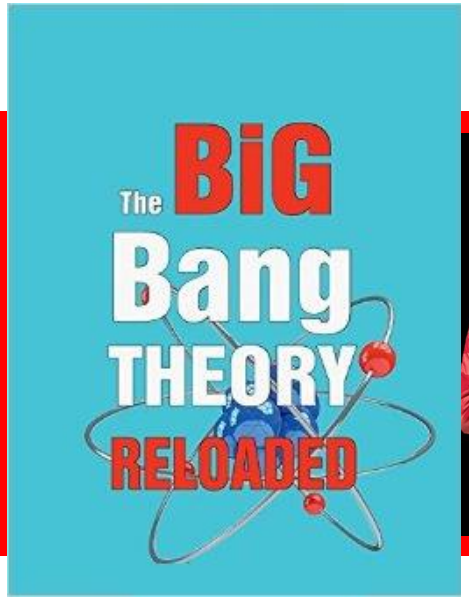


Performance Tuning: Como escribir y correr correctamente una sentencia SQL

By Ronald Vargas Quesada, Oracle ACE Director
Expertise Database Management & Performance
Business Development Manager, Crux Consultores S.A.
Oracledbacr.blogspot.com
@rovaque

OTN Tour 2016

LAOUC

PEOUG
PERU ORACLE USERS GROUP

ORACLE
ACE Director

Performance Tuning: How to write & run correctly

Background

Why ???

