backup.status = 'ACTIVE';
```
Media Recovery (cont’d)

3. Execute The Alter Database Statement

```
SQL> alter database
    datafile 'D:\ORACLE\ORADATA\B816\SFI_DEBUG_1.DBF'
    end backup;
Database altered.
SQL> alter database open;
Database altered.
```
Here’s the Plug…

DBXray
- Real-time Performance Monitoring

SQL Explorer
- Collect/Tune/Test SQL Statements

Space Expert
- Identify Storage Problems/Recommend and Implement Solutions

PATROL for Oracle
- 24x7 Database Monitoring

SQL Programmer
- SQL Development

SQL Backtrack
- Efficient Backups and Guided Recoveries
Here’s the Credits

Oracle Performance Tuning
Tips & Techniques

Expert Technical Knowledge at the Turn of a Page. Tune and tailor you Oracle7, or Oracle8 system to perform at its peak using this one-of-a-kind reference, designed for every Oracle database administrator, programmer, and user.
Thanks for Coming!

About the Presenter:

Mike Bell is a Technical Product Manager for Oracle for BMC Software, a major provider of Enterprise Monitoring and Management software. Mike can be reached at michael_bell@bmc.com.

References:

Oracle Design; Dave Ensor & Ian Stevenson; O’REILLY
Oracle Performance Tuning Tips & Techniques; Richard J. Niemiec; Oracle Press
Oracle PL/SQL Tips and Techniques, Joseph C. Trezzo; Oracle Press
“Oracle Scripts Every DBA Should Know” by Joseph C. Trezzo; TUSC
“The V$ Views; A DBA’s Best Friend” by Joseph C. Trezzo; TUSC
“Performance Tuning for the Expert; Others Will Require Oxygen!” by Richard J. Niemiec; TUSC
Cheetah Conservation Fund: The WILD Foundation PO Box 1380 Ojai, CA, USA e-mail - info@cheetah.org

Neither BMC Software nor the presenter guarantee this document to be error-free. The opinions expressed in this presentation are the presenter’s and may not reflect the opinions of BMC Software or of any living person. Please provide comments and/or questions to michael_bell@bmc.com.