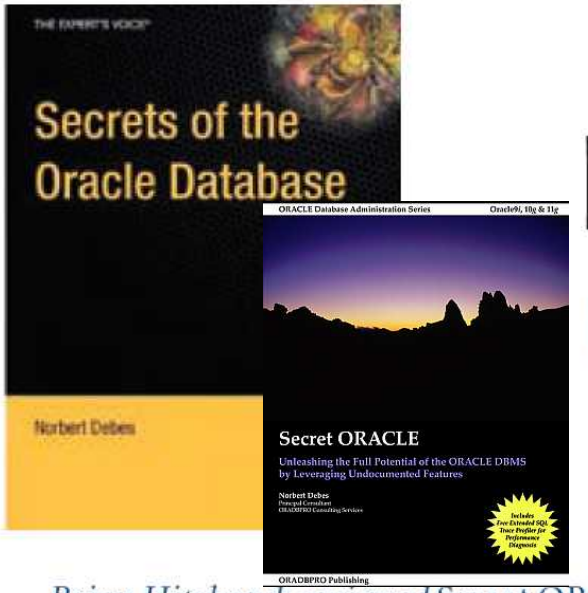
Performancediagnose und Optimierung unter Nutzung von SQL Trace

Norbert Debes

Official Publication of the Northern California Oracle Users Group
NoCOUG JOURNAL
Vol. 23, No. 2 · MAY 2009 $15

SPECIAL FEATURE



ASM Test Environment

from *Secrets of the Oracle Database*

by Norbert Debes

Norbert Debes

Brian Hitchcock reviewed Secret ORACLE—Unleashing the Full Potential of the ORACLE DBMS by Leveraging Undocumented Features *by Norbert Debes in the November 2008 issue of* NoCOUG Journal *and had this to say: "How much did I like this book? In the first 30 pages I found so many things that I didn't know but that I think are worth knowing that I can't describe them all to you in this review. That's how good this book is. I was surprised that a self-published book would have many fewer errors than a book from a major publisher. This is the first time I've read an Oracle book that was not associated in some way with a publisher. And this book is better in many ways than most I've read that came from major publishers." We sent Brian's review to Apress and they decided to publish Norbert's book. Here is an excerpt.*

Cooked files are simply the opposite of raw devices—files in a file system. After all, something that's not raw has to be cooked, right?

Setting Up Oracle Clusterware for ASM

To start an ASM instance, a stripped down version of Oracle Clusterware must be running on the same system. This is accomplished with the command `%ORACLE_HOME%\bin\localconfig add`, which creates an ORACLE cluster registry (OCR) in `%ORACLE_HOME%\cdata\localhost\local.ocr`. It also creates a new Windows service for the OCSSD Clusterware daemon. OCSSD logging goes to the file `%ORACLE_HOME%\log\<host name>\cssd\ocssd.log`.

```
C:> localconfig add
```

Themen

- Das Dateiformat des Oracle11*g* Release 2 SQL Trace
- Fehleranalyse mit SQL Trace
- Methoden zur manuellen und automatischen Aktivierung von SQL Trace
- Maschinelle Auswertung des SQL Trace mit Profilern
- Performanceoptimierung mit SQL Trace (Generierung von Testfällen, Reproduktion, Interpretation)
- Korrelation des SQL Trace mit Data Dictionary, Dynamic Performance Views, AWR, ASH und Statspack

Das Dateiformat des Oracle11*g* Release 2 SQL Trace

Messung

- extended SQL Trace
  - Datenbankoperationen (Stufe 1)
    - PARSE, EXEC, FETCH, CLOSE, XCTEND
  - Warteereignisse (Stufe 8)
    - WAIT
  - Werte von Platzhaltern (Stufen 4 und 12)
    - BIND
  - Instrumentierung
    - Service Name, Module, Action, Client Identifier, Timestamps
  - Messdaten
    - CPU-Verbrauch (c), Wartezeit (ela), Plattenzugriffe, Buffer Cache Zugriffe (STAT), Latch Misses, etc.