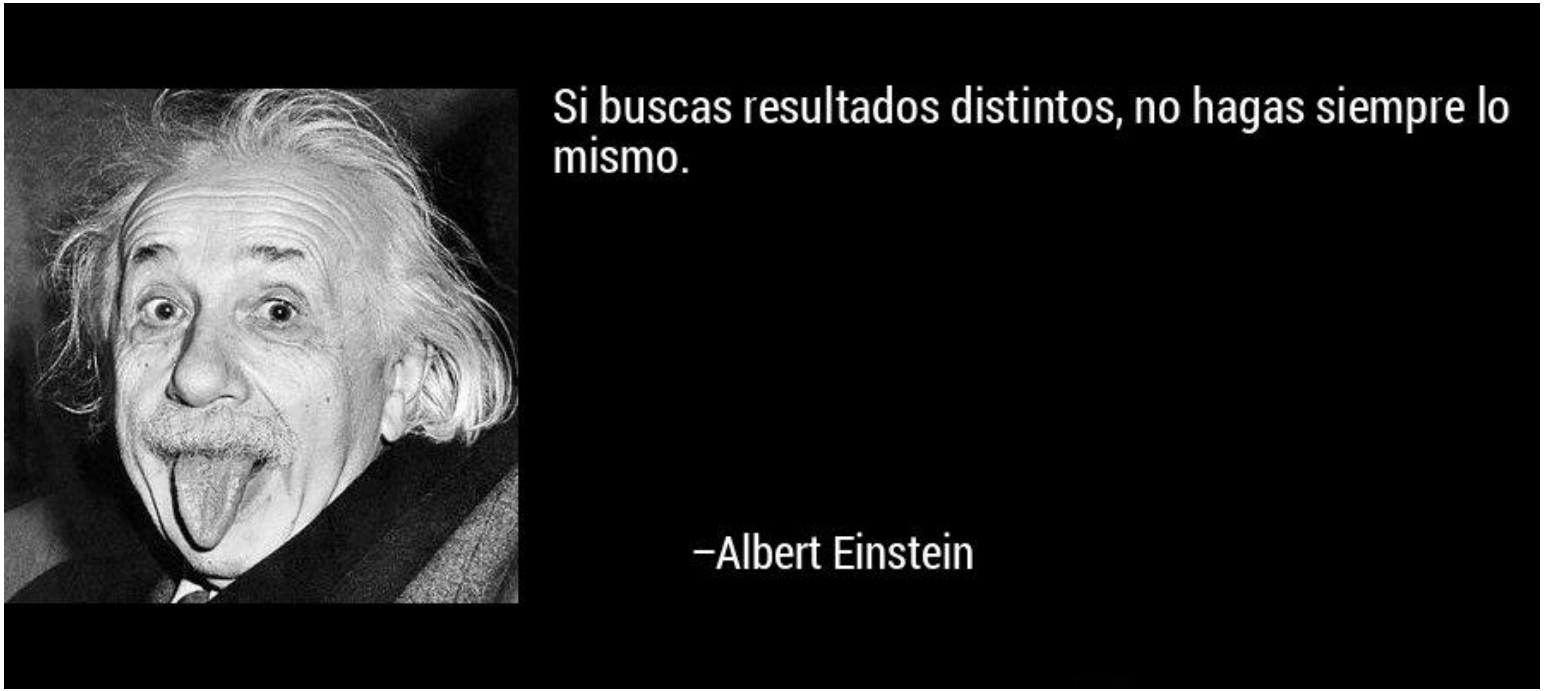
50% mistype a statement

&

9 out of **10** the incorrectly executed

Performance Tuning: How to write & run correctly

Programar siempre y de la misma manera es eficiente ! **No!!**



Performance Tuning: How to write & run correctly

Continuo aprendizaje

Performance Tuning: How to write & run correctly

Declaración #1 absoluta

There's a tradeoff between efficiency (fewer conditional statements) and ease of comprehension.

Existe un equilibrio entre la eficiencia (menos sentencias condicionales) y la facilidad de comprensión.

Performance Tuning: How to write & run correctly

Declaración #2 absoluta

The real world is very complicated; the software we write is supposed to map those complexities into applications. The result is that we often end up needing to deal with convoluted logical expressions.

El mundo real es muy complicado; el software que escribimos supone un mapeo de esas complejidades en las aplicaciones. El resultado es que a menudo terminamos con la necesidad de lidiar con expresiones lógicas complejas.

Performance Tuning: How to write & run correctly

Declaración #3 absoluta

Following this best practice will make your code more readable and expressive.

You avoid redundant code, always bad news in a program, since it increases maintenance costs and the chance of introducing bugs into your code.

Aplicando buenas prácticas se puede lograr que su código sea más legible y expresivo.

Evite el código redundante, este siempre es una mala noticia en un programa, ya que aumenta los costos de mantenimiento y la posibilidad de introducir errores en el código.

Performance Tuning: How to write & run correctly

Declaración #4 absoluta

You need not take out "programmer's insurance": "Gee, I don't know if I need to declare that or not, so I'd better declare it." Instead, you make certain you understand how PL/SQL works and write appropriate code.

"Vaya, no sé si tengo que declararlo o no, así que será mejor que lo declare.“

En su lugar, asegúrese de que entiende cómo funciona el PL/SQL y escriba el código apropiado.

Performance Tuning: How to write & run correctly

Declaración #5 absoluta

Your code doesn't do any unnecessary work and so executes more efficiently.

TIP: You can, in general, expect the performance of built-in functions such as SUBSTR to work more efficiently in SQL than in PL/SQL, so move the processing to the SQL layer whenever possible.

“El código no hace ningún trabajo innecesario y así se ejecuta de manera más eficiente.”

En general el desempeño de las funciones integradas tales como **SUBSTR** trabajan más eficientemente en SQL que en PL/SQL.

Performance Tuning: How to write correctly

That´s it !

That´s it friends!

Performance Tuning: How to write correctly

quick example #1

employees by salary

Performance Tuning: How to write correctly

```
SQL> select employee_id, job_id, hire_date, salary
  2  from employees
  3  order by salary;

EMPLOYEE_ID JOB_ID     HIRE_DATE     SALARY
----------- ---------- --------- ----------
        132 ST_CLERK   10-APR-07       2100
        136 ST_CLERK   06-FEB-08       2200
        128 ST_CLERK   08-MAR-08       2200
        127 ST_CLERK   14-JAN-07       2400
        135 ST_CLERK   12-DEC-07       2400
        191 SH_CLERK   19-DEC-07       2500
        119 PU_CLERK   10-AUG-07       2500
        140 ST_CLERK   06-APR-06       2500
        144 ST_CLERK   09-JUL-06       2500
        182 SH_CLERK   21-JUN-07       2500
        131 ST_CLERK   16-FEB-05       2500
```

Performance Tuning: How to write correctly

Who hired first ?

