The Fast Lane To Database Success

aka, hints, tips and tricks from an old dude

Connor McDonald
Developer Advocate
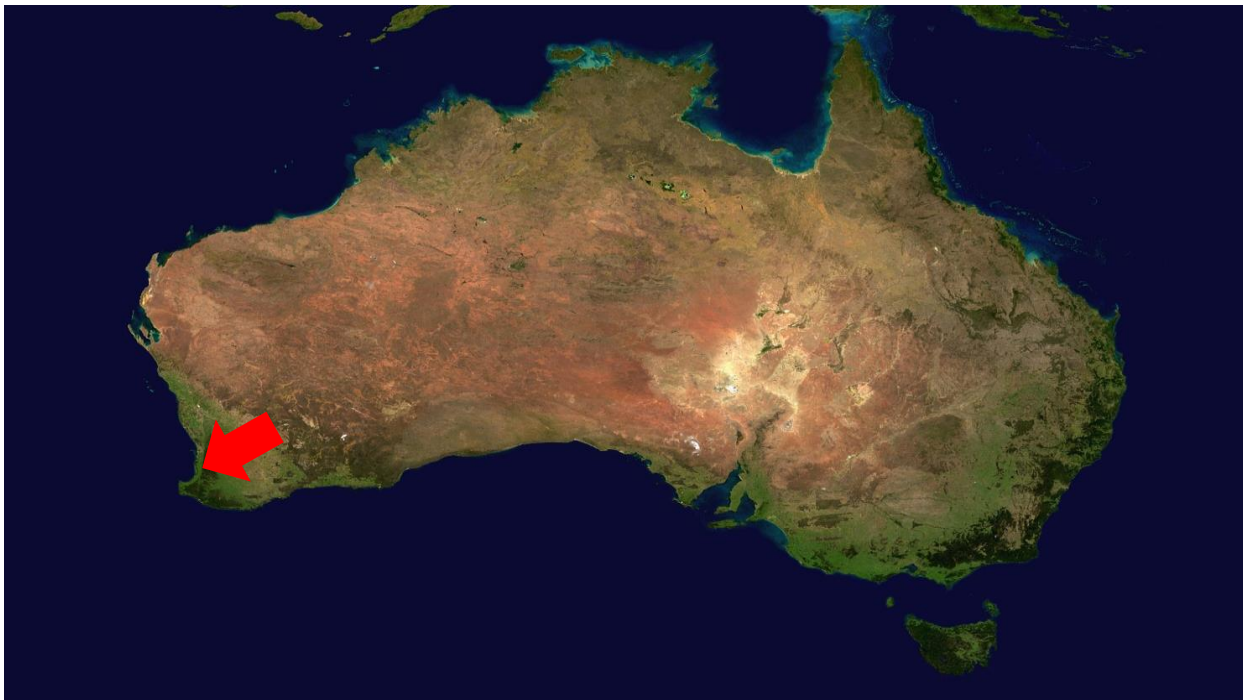
ORACLE®

Connor McDonald

ORACLE®

Stuff

blog   bit.ly/blog-connor

400+ posts mainly on database & development

youtube bit.ly/youtube-connor

250 technical videos, new uploads every week

twitter   bit.ly/twitter-connor

rants and raves on tech and the world :-)

etc…

facebook   bit.ly/facebook-connor

linkedin   bit.ly/linkedin-connor

instagram   bit.ly/instagram-connor

slideshare   bit.ly/slideshare-connor

ORACLE   Ask TOM

Questions   Office Hours   Resources   About

Feedback   Sign In

Answers for Oracle Developers
Thousands of database development questions asked and answered.

Search questions

Upcoming Office Hours

All Office Hours

Free training, how-tos and Q&A with Oracle experts, every month! Learn more

https://asktom.oracle.com

Ask TOM Office Hours
Free training, how-tos and Q&A with Oracle experts every month. Read our FAQ.

Search sessions

Office Hours Series
Click on a title to see more details. Subscribe and attend to stay in the know!

All Office Hours   Watch Past Sessions

PL/SQL 101
Next: May 1 14:00 UTC

Steven Feuerstein
Developer Advocate for PL/SQL

Sign in to Subscribe

Optimizer and Statistics

Nigel Bayliss

Sign in to Subscribe

https://asktom.oracle.com/officehours

100 hours of free access (so far)

why tips and techniques ?

SO...

HERE

IS

WHAT

HAPPENED

Thanks for the question,

Asked:

Answered by: Connor McDonald

Category: Database Development - Version: 12c

ℹ Whilst you are here, check out some content from the AskTom team: OTN tour 2016 APAC

Latest Followup

You Asked

How can I pass a parameter to a view?

and we said...

Often you will not *need* to pass a parameter, because we can "push" a predicate from a query against the view down into the view. However, for special cases we might need it. In that situation, we can

Reviews

Write a Review

How?