SQL Trace Protokolldatei

```
*** SESSION ID:(18.147) 2012-02-19 12:35:45.421
*** CLIENT ID:() 2012-02-19 12:35:45.421
*** SERVICE NAME:(ELEVEN2.oradbpro.com) 2012-02-19 12:35:45.421
*** MODULE NAME:(img_load) 2012-02-19 12:35:45.421
*** ACTION NAME:() 2012-02-19 12:35:45.421

WAIT #254793480: nam='SQL*Net message to client' ela= 8 driver id=1413697536
#bytes=1 p3=0 obj#=-1 tim=225741696755

*** 2012-02-19 12:35:45.453
WAIT #254793480: nam='SQL*Net message from client' ela= 38001 driver
id=1413697536 #bytes=1 p3=0 obj#=-1 tim=225741737256
CLOSE #254793480:c=0,e=17,dep=0,type=1,tim=225741737882
=====================
PARSING IN CURSOR #254799104 len=37 dep=0 uid=90 oct=3 lid=90 tim=225741741791
hv=3344510127 ad='26a94d5c' sqlid='g4mp88m3pkb5g'
SELECT image_id_seq.NEXTVAL FROM dual
END OF STMT
PARSE
#254799104:c=0,e=3293,p=0,cr=0,cu=0,mis=1,r=0,dep=0,og=1,plh=859217952,tim=225741
741788
WAIT #254799104: nam='SQL*Net message to client' ela= 8 driver id=1413697536
#bytes=1 p3=0 obj#=-1 tim=225741742015
WAIT #254799104: nam='SQL*Net message from client' ela= 528 driver id=1413697536
#bytes=1 p3=0 obj#=-1 tim=225741742621
EXEC
#254799104:c=0,e=604,p=0,cr=0,cu=0,mis=0,r=0,dep=0,og=1,plh=859217952,tim=2257417
43305
WAIT #254799104: nam='SQL*Net message to client' ela= 6 driver id=1413697536
#bytes=1 p3=0 obj#=-1 tim=225741743421
```

PARSE, EXEC, FETCH

- #n: Cursor Nummer
- c: CPU-Verbrauch ab 9*i* in Mikrosekunden
- e: Dauer (elapsed time) ab 9*i* in Mikrosekunden
- p: Plattenzugriffe (physical reads)
- cr: Consistent Reads
- cu: Current Reads
- mis: Cursor Misses
- r: verarbeitete Zeilen (rows)
- dep: Schachtelungstiefe (dep=0 summiert dep>0)
- og: Optimizer Goal: 1=All_Rows, 2=First_Rows, 3=Rule, 4=Choose
- plh: Plan Hash Value (11.1.0.7)
- tim: Zeitstempel ab 9*i* in Mikrosekunden

Binds und SQL Trace

```
Trace file /oradg/diag/rdbms/eleven2/ELEVEN2/trace/ELEVEN2_ora_15991.trc
PARSING IN CURSOR #9 len=463 dep=0 uid=85 oct=3 lid=0 tim=6569254549019 hv=273968775 ad='
SELECT emp.last_name, emp.first_name, j.job_title, d.department_name, l.city,
       l.state_province, l.postal_code, l.street_address, emp.email,
       emp.phone_number, emp.hire_date, emp.salary, mgr.last_name
FROM hr.employees emp, hr.employees mgr, hr.departments d, hr.locations l, hr.jobs j
WHERE l.city=:loc
AND emp.manager_id=mgr.employee_id
AND emp.department_id=d.department_id
AND d.location_id=l.location_id
AND emp.job_id=j.job_id
AND l.postal_code=:postal_code
END OF STMT
PARSE #9:c=0,e=518,p=0,cr=0,cu=0,mis=1,r=0,dep=0,og=1,plh=0,tim=6569254549018
BINDS #9:
 Bind#0
  oacdty=01 mxl=32(32) mxlc=00 mal=00 scl=00 pre=00
  oacflg=03 fl2=1000000 frm=01 csi=178 siz=56 off=0
  kxsbbbfp=fffffd7ffc510258  bln=32  avl=19  flg=05
  value="South San Francisco"
 Bind#1
  oacdty=02 mxl=22(22) mxlc=00 mal=00 scl=00 pre=00
  oacflg=03 fl2=1000000 frm=00 csi=00 siz=0 off=32
  kxsbbbfp=fffffd7ffc510278  bln=22  avl=04  flg=01
  value=99236
EXEC #9:c=0,e=7644,p=0,cr=2,cu=0,mis=1,r=0,dep=0,og=1,plh=3841908205,tim=6569254556889
```