hiring sequence

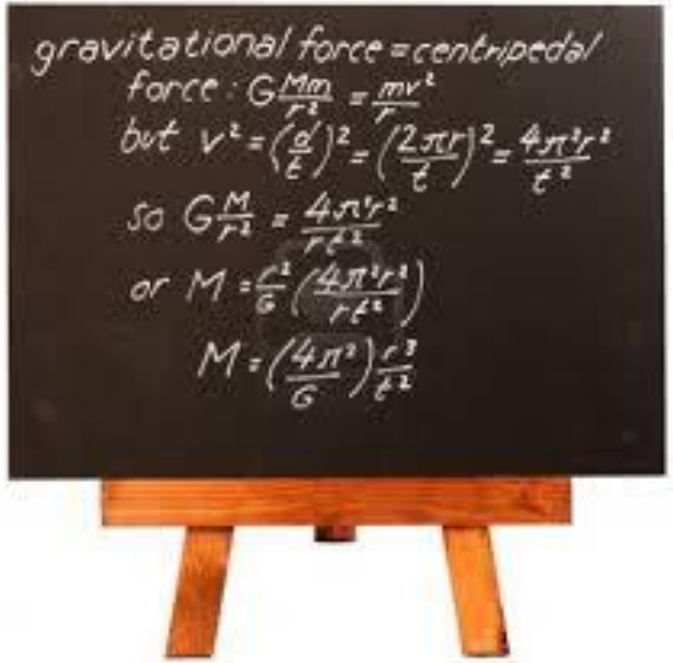
Performance Tuning: How to write correctly

```
 1  select e.employee_id, e.first_name, e.job_id, e.hire_date, e.salary, x.sequence
  2  from employees e,
  3  (select e2.employee_id, count(*) sequence from employees e1, employees e2
  4  where e1.hire_date <= e2.hire_date
  5  group by e2.employee_id
  6  ) x
  7  where e.employee_id = x.employee_id
  8* order by salary
SQL> /

EMPLOYEE_ID FIRST_NAME           JOB_ID     HIRE_DATE     SALARY   SEQUENCE
----------- -------------------- ---------- --------- ---------- ----------
        132 TJ                   ST_CLERK   10-APR-07       2100         85
        128 Steven               ST_CLERK   08-MAR-08       2200        104
        136 Hazel                ST_CLERK   06-FEB-08       2200        102
        135 Ki                   ST_CLERK   12-DEC-07       2400         95
        127 James                ST_CLERK   14-JAN-07       2400         78
        144 Peter                ST_CLERK   09-JUL-06       2500         71
```

very complicated !!

Performance Tuning: How to write correctly

Is very simple ?



Performance Tuning: How to write correctly

Much or little work ?