July 30, 2018 - 6:46 pm UTC

Reviewer: A reader

How do you just know the answer? How did you work it out ?

Connor McDonald (local) *Followup*

July 30, 2018 - 6:48 pm UTC

It's hard to explain. I just knew. Experience I guess.

July 30, 2018 - 6:48 pm UTC

Reviewer:

maybe they were right ?

I hope not :-)

it's not about manuals

it's not about re-invention

it's about sharing

it's about community

that's why

Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

if it's not in the documentation

- contact Support
- clarify usage before proceeding

1

rogue session

```
SQL> select sid, last_call_et, status,
  3  from    v$session s;

       SID LAST_CALL_ET STATUS
---------- ------------ ----------
        39         7376 ACTIVE
        40          412 INACTIVE
        44          421 INACTIVE



        ...
        80          453 INACTIVE
```

old days

idiot in database

---

much more modern approach...
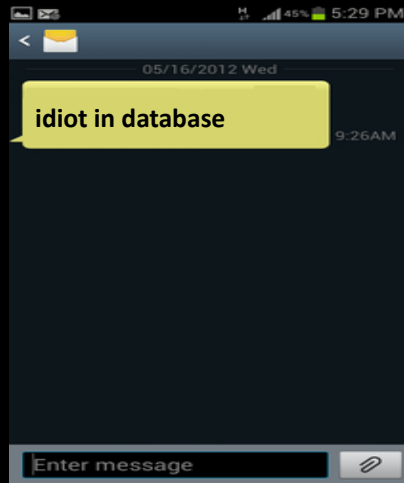
```
SQL> alter system kill session ...
```

kill -9 pid

```
SQL> oradebug setorapid nnn
SQL> oradebug suspend

       SID LAST_CALL_ET STATUS
---------- ------------ ----------
        39         7376 ACTIVE
        40          412 INACTIVE
        41          412 INACTIVE
        44          421 INACTIVE
        46           12 ACTIVE
        51            9 ACTIVE
        53           15 ACTIVE
        58            8 ACTIVE

SQL> oradebug resume
```

2

maybe ...

... you don't want to resume

query

rogue ses~~si~~on

```
SQL> alter system kill session ...
```

18c
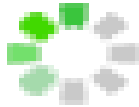
```
SQL> alter system cancel sql '123,456';
```

3

/*+ GATHER_PLAN_STATISTICS */

for the optimizer

cardinality is everything

```
SQL> explain plan for
  2  select count(*)
  3  from    VEHICLE
  4  where   MAKE = 'HONDA' and MODEL = 'CIVIC';

SQL> SELECT * FROM TABLE(DBMS_XPLAN.DISPLAY_PLAN);

-------------------------------------------------------
| Id  | Operation            | Name     |  Rows |  Bytes |
-------------------------------------------------------
|   1 |  SORT AGGREGATE      |          |     1 |      1 |
|*  2 |    TABLE ACCESS FULL | VEHICLE  |   20K |   270K |
-------------------------------------------------------
```

?

actual versus estimate

```
SQL> select /*+ GATHER_PLAN_STATISTICS */ count(*)
  2  from    VEHICLE
  3  where   MAKE = 'HONDA' and MODEL = 'CIVIC';

  COUNT(*)
----------
    114468

SQL> SELECT * FROM TABLE(DBMS_XPLAN.DISPLAY_CURSOR(
  2               NULL, NULL, 'ALLSTATS LAST'));
```

Id	Operation	Name	Starts	E-Rows	A-Rows
1	SORT AGGREGATE		1	1	1
* 2	TABLE ACCESS FULL	VEHICLE	1	20K	114K

4

tracing

alter session set sql_trace = true

ORACLE

```
C:\oracle\diag\rdbms\db122\db122\trace>dir *.trc
 Volume in drive C is OS
 Volume Serial Number is 9CB0-0212

 Directory of C:\oracle\diag\rdbms\db122\db122\trace

27/09/2018  09:18 PM             1,314 db122_ora_10020.trc
20/10/2018  06:21 PM             2,691 db122_ora_10048.trc
30/09/2018  12:21 AM             1,345 db122_ora_10104.trc
29/09/2018  03:18 PM             1,314 db122_ora_10144.trc
21/10/2018  06:21 AM             1,346 db122_ora_10156.trc
11/10/2018  05:18 PM             1,312 db122_ora_1016.trc
02/10/2018  08:49 PM             1,346 db122_ora_10192.trc
29/09/2018  12:21 PM             1,345 db122_ora_10320.trc
12/10/2018  12:21 AM             1,501 db122_ora_10348.trc
15/10/2018  12:18 AM             1,314 db122_ora_10352.trc
13/10/2018  11:21 AM             1,499 db122_ora_1036.trc
28/09/2018  05:18 AM             1,315 db122_ora_10368.trc
18/10/2018  05:18 PM             1,824 db122_ora_10384.trc
05/10/2018  05:18 AM             1,178 db122_ora_10388.trc
28/09/2018  11:21 PM             1,346 db122_ora_10400.trc
14/10/2018  11:21 PM             1,501 db122_ora_10404.trc
30/09/2018  06:25 AM             1,346 db122_ora_10440.trc
```