Binds und SQL Trace

- Type Codes (Bind Parameter oacdty) stehen im SQL Reference

Oracle Built-in Data Types

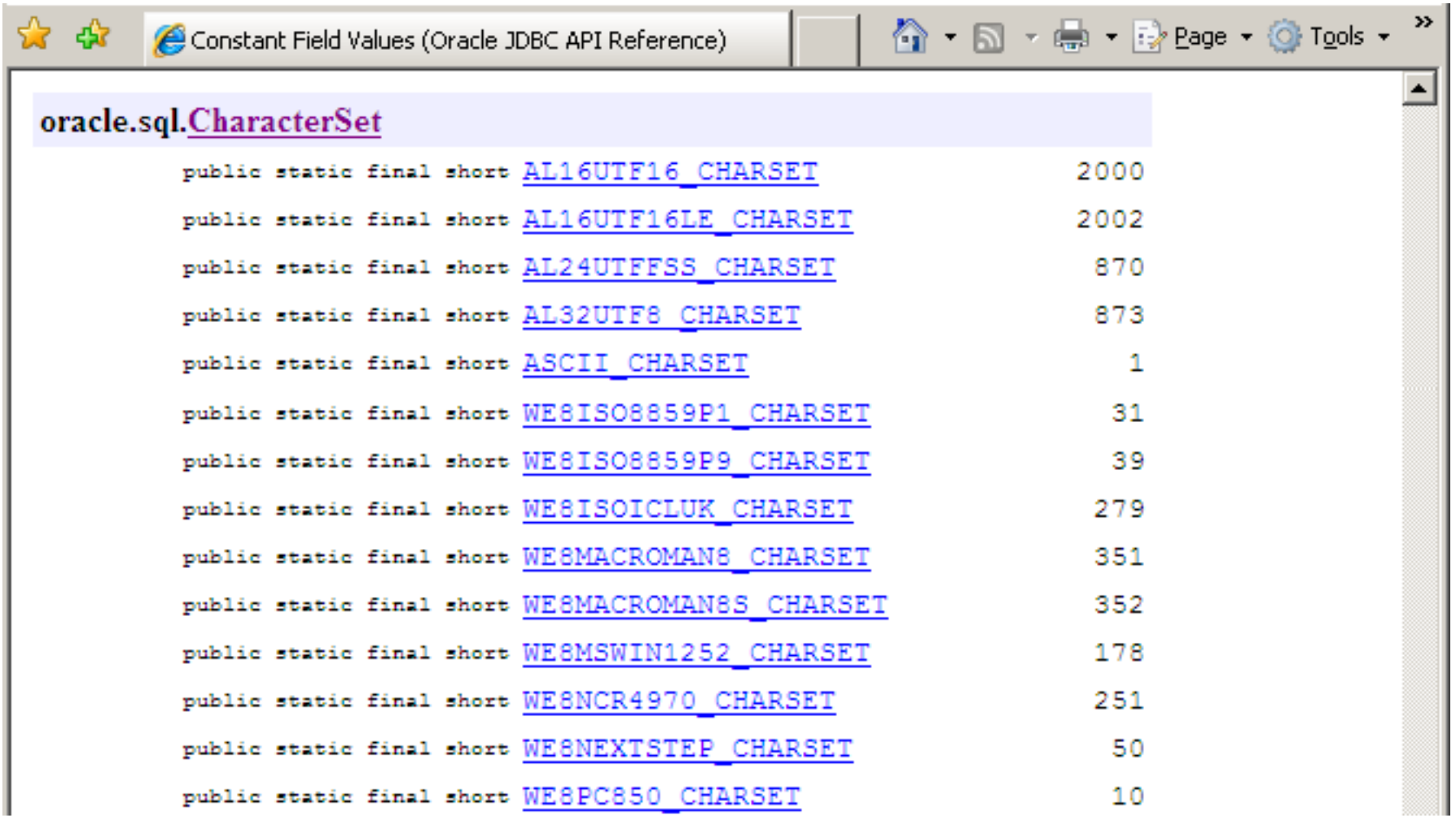
The table that follows summarizes Oracle built-in data types. Refer to the syntax in the preceding sections for the syntactic elements. The codes listed for the data types are used internally by Oracle Database. The data type code of a column or object attribute is returned by the DUMP function.