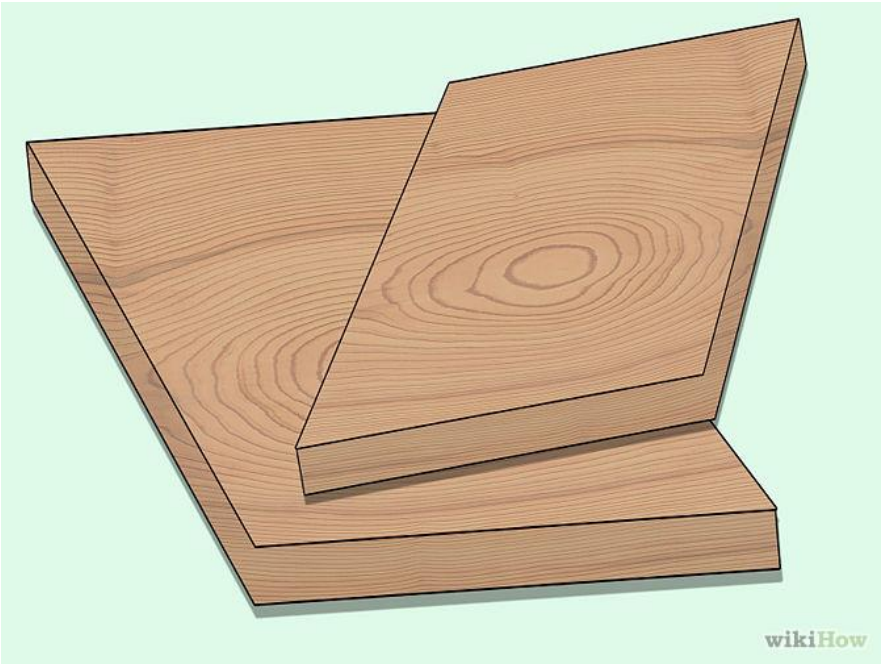
Performance Tuning: How to write correctly

```
PLAN_TABLE_OUTPUT
-------------------------------------------------------------------------------------------
Plan hash value: 2074273239
-------------------------------------------------------------------------------------------
| Id  | Operation                | Name      | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT         |           |   107 |  5243 |    14  (36)| 00:00:01 |
|   1 |  SORT ORDER BY           |           |   107 |  5243 |    14  (36)| 00:00:01 |
|*  2 |   HASH JOIN              |           |   107 |  5243 |    13  (31)| 00:00:01 |
|   3 |    TABLE ACCESS FULL     | EMPLOYEES |   107 |  3424 |     3   (0)| 00:00:01 |
|   4 |    VIEW                  |           |   107 |  1819 |     9  (34)| 00:00:01 |
|   5 |     HASH GROUP BY        |           |   107 |  2140 |     9  (34)| 00:00:01 |
|   6 |      MERGE JOIN          |           |  5783 |   112K|     8  (25)| 00:00:01 |
|   7 |       SORT JOIN          |           |   107 |   856 |     4  (25)| 00:00:01 |
|   8 |        TABLE ACCESS FULL| EMPLOYEES |   107 |   856 |     3   (0)| 00:00:01 |
|*  9 |       SORT JOIN          |           |   107 |  1284 |     4  (25)| 00:00:01 |
|  10 |        TABLE ACCESS FULL| EMPLOYEES |   107 |  1284 |     3   (0)| 00:00:01 |
-------------------------------------------------------------------------------------------
Predicate Information (identified by operation id):
---------------------------------------------------

   2 - access("E"."EMPLOYEE_ID"="X"."EMPLOYEE_ID")
   9 - access("E1"."HIRE_DATE"<="E2"."HIRE_DATE")
       filter("E1"."HIRE_DATE"<="E2"."HIRE_DATE") 24 rows selected.
```

3X FTS EMPLOYEES

Performance Tuning: How to write correctly

Make it Simple !!



Performance Tuning: How to write correctly

```
SQL> select employee_id, first_name, job_id, hire_date, salary, rank() over (order by
hire_date) as hire_seq
  2  from employees
  3  order by salary;

EMPLOYEE_ID FIRST_NAME           JOB_ID     HIRE_DATE     SALARY   HIRE_SEQ
----------- -------------------- ---------- --------- ---------- ----------
        132 TJ                   ST_CLERK   10-APR-07       2100         85
        136 Hazel                ST_CLERK   06-FEB-08       2200        102
        128 Steven               ST_CLERK   08-MAR-08       2200        104
        127 James                ST_CLERK   14-JAN-07       2400         78
        135 Ki                   ST_CLERK   12-DEC-07       2400         95
        131 James                ST_CLERK   16-FEB-05       2500         28
        140 Joshua               ST_CLERK   06-APR-06       2500         65
        144 Peter                ST_CLERK   09-JUL-06       2500         71
```

Performance Tuning: How to write correctly

```
SQL> select * from table(dbms_xplan.display);

PLAN_TABLE_OUTPUT
--------------------------------------------------------------------------------
Plan hash value: 1701542182

-------------------------------------------------------------------------------
| Id  | Operation           | Name      | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT    |           |   107 |  3424 |     5  (40)| 00:00:01 |
|   1 |  SORT ORDER BY      |           |   107 |  3424 |     5  (40)| 00:00:01 |
|   2 |   WINDOW SORT       |           |   107 |  3424 |     5  (40)| 00:00:01 |
|   3 |    TABLE ACCESS FULL| EMPLOYEES |   107 |  3424 |     3   (0)| 00:00:01 |
-------------------------------------------------------------------------------


10 rows selected.
```

Performance Tuning: How to write correctly

When Should I Use an Index?