ORACLE Co

too many trace files

```
SQL> alter session
  2      set tracefile_identifier = SALESPGM;

Session altered.

SQL> host ls *.trc

db10r2_ora_3248.trc
db10r2_ora_3284.trc
db10r2_ora_3356.trc
db10r2_ora_3492.trc
db10r2_ora_3504_SALESPGM.trc
db10r2_ora_3608.trc
db10r2_ora_3620.trc
...
...
```

5

too much trace data

```
SQL> alter session
  2      set events
  3    'immediate trace name trace_buffer_on level 1048576';

Session altered.

SQL> host ls -l *.trc

drw------- oracle dba Jun17 15:10 1048576 db10r2_ora_3248.trc
```

rolling trace window

---

bonus tip

```
SQL> select name, value from v$diag_info;

NAME                      VALUE
----------------------    ------------------------------------------
Diag Enabled              TRUE
ADR Base                  C:\ORACLE
ADR Home                  C:\ORACLE\diag\rdbms\db122\db122
Diag Trace                C:\ORACLE\diag\rdbms\db122\db122\trace
Diag Alert                C:\ORACLE\diag\rdbms\db122\db122\alert
Diag Incident             C:\ORACLE\diag\rdbms\db122\db122\incident
Diag Cdump                c:\oracle\diag\rdbms\db122\db122\cdump
Health Monitor            C:\ORACLE\diag\rdbms\db122\db122\hm
Default Trace File        C:\ORACLE\...\trace\db122_ora_17296.trc
```

```
SQL> select PAYLOAD
  2  from    V$DIAG_TRACE_FILE_CONTENTS
  3  where   TRACE_FILENAME = 'db122_ora_12484.trc';

PAYLOAD
----------------------------------------------------------------------------
Trace file C:\ORACLE\diag\rdbms\db122\db122\trace\db122_ora_12484.trc
Oracle Database 12c Enterprise Edition Release 12.2.0.1.0 - 64bit Production
Build label:    RDBMS_12.2.0.1.0WINDBBP_WINDOWS.X64_180202
Windows NT Version V6.2
ORACLE_HOME = c:\oracle\product\12.2.0.1
Node name          : XPS13
CPU                : 4 - type 8664, 2 Physical Cores
Process Affinity   : 0x0x0000000000000000
Memory (Avail/Total): Ph:9411M/16235M, Ph+PgF:4292M/20985M
Instance name: db122
Redo thread mounted by this instance: 1
...
...
```

6

19c

automatic indexing

Automatic Indexing Methodology

Cycle diagram: Capture → Identify → Verify → Decide → Online Validation → Monitor → (back to Capture)
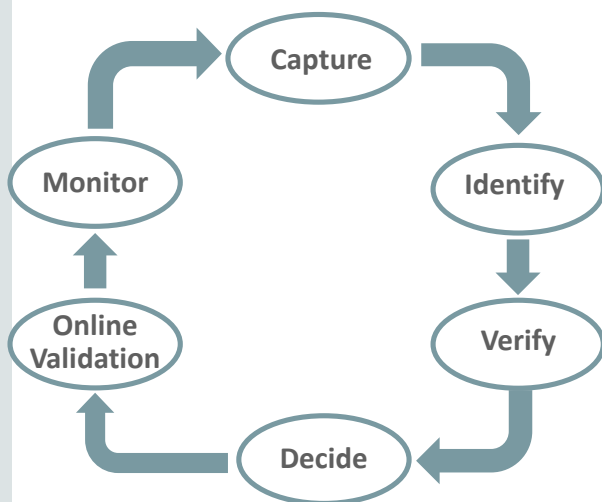
- The Automatic Indexing methodology is based on a common approach to manual SQL tuning
- It identifies candidate indexes and validates them before implementing
- The entire process is full automatic
- Transparency is equally important as sophisticated automation
  - All tuning activities are auditable via reporting

until then …

"I need a new index on this 6TB table…"

"will it speed up things ?"