Table 3–1 Built-in Data Type Summary

Code	Data Type	Description
1	VARCHAR2(*size* [BYTE | CHAR])	Variable-length character string having maximum length *size* bytes or characters. Maximum *size* is 4000 bytes or characters, and minimum is 1 byte or 1 character. You must specify *size* for VARCHAR2. BYTE indicates that the column will have byte length semantics. CHAR indicates that the column will have character semantics.
1	NVARCHAR2(*size*)	Variable-length Unicode character string having maximum length *size* characters. The number of bytes can be up to two times *size* for AL16UTF16 encoding and three times *size* for UTF8 encoding. Maximum *size* is determined by the national character set definition, with an upper limit of 4000 bytes. You must specify *size* for NVARCHAR2.
2	NUMBER [(*p* [, *s*])]	Number having precision *p* and scale *s*. The precision *p* can range from 1 to 38. The scale *s* can range from -84 to 127. Both

csi - Character Set Identifier

- Character Set Identifier sind im Oracle JDBC API Reference im Abschnitt "Constant Field Values" dokumentiert (OTN)



Wait Event

- Warteereignis
- Response Time (R) = CPU Time + Wait Time

$$R = \sum_{dep=0} e + \sum_{inter\ db\ call} ela$$

```
WAIT #2: nam='SQL*Net message to
client' ela= 10 driver id=1413697536
#bytes=1 p3=0 obj#=-1 tim=4432543745
WAIT #2: nam='log file sync' ela= 3854
buffer#=100 p2=0 p3=0 obj#=-1
tim=4432547798
```

STAT - Ausführungspläne

- id: Reihenfolge im Plan
- pid: Parent ID
- cnt: Anzahl Zeilen (tatsächlich, nicht geschätzt)
- pos: Reihenfolge der abhägigen Operation (z.B. bei NESTED LOOPS)
- obj: Datenbankobjektnummer
- op: Row Source Operation
- cr: consistent reads
- pr: physical reads
- pw: physical writes (beim Sortieren)
- time: Ausführungszeit (kumulativ)
- cost: CBO Kosten (kumulativ)

Fehleranalyse mit SQL Trace

ERROR und PARSE ERROR

- Fehlernummer im Error Messages Guide nachschlagen oder oerr ora <Fehlernummer> verwenden

ERROR #21:err=1 tim=5355648475261

ERROR #37:err=39096 tim=5356682467527

- Parse Error:

PARSE ERROR #254074284:len=34 dep=0 uid=0 oct=85 lid=0 tim=410530328692 err=942

truncate table STATS$TEMP_SQLSTATS

Methoden zur manuellen und automatischen Aktivierung von SQL Trace

Tracing einer fremden Session

- ab Oracle10g DBMS_MONITOR verwenden
- erste dokumentierte Möglichkeit SQL Trace mit Level 1, 4, 8 und 12 einzuschalten
- leider nur für Rollen DBA und OEM_MONITOR verfügbar
- GRANT an Nicht-Administratoren problematisch
- Wrapper erforderlich, damit Applikationen für die eigene Session DBMS_MONITOR nutzen können

DBMS_MONITOR (11g Version)

```
PROCEDURE session_trace_enable(
    session_id IN BINARY_INTEGER DEFAULT NULL,
    serial_num IN BINARY_INTEGER DEFAULT NULL,
    waits IN BOOLEAN DEFAULT TRUE,
    binds IN BOOLEAN DEFAULT FALSE,
    plan_stat  IN VARCHAR2 DEFAULT NULL
);
```