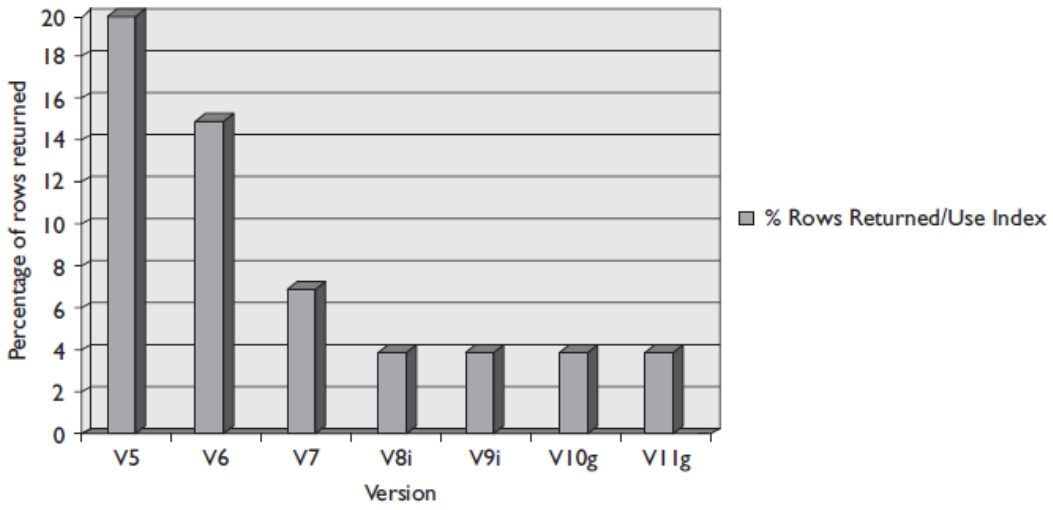
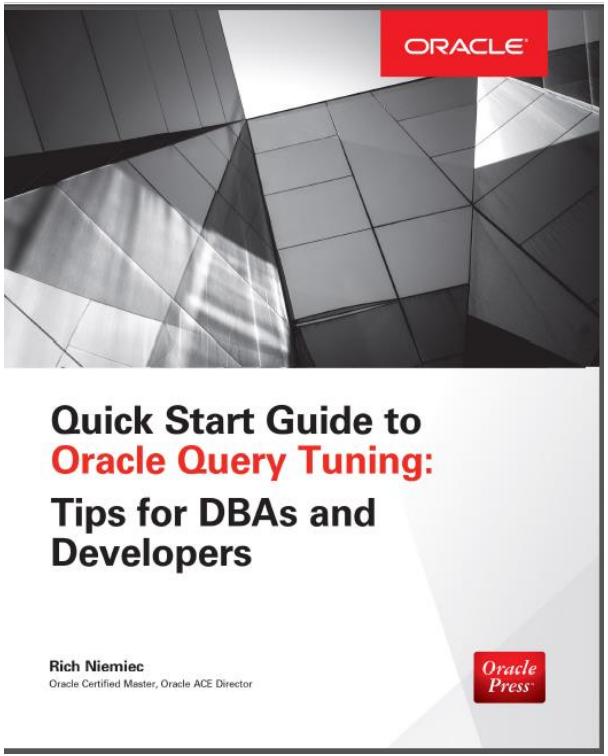
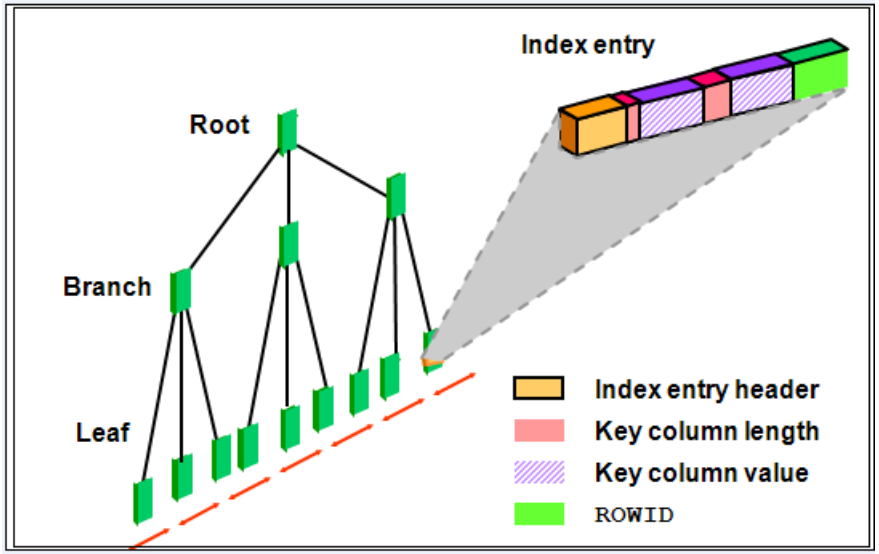


FIGURE 2-1. *When to generally use an index based on the percentage of rows returned by a query*



Performance Tuning: How to write correctly

When Should I Use an Index?