"I *think* so"

virtual indexes

```
SQL> create index TEST on
  2  MYTAB ( SALES_DATE ) NOSEGMENT;

Index created.
```

data dictionary only

```
SQL> begin
  2      dbms_stats.generate_stats(
  3              'SCOTT',
  4              'MYTAB');
  5  end;
  6  /
```

```
SQL> alter session
  2      set "_use_nosegment_indexes" = true;

SQL> explain plan for
  2  select * from MYTAB
  3  where sales_date > sysdate-7;
```

```
---------------------------------------------------------------------------
| Id  | Operation                    | Name  | Rows  | Bytes | Cost (%CPU)|
---------------------------------------------------------------------------
|   0 | SELECT STATEMENT             |       |   950 | 90109 |   962  (34)|
|   1 |  TABLE ACCESS BY INDEX ROWID| MYTAB |   950 | 90109 |   962  (34)|
|*  2 |   INDEX RANGE SCAN           | TEST  |   950 |       |   962  (50)|
---------------------------------------------------------------------------
```

7

invisible indexes

```
SQL> create table T
  2  as select * from all_objects;

Table created.

SQL> create index OBJ_IX on T ( OBJECT_ID );

Index created.

SQL> exec dbms_stats.gather_table_stats('','T')

PL/SQL procedure successfully completed.
```

```
SQL> set autotrace traceonly explain

SQL> select * from T
  2  where  OWNER = 'SCOTT'
  3  and    created > sysdate - 1
  4  /

-----------------------------------------------------------------
| Id  | Operation         | Name | Rows  | Bytes | Cost (%CPU)|
-----------------------------------------------------------------
|   0 | SELECT STATEMENT  |      |    6  |   546 |   281    (1)|
|*  1 |  TABLE ACCESS FULL| T    |    6  |   546 |   281    (1)|
-----------------------------------------------------------------
```

"*Can I index OWNER..*"

"will it speed up things ?"

"I know so"

```
SQL> create index NEW_IX on T ( OWNER );

Index created.

SQL> select * from T
  2   where   OWNER = 'SCOTT'
  3   and     created > sysdate - 1
  4   /
```

```
-----------------------------------------------------------------------
| Id  | Operation                    | Name   | Rows  | Bytes | Cost (%CPU)|
-----------------------------------------------------------------------
|   0 | SELECT STATEMENT             |        |     6 |   546 |   122   (0)|
|*  1 |  TABLE ACCESS BY INDEX ROWID | T      |     6 |   546 |   122   (0)|
|*  2 |   INDEX RANGE SCAN           | NEW_IX |  4107 |       |    10   (0)|
-----------------------------------------------------------------------
```

before the new index

```
SQL> set autotrace traceonly explain

SQL> select * from T
  2  where  OWNER = 'SYS'
  3  and    OBJECT_ID between 10 and 8000
  4  /


-----------------------------------------------------------------------
| Id  | Operation                    | Name   | Rows  | Bytes | Cost (%CPU)|
-----------------------------------------------------------------------
|   0 | SELECT STATEMENT             |        |   462 | 42042 |   143   (0)|
|*  1 |  TABLE ACCESS BY INDEX ROWID | T      |   462 | 42042 |   143   (0)|
|*  2 |   INDEX RANGE SCAN           | OBJ_IX |  7851 |       |    20   (0)|
-----------------------------------------------------------------------
```

```
SQL> set autotrace traceonly stat
SQL> select * from T
  2  where   owner = 'SYS'
  3  and     object_id between 10 and 8000;

4967 rows selected.

Statistics
----------------------------------------------------------
          0  recursive calls
          0  db block gets
        784  consistent gets
         15  physical reads
          0  redo size
```

after the new index

```
SQL> select * from T
  2  where   owner = 'SYS'
  3  and     object_id between 10 and 8000
  4  /
```

the new index

```
---------------------------------------------------------------------------
| Id  | Operation                   | Name   | Rows  | Bytes | Cost (%CPU)|
---------------------------------------------------------------------------
|   0 | SELECT STATEMENT            |        |   462 | 42042 |   122   (0)|
|*  1 |   TABLE ACCESS BY INDEX ROWID| T     |   462 | 42042 |   122   (0)|
|*  2 |    INDEX RANGE SCAN         | NEW_IX | 4105 |       |    10   (0)|
---------------------------------------------------------------------------
```

```
SQL> set autotrace traceonly stat
SQL> select * from T
  2  where   owner = 'SYS'
  3  and     object_id between 10 and 8000;

4967 rows selected.

Statistics
-----------------------------------------------------------
          0   recursive calls
          0   db block gets
       1522   consistent gets
         62   physical reads
          0   redo size
```

the solution ?

invisible indexes

```
SQL> alter index NEW_IX invisible;

Index altered.
```

```
SQL> select * from T
  2  where  owner = 'SYS'
  3  and    object_id between 10 and 8000
  4  /

-----------------------------------------------------------------------------
| Id  | Operation                    | Name   | Rows  | Bytes | Cost (%CPU)|
-----------------------------------------------------------------------------
|   0 | SELECT STATEMENT             |        |   462 | 42042 |   143   (0)|
|*  1 |   TABLE ACCESS BY INDEX ROWID| T      |   462 | 42042 |   143   (0)|
|*  2 |    INDEX RANGE SCAN          | OBJ_IX | 7851 |       |    20   (0)|
-----------------------------------------------------------------------------
```