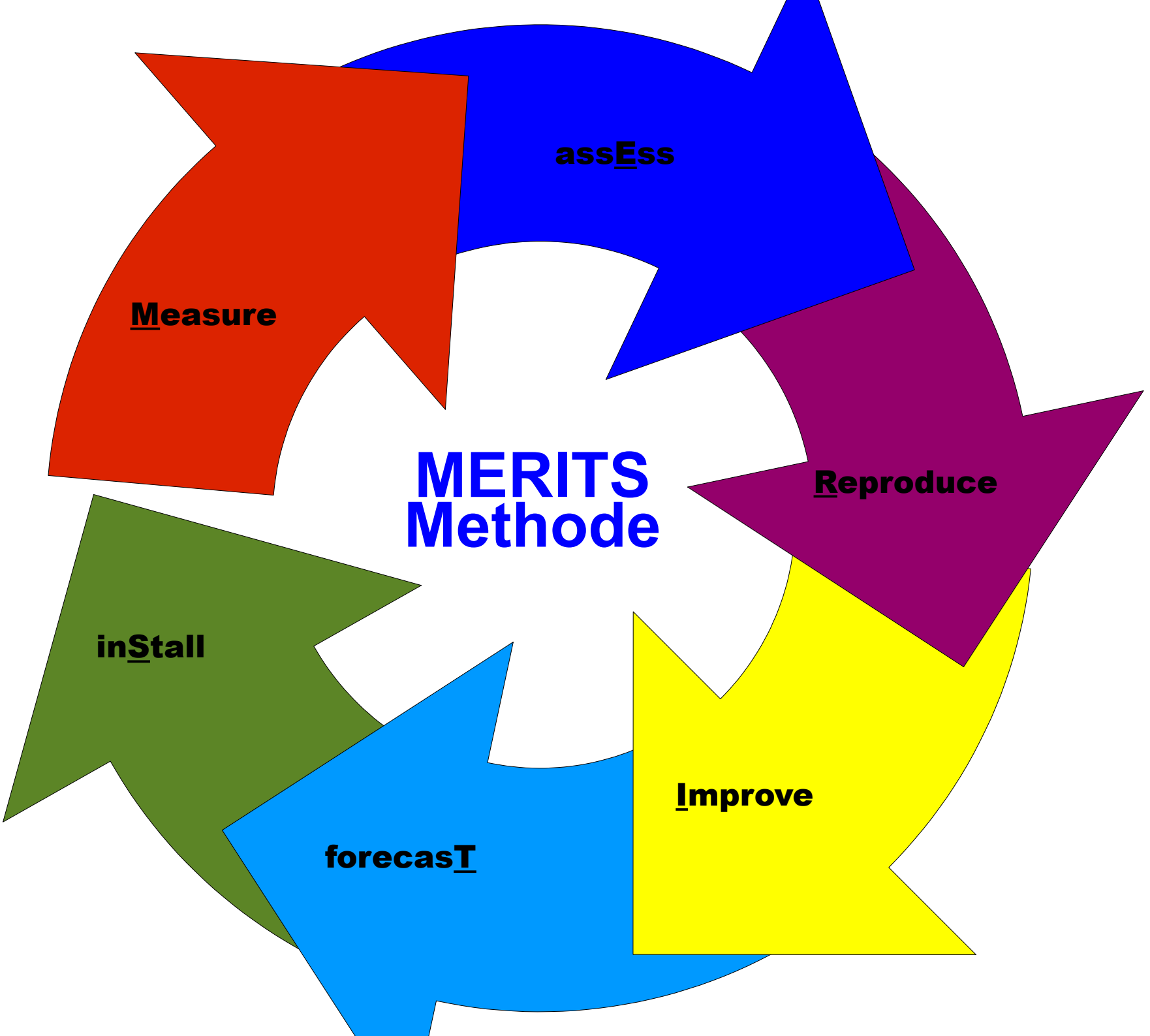
Logon Trigger

- Besitzer des Triggers benötigt Systemprivileg ALTER SESSION
  - EVENT 10046 verwenden
  - ALTER SESSION für Applikationsbenutzer nicht erforderlich
  - `ALTER SESSION SET EVENTS '10046 TRACE NAME CONTEXT FOREVER, LEVEL 12';`
- Automatisches Tracing ab Logon für
  - alle Sessions eines oder mehrerer Benutzer