- No es un puntero
- Segmento con almacenamiento en un tablespace
- Parámetros de concurrencia
- Mejoran el acceso a los datos
- Degrada gradualmente los procesos de Update, Insert y Delete de tuplas.
- No son libres de mantenimiento
- +20% entradas inválidas, el motor omite el uso del índice.
- Requieren cada cierto tiempo recrearse para balancear el árbol y eliminar las entras inválidas

Performance Tuning: How to write correctly

When Should I Use an Index?

Optimización basada en RULE (Regla) 10gR2 o inferior

a	a	a	a	a	b	b	b	b
	b	b	b	b	a	a	a	a
		c	c	c		c	c	c
			d	d				d
				e			...	

Performance Tuning: How to write correctly

When Should I Use an Index?

Optimización basada en COST (Costo) 7.x o superior,

11g y superior sólo optimiza basado en COSTO

a

b

c

d

e

Un columna que es llave foránea Siempre debe estar indexada.

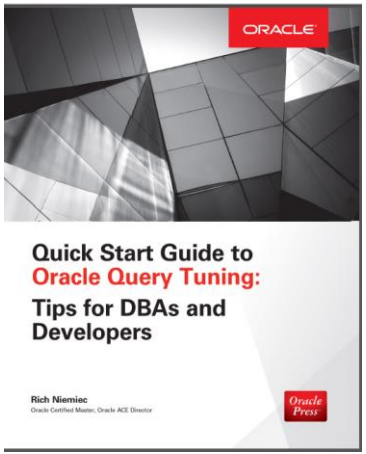
Performance Tuning: How to write correctly

Driving Table, (RBO Only)

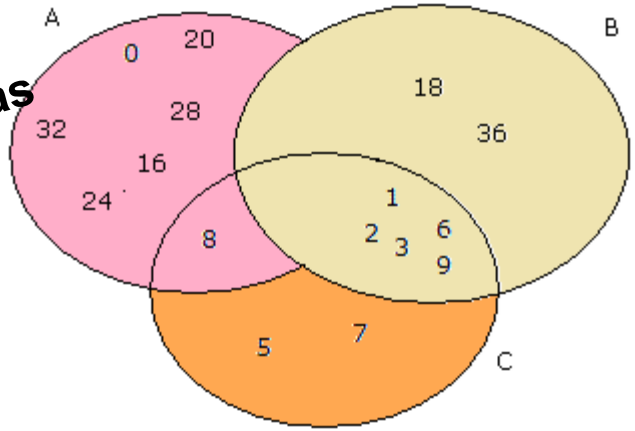
-5 % de consultas

TIP

Using cost-based optimization, when a large table and a small table are joined, the smaller table is the driving table (accessed first), and the smaller table is used to build a hash table in memory on the join key.



3 tablas o + involucradas



Performance Tuning: How to write correctly

Parseo de instrucciones

La fase de análisis sintáctico para declaraciones se puede disminuir mediante el uso eficiente de alias. Si un alias no está presente, el motor debe resolver qué tablas poseen las columnas especificadas. El siguiente es un ejemplo.

Bad Statement

```
SELECT first_name,
       last_name,
       country
FROM   employee,
       countries
WHERE  country_id = id
AND    lastname   = 'HALL';
```

Good Statement

```
SELECT e.first_name,
       e.last_name,
       c.country
FROM   employee e,
       countries c
WHERE  e.country_id = c.id
AND    e.last_name  = 'HALL';
```

Performance Tuning: How to write correctly

Exists vs IN

```
TABLE1 - 1000 rows
TABLE2 - 1000 rows
```

```
(A)
SELECT t1.id
FROM   table1 t1
WHERE  t1.code IN (SELECT t2.code
                   FROM   table2 t2);
```

```
(B)
SELECT t1.id
FROM   table1 t1
WHERE  EXISTS (SELECT '1'
               FROM   table2 t2
               WHERE  t2.code = t1.code)
```

- Para la **consulta A**, todas las filas de TABLA2 serán leídos por cada fila en la Tabla 1. El efecto será 1.000.000 filas leídas de artículos.
- En el caso de la **consulta B**, se leerá un máximo de 1 registro de la TABLA2 para cada fila en la TABLA1, reduciendo así la carga de procesamiento de la declaración

Si la mayoría de los criterios de filtrado se encuentran en la subconsulta entonces la variación **IN** es más eficiente.