"gee... thanks for nothing"

```
SQL> set autotrace traceonly explain

SQL> select * from T
  2  where   OWNER = 'SCOTT'
  3  and     created > sysdate - 1
  4  /

-----------------------------------------------------------------
| Id  | Operation          | Name | Rows  | Bytes | Cost (%CPU)|
-----------------------------------------------------------------
|   0 | SELECT STATEMENT   |      |     6 |   546 |   281    (1)|
|*  1 |  TABLE ACCESS FULL | T    |     6 |   546 |   281    (1)|
-----------------------------------------------------------------
```

```
SQL> alter session set
  2          optimizer_use_invisible_indexes = true;

Session altered.

SQL> select * from T
  2  where   OWNER = 'SCOTT'
  3  and     created > sysdate - 1
  4  /

---------------------------------------------------------------------------
| Id  | Operation                    | Name   | Rows | Bytes | Cost (%CPU)|
---------------------------------------------------------------------------
|   0 | SELECT STATEMENT             |        |  462 | 42042 |   122    (0)|
|*  1 |  TABLE ACCESS BY INDEX ROWID | T      |  462 | 42042 |   122    (0)|
|*  2 |   INDEX RANGE SCAN           | NEW_IX | 4105 |       |    10    (0)|
---------------------------------------------------------------------------
```

wrap your SQL to protect others

8

better data guard use...

...for free

```
SQL> ALTER DATABASE CONVERT TO SNAPSHOT STANDBY;
```

open standby read/write

```
SQL> ALTER DATABASE CONVERT TO PHYSICAL STANDBY;

SQL> ALTER DATABASE RECOVER MANAGED STANDBY
  2      DATABASE DISCONNECT;
```

flashback to resetlogs

resume managed recovery

archives still transmitted

9

faster queries ...

```
SQL> create table cust_trans
  2    ( txn_id     int,
  3      txn_date   date,
  4      cust_id    int,
  5      amt        number(10,2)
  6    );
```

```
SQL> insert /*+ APPEND */ into cust_trans
  2  select rownum,
  3          sysdate-1000+rownum/1000  txn_date,
  4          trunc(dbms_random.value(1,1000)) cust_id,
  5          dbms_random.value(1,100) amount
  6  from
  7  ( select 1 from dual
  8    connect by level <= 10000000 );

10000000 rows created.

SQL> create index cust_trans_ix
  2     on cust_trans (cust_id );

Index created.
```

```
SQL> select max(amt)
  2  from    cust_trans
  3  where   cust_id = 123;
```

```
SQL> select max(amt)
  2  from    cust_trans
  3  where   cust_id = 123;


------------------------------------------------------------------------------
| Id  | Operation                           | Name          | Rows  | Bytes |
------------------------------------------------------------------------------
|   0 | SELECT STATEMENT                    |               |     1 |     8 |
|   1 |  SORT AGGREGATE                     |               |     1 |     8 |
|   2 |   TABLE ACCESS BY INDEX ROWID BATCHED| CUST_TRANS   | 10010 | 80080 |
|*  3 |    INDEX RANGE SCAN                 | CUST_TRANS_IX | 10010 |       |
------------------------------------------------------------------------------

Statistics
----------------------------------------------------------
          0  recursive calls
          0  db block gets
       8867  consistent gets
       8854  physical reads
          0  redo size
        543  bytes sent via SQL*Net to client
        607  bytes received via SQL*Net from client
```
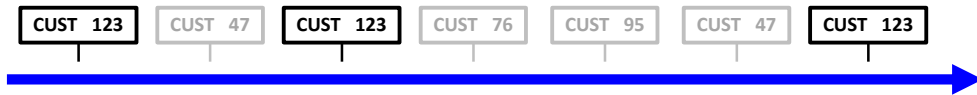
CUST 123   CUST 47   CUST 123   CUST 76   CUST 95   CUST 47   CUST 123

```
SQL> alter table cust_trans
  2       add clustering by linear order(cust_id);
```
                                                                    12.1

```
Table altered.


SQL> alter table cust_trans move online;

Table altered.
```
                                                                    12.2

```
SQL> select max(amt)
  2  from    cust_trans
  3  where   cust_id = 123;


--------------------------------------------------------------------------------
| Id  | Operation                             | Name          | Rows  | Bytes  |
--------------------------------------------------------------------------------
|   0 | SELECT STATEMENT                      |               |     1 |      8 |
|   1 |  SORT AGGREGATE                       |               |     1 |      8 |
|   2 |   TABLE ACCESS BY INDEX ROWID BATCHED | CUST_TRANS    | 10010 | 80080 |
|*  3 |    INDEX RANGE SCAN                   | CUST_TRANS_IX | 10010 |        |
--------------------------------------------------------------------------------


Statistics
-------------------------------------------------------
          0  recursive calls
          0  db block gets
         62  consistent gets
         23  physical reads
          0  redo size
        543  bytes sent via SQL*Net to client
        607  bytes received via SQL*Net from client
```