Performanceoptimierung mit SQL Trace

Die MERITS Optimierungsmethode

- Methodisches Optimierungsverfahren bestehend aus sechs Phasen:
  - Phase 1: Measurement
  - Phase 2: Assessment
  - Phase 3: Reproduction
  - Phase 4: Improvement
  - Phase 5: Extrapolation
  - Phase 6: Installation


assEss
Measure
MERITS
Methode
Reproduce
inStall
Improve
forecasT

Extended SQL Trace

- Vorteile
  - enger Bezug zum Verursacher der Last (Problem: Connection Pooling ohne Instrumentierung)
  - Algorithmus erkennbar
  - feine zeitliche Auflösung
  - protokolliert Schachtelungstiefe (dep=*n*)
  - Latch/Enqueue Details
  - bessere Profiler als TKPROF von Dritten verfügbar
- Nachteile
  - Ausführungspläne (STAT) nur bei Schließen eines Cursors enthalten (besser ab 11g)
  - Instanzparameter fehlen
  - Instanzstatistiken fehlen
  - keine Session Statistiken außer PARSE, EXEC, FETCH, WAIT, XCTEND, CLOSE
  - keine Advisors
  - keine rückblickende Analyse
  - keine Messwerte auf Betriebssystemebene (V$OSSTAT)

SQL Trace - weitere Vorteile

- Zuordnung der (rekursiven) SQL-Anweisungen (PARSE, EXEC, FETCH), Wait Events, Transaktionsgrenzen (XCTEND)
- **tatsächliche** Ausführungspläne (STAT)
- **Antwortzeit berechenbar (vgl. DB Time)**
- Werte und Datentypen der Bind Variablen verfügbar (→ Reproduktion)
- Latch/Enqueue sowie Read/Write Details verfügbar
- Think Time berechenbar (→ Vorhersage/Potential)
- Statistiken pro Instance, Service, Modul, Action, Client Identifier berechenbar

Antwortzeitprofil

Antwortzeitprofil (MERITS Profiler)

Response Time Contributor	Duration (s)	Percent (%)	Count	Average (s)
think time	255.237	41.9	1050	0.243083
CPU	217.378	35.7	647142	0.000336
SQL*Net message from client	85.725	14.1	270766	0.000317
unknown	41.542	6.8	63	0.659389
log file switch completion	4.568	0.7	18	0.253801
log file switch (checkpoint incomplete)	3.879	0.6	12	0.323286
latch: shared pool	0.525	0.1	252	0.002084
SQL*Net message to client	0.467	0.1	270766	0.000002
log file sync	0.127	0.0	52	0.002450
log buffer space	0.064	0.0	2	0.031800
SQL*Net break/reset to client	0.048	0.0	272	0.000177
latch: library cache	0.045	0.0	87	0.000512
db file sequential read	0.021	0.0	5	0.004222
latch: enqueue hash chains	0.004	0.0	1	0.003633
Total	609.631	100.0	1190488	0.000512

Maschinelle Auswertung des SQL Trace mit Profilern

SQL Trace Profiler

- Programm das SQL Trace Dateien auswertet und Resource Profiles, Histogramme, Ausführungspläne darstellt.
- TKPROF genügt nicht der obigen Definition

MERITS Profiler

- Basisversion kostenlos, aber limitiert auf Tracefiles von 1 MB, Echtzeitmodus ebenfalls kostenpflichtig
- plattformunabhängig (Java)
- umfangreicher Bericht inklusive Database Call Statistiken und Wait Event Histogrammen
- ausführliche Aufbereitung der Ausführungspläne (Kosten pro Row Source)
- im Echtzeitmodus zahlreiche Korrelationen zwischen SQL Trace, V$ Views, Data Dictionary, Statspack Repository, AWR, ASH
- Think Time unterstützt und konfigurierbar (in Basisversion enthalten)
- Erkennung und Aggregation ähnlicher SQL Anweisungen bei Fehlen von Bind Variablen