10

DBMS_XPLAN

```
SQL> explain plan for ...

SQL> select * from
  2    table(dbms_xplan.display_plan)
```

extensions

... the (in)famous hint ignore issue

```
SQL> select *
  2  from   emp e,
  3         dept d
  4  where e.deptno = d.deptno
  5  and    d.dname = 'SALES';


--------------------------------------------------------------
| Id  | Operation                   | Name    | Rows  |
--------------------------------------------------------------
|   0 | SELECT STATEMENT            |         |     5 |
|   1 |  MERGE JOIN                 |         |     5 |
|*  2 |   TABLE ACCESS BY INDEX ROWID| DEPT    |     1 |
|   3 |    INDEX FULL SCAN          | PK_DEPT |     4 |
|*  4 |   SORT JOIN                 |         |    14 |
|   5 |    TABLE ACCESS FULL        | EMP     |    14 |
--------------------------------------------------------------
```

"I want a hash join"

```
SQL> select /*+ use_hash(d) */ *
  2  from   emp e,
  3         dept d
  4  where e.deptno = d.deptno
  5  and    d.dname = 'SALES';


-------------------------------------------------------------
| Id  | Operation                    | Name     | Rows  |
-------------------------------------------------------------
|   0 | SELECT STATEMENT             |          |    5  |
|   1 |  MERGE JOIN                  |          |    5  |
|*  2 |   TABLE ACCESS BY INDEX ROWID| DEPT     |    1  |
|   3 |    INDEX FULL SCAN           | PK_DEPT  |    4  |
|*  4 |   SORT JOIN                  |          |   14  |
|   5 |    TABLE ACCESS FULL         | EMP      |   14  |
-------------------------------------------------------------
```

if joining into DEPT...

```
/*+ use_hash(d) */
```

use a hash join

we're not...

```
--------------------------------------------------------------
| Id  | Operation                     | Name     | Rows  |
--------------------------------------------------------------
|   0 | SELECT STATEMENT              |          |     5 |
|   1 |  MERGE JOIN                   |          |     5 |
|*  2 |   TABLE ACCESS BY INDEX ROWID| DEPT     |     1 |
|   3 |    INDEX FULL SCAN            | PK_DEPT  |     4 |
|*  4 |   SORT JOIN                   |          |    14 |
|   5 |    TABLE ACCESS FULL          | EMP      |    14 |
--------------------------------------------------------------
```

if we want to join into DEPT...

...we must be starting with EMP

```
SQL> select /*+ leading(e) use_hash(d) */ *
  2  from   emp e,
  3         dept d
  4  where e.deptno = d.deptno
  5  and   d.dname = 'SALES';


---------------------------------------------------------
| Id  | Operation            | Name | Rows  | Bytes |
---------------------------------------------------------
|   0 | SELECT STATEMENT     |      |     5 |   285 |
|*  1 |  HASH JOIN           |      |     5 |   285 |
|   2 |   TABLE ACCESS FULL| EMP  |    14 |   518 |
|*  3 |   TABLE ACCESS FULL| DEPT |     1 |    20 |
---------------------------------------------------------
```

not enough hints

*"hints are like violence…
if they do not work, use more"*

back to dbms_xplan

```
SQL> SELECT *
  2  from table(dbms_xplan.display(.... ,
                    format=>'typical +OUTLINE))

/*+ BEGIN_OUTLINE_DATA
    USE_HASH(@"SEL$1" "B"@"SEL$1")
    USE_HASH(@"SEL$1" "D"@"SEL$1")
    LEADING(@"SEL$1" "E"@"SEL$1" "D"@"SEL$1" "B"@"SEL$1")
    FULL(@"SEL$1" "B"@"SEL$1")
    FULL(@"SEL$1" "D"@"SEL$1")
    FULL(@"SEL$1" "E"@"SEL$1")
    OUTLINE_LEAF(@"SEL$1")
    ALL_ROWS
    DB_VERSION('11.2.0.2')
    OPTIMIZER_FEATURES_ENABLE('11.2.0.2')
    END_OUTLINE_DATA
*/
```

consider SQL Plan Management

bonus tip !

```
SQL> select /*+ INDEX(accounts_pk) */
  2  from    ACCOUNTS
  3  where   ...

SQL> select /*+ INDEX(a) */
  2  from    ACCOUNTS
  3  where   ...

SQL> select /*+ INDEX(scott.accounts) */
  2  from    SCOTT.ACCOUNTS
  3  where   ...

SQL> select /* INDEX(A) */
  2  from    SCOTT.ACCOUNTS A
  3  where   ...
```

19c

```
SQL> SELECT *
  2  from table(dbms_xplan.display(.... ,
                   format=>'typical +HINT_REPORT'))
```

11

finding bad SQL ...

... is probably bad SQL

```
SQL> select sql_fulltext
  2  from    v$sql
  3  where   buffer_gets > 1000000 or
  4          executions > 10000 or
  5          disk_reads > 100000;

SQL_FULLTEXT
-------------------------------------------------
SELECT ...
FROM    ...
```

you can do better

```
SQL> select sql_fulltext
  2  from    v$sqlstats
  3  where   buffer_gets > 1000000 or
  4          executions > 10000 or
  5          disk_reads > 100000;

SQL_FULLTEXT
--------------------------------------------------
SELECT ...
FROM    ...
```

"The column definitions for columns in V$SQLSTATS are identical to those in the V$SQL and V$SQLAREA views. However, the V$SQLSTATS view differs from V$SQL and V$SQLAREA in that it is faster, more scalable, and has a greater data retention (the statistics may still appear in this view, even after the cursor has been aged out of the shared pool)."

12

sqlplus hash tags
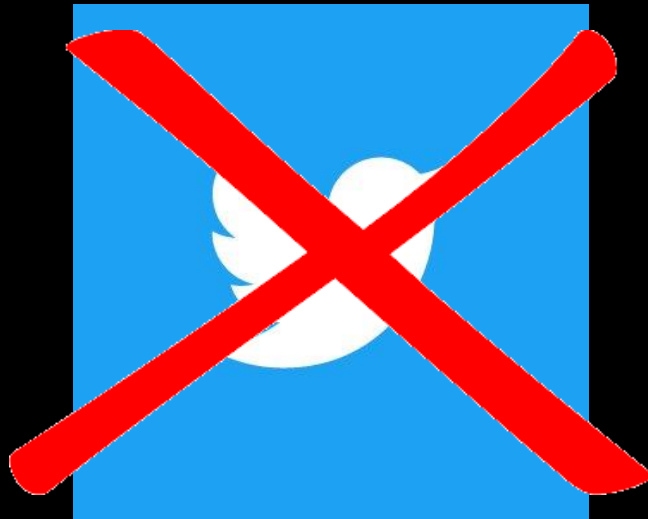
```
SQL> select SAL
   2   from    EMP
   3   where
```
"hmmmm....."

```
SQL> select SAL
  2  from    EMP
  3  where
  4  #desc EMP
 Name                            Null?    Type
 ------------------------------- -------- -------------
 EMPNO                           NOT NULL NUMBER(4)
 ENAME                                    VARCHAR2(10)
 JOB                                      VARCHAR2(9)
  ...
  4  job = 'CLERK';


      SAL
----------
      800
     1100
```

```
SQL> declare
  2     x int;
  3  begin
  4     select max(sal)
  5     into    x
  6     from    emp;
  7
  8     dbms_output.put_line(x);
  9  #set serverout on
  9  end;
 10  /

 10000
```

13

sqlplus error logging

```
SQL> set errorlogging on
```

```
SQL> set errorlogging on

SQL> desc SPERRORLOG

 Name                                        Type
 ---------------------------------------- ----------------
 USERNAME                                    VARCHAR2(256)
 TIMESTAMP                                   TIMESTAMP(6)
 SCRIPT                                      VARCHAR2(1024)
 IDENTIFIER                                  VARCHAR2(256)
 MESSAGE                                     CLOB
 STATEMENT                                   CLOB
```

```
SQL> select * from THE_WRONG_NAME;
select * from THE_WRONG_NAME
              *
ERROR at line 1:
ORA-00942: table or view does not exist

SQL> desc THE_WRONG_NAME;

ERROR:
ORA-04043: object THE_WRONG_NAME does not exist
```

```
SQL> select timestamp, message, statement
  2  from SPERRORLOG;

TIMESTAMP
------------------------------------------------------------
MESSAGE
------------------------------------------------------------
STATEMENT
------------------------------------------------------------
01-APR-08 02.29.58.000000 PM
ORA-00942: table or view does not exist
select * from THE_WRONG_NAME

01-APR-08 02.29.58.000000 PM
ORA-04043: object THE_WRONG_NAME does not exist
desc THE_WRONG_NAME;
```

```
SQL> set errorlogging on
SQL> @create_all_objects
```
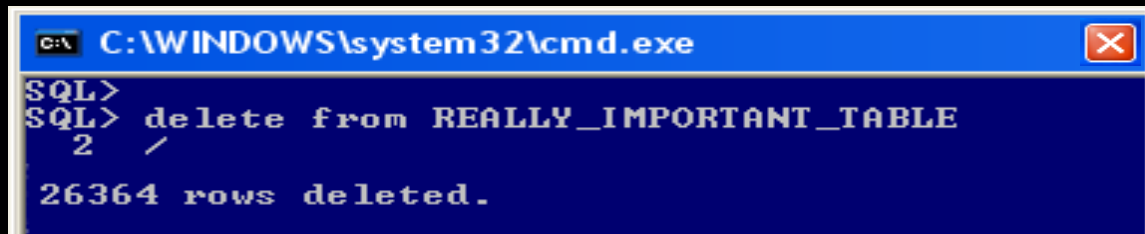
installation scripts

14

sqlplus transaction safety

```
C:\WINDOWS\system32\cmd.exe

SQL>
SQL> delete from REALLY_IMPORTANT_TABLE
  2  /

26364 rows deleted.
```

```
SQL> set exitcommit
```

15

context

8.1.5

"... within a namespace, define, set and access variable-length application attributes and values in a secure data cache available in UGA and SGA."