Demonstration

- Laden von LOBs
- mprof sql_trace_file=eleven2_ora_6204_img_load.trc

Korrelation des SQL Trace mit Data Dictionary, Dynamic Performance Views, AWR, ASH und Statspack

Diagnostische Lücke des SQL Trace

- SQL Anweisungen können fehlen, falls die Empfehlung "***einmal parsen, oft ausführen"*** beachtet wird
- Pläne fehlen, falls
  - Cursor nicht während des Trace geschlossen wird
  - Packages verwendet werden (halten Cursor offen)
- Optimizer Environment fehlt:
  - Event 10053 (CBO Trace) zieht nur bei Hard Parse
  - SQL Profile fehlt
  - Outline fehlt
  - Outline Type (public/private) fehlt
- Tabellen-, Indexstruktur und Statistiken fehlen
- Workarea Statistiken fehlen
- Instance Parameter fehlen
- Session Level Parameter fehlen

Warum Echtzeitauswertung?

- Informationslücken des SQL Trace Dateiformats schließen (z.B. Ausführungspläne von V$SQL_PLAN)
- zusätzliche Informationen sammeln
  - Session Statistiken, Optimizer Environment
  - Tabellen-, Indexstruktur und Objektstatistiken
    - Tabellen, Indizes, Partitionen
      - wie gross sind die Segmente?
      - wie selektiv sind Spalten? Clustering Factor?
      - lohnt zusätzlicher Index?
      - Fragmentierungsgrad?
  - Systemstatistiken
  - Buffer Cache Inhalt
  - frühere Ausführungspläne abrufen (Statspack, AWR)

Statspack

- Bericht liefert zusätzliche Informationen:
  - Session Statistiken, Session Wait Events, Session Time Model
  - auch als Kontrolle zu den Meßdaten im SQL Trace
  - Parameter
  - Advisors
  - „heiße“ Segmente (logical/physical reads; level 7)
  - Instanzweite Sicht vs. sitzungsspezifische Sicht
    - z.B. Ressourcenverbrauch der teuren Statements im SQL Trace auf Instanzebene

Merkmale des MERITS Profilers

- Parser für extended SQL Trace ab Oracle10g Release 2 bis Oracle11g Version 11.2
- **Resource Profiles schlüsseln die Antwortzeit auf**
- bequemes Ein- u. Ausschalten der Protokollierung durch den Profiler selbst (SQL Trace Levels 1, 4, 8 und 12 konfigurierbar)
- Aufrufhierarchie (dep) und Antwortzeit pro Rekursionsstufe
- Think-Time (Schwellwert konfigurierbar)
- Hinweise auf Tabellen-Fragmentierung (estimated space efficiency) als Teil der Objektstatistiken
- STATISTICS_LEVEL=ALL automatisch und temporär einstellbar
- Echtzeit-Korrelation, z.B.:
  - V$SQL_PLAN
  - Buffer Cache Inhalt (Entscheidungsgrundlage für multiple Buffer Pools)
- automatische Erstellung von Statspack (Level 0-10 konfigurierbar) oder AWR Snapshots am Beginn und Ende des Messintervalls
- siehe http://www.oradbpro.com

Zusammenfassung

- neues Paradigma: **Echtzeit-Analyse des SQL Trace**
- Umfangreiche Korrelationen zwischen Trace-File, V$ Views, Dictionary Views, Statspack/AWR Daten
  - Komplettierung der Performancediagnosedaten
  - Berücksichtigung der Historie
  - erhebliche Zeitersparnis durch automatischen Abruf der Tabellen- und Indexstatistiken sowie Struktur
  - Berücksichtigung sehr vieler Facetten
    - Objektstatistiken (z.B. Clustering Factor)
    - Systemstatistiken (MBRC, CPUSPEED, etc.)
    - Optimizer Environment
    - System/Session Time Model Statistiken
    - SQL Profiles bzw. Stored Outlines
    - Inhalte der Buffer Cache Pools (Segmente im Cache)
    - Latch/Enqueue Details, physical Reads pro Segment, etc.