?

in-memory table structure

context

attributes

values

DETAILS

SURNAME	McDonald
FORENAME	Connor
DOB	20FEB1990

```
SQL> select SYS_CONTEXT('USERENV','SID') sname
  2  from dual;

SID
--------------------
123
```

```
SQL> select SYS_CONTEXT('DETAILS','SURNAME') sname
  2  from dual;

SNAME
--------------------
McDonald
```

default values

if you are lucky

```
SQL> create table PEOPLE
  2  ( person_id  number,
  3    forename1  varchar2(20),
  4    surname    varchar2(30),
  5    created_by varchar2(10) default USER );

Table created.
```

12c

20 frickin' years!

```
SQL> create table PEOPLE
  2  ( person_id  number default per_seq.nextval,
  3    forename1  varchar2(20),
  4    surname    varchar2(30),
  5    created_by varchar2(10) );

Table created.
```

```
SQL> create table PEOPLE
  2  ( person_id  number,
  3    forename1  varchar2(20),
  4    surname    varchar2(30),
  5    created_by varchar2(10)
  6        default <some session specific value> );

Table created.
```

triggers

```
SQL> create or replace
  2  trigger WHO_DID_THIS_ROW
  3  before insert or update
  4  on PEOPLE
  5  for each row
  6  begin
  7    :new.created_by := my_package.global_var;
  8  end;
  9  /

Trigger created.
```

over rated ...

```
insert /* without trigger */ into PEOPLE
select rownum, 'x','y', null
from all_objects
```

call	count	cpu	elapsed	disk	query	current
Parse	1	0.03	0.02	0	0	0
Execute	1	1.39	1.37	0	64619	1257
Fetch	0	0.00	0.00	0	0	0
total	2	1.42	1.40	0	64619	1257

```
insert /* with trigger */ into PEOPLE
select rownum, 'x','y', null
from all_objects
```

call	count	cpu	elapsed	disk	query	current
Parse	1	0.08	0.07	0	0	0
Execute	1	2.73	2.73	0	64631	51763
Fetch	0	0.00	0.00	0	0	0
total	2	2.81	2.80	0	64631	51763

```
SQL> create table PEOPLE
  2  ( person_id  number,
  3    forename1  varchar2(20),
  4    surname    varchar2(30),
  5    created_by varchar2(10)
  6       default <some session specific value> );

Table created.
```

"... within a namespace, define, set and access variable-length application attributes and values in a secure data cache available in UGA and SGA."

```
SQL> create table PEOPLE
  2  ( person_id  number,
  3    forename1  varchar2(20),
  4    surname    varchar2(30),
  5    created_by varchar2(10)
  6      default SYS_CONTEXT('MY_CTX','SOME_ATTRIB') );

Table created.
```

```
insert /* with context */ into PEOPLE
select rownum, 'x','y', null
from all_objects

call      count       cpu    elapsed       disk      query    current
-------  ------  --------  ---------  ---------  ---------  ---------
Parse         1      0.03       0.02          0          0          0
Execute       1      1.38       1.40          0      64623       1257
Fetch         0      0.00       0.00          0          0          0
-------  ------  --------  ---------  ---------  ---------  ---------
total         2      1.41       1.43          0      64623       1257
```

it costs you nothing !

pass parameters to views

```
QL> create or replace
  2  view MY_VIEW as
  3  select ...
  4  from    employees
  5  where   empno in
  6     ( select contractor_empno
  7       from    contractors
  8       where   assigned_mgr = sys_context('EMPCTX','MGR')
  9     )
```

wrap up

poaching ideas is fine

learn something new

I love your feedback!
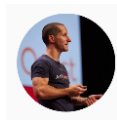
Rate your ~~trip~~ talk with Connor

☆ ☆ ☆ ☆ ☆

Tomorrow !

Don't Be the Person Who Discombobulates Your Database [TRN4096]

SPEAKERS

Connor Mcdonald, Developer Advocate for SQL, Oracle

[+] Thursday, Oct 25, 12:00 p.m. - 12:45 p.m. | Moscone West - Room 3010

aka, how to avoid the silly mistakes
that will destroy a successful database

ORACLE®

Follow and Subscribe!

blog   bit.ly/blog-connor

youtube   bit.ly/youtube-connor

twitter   bit.ly/twitter-connor