Si la mayoría de los criterios de filtrado están en la consulta entonces la variación de **EXISTS** es más eficiente.

Se sugiere que usted debe tratar ambas variantes y ver cuál funciona mejor.

Performance Tuning: How to write correctly

Not using bind variable

```
SQL> host
[oracle@LAB1 ~]$ vi script1.sql
[oracle@LAB1 ~]$ more script1.sql
set timing on
begin
for i in 1 .. 100000
loop
        execute immediate
        'insert into t (x,y)
        values ( ' || i || ', ''x'') ';
end loop;
end;
/
[oracle@LAB1 ~]$
```

Parsing 100K veces

Performance Tuning: How to write correctly

Using Bind variable

```
[oracle@LAB1 ~]$ more script2.sql
set timing on
begin
        for i in 1 .. 100000
        loop
                execute immediate
                'insert into t (x,y)
                values ( :i, ''x'')'
                using i;
        end loop;
end;
/

[oracle@LAB1 ~]$
```

Parsing 1X !!!

(*) Up to 1320% higher

Performance Tuning: How to run correctly

PLSQL_CODE_TYPE

PLSQL_CODE_TYPE specifies the compilation mode for PL/SQL library units.

Syntax	PLSQL_CODE_TYPE = { INTERPRETED | NATIVE }
Default	INTERPRETED
Modifiable	Alter session, Alter system
Basic	No

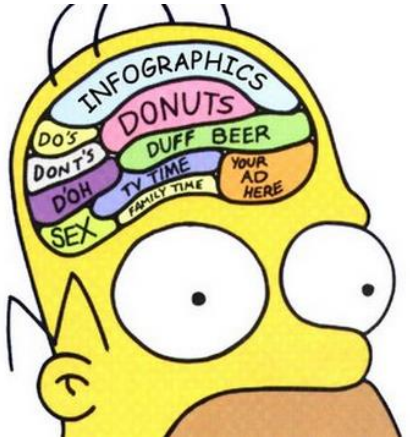
INTERPRETED = interpreter engine / **NATIVE** = native machine code, without incurring any interpreter overhead

Performance Tuning: How to run correctly

PLSQL_CODE_TYPE

When the value of this parameter is changed, it has no effect on PL/SQL library units that have already been compiled. The value of this parameter is stored persistently with each library unit.

If a PL/SQL library unit is compiled native, all subsequent automatic recompilations of that library unit will use native compilation.



Performance Tuning: How to run correctly

Environment

```
[oracle@lab1 ~]$ sqlplus /nolog

SQL*Plus: Release 12.1.0.1.0 Production on Wed Apr 23 17:19:08 2014

Copyright (c) 1982, 2013, Oracle.  All rights reserved.

SQL> connect / as sysdba
Connected to an idle instance.
SQL> startup
ORACLE instance started.

Total System Global Area 2087780352 bytes
Fixed Size                  2290264 bytes
Variable Size            1291849128 bytes
Database Buffers          788529152 bytes
Redo Buffers                5111808 bytes
Database mounted.
Database opened.
SQL> exit
Disconnected from Oracle Database 12c Enterprise Edition Release 12.1.0.1.0 - 64bit Production
With the Partitioning, OLAP, Advanced Analytics and Real Application Testing options
```

Performance Tuning: How to run correctly

Example

```
CREATE OR REPLACE FUNCTION testeo ( n positive ) return integer
is
begin
if ( n = 1) OR ( n = 2 ) then
return 1;
else
return testeo(n-1) + testeo(n-2);
end if;
end testeo;
/
```

Performance Tuning: How to run correctly

Example

```
set serveroutput on
set timing on

--- anonymous block
declare
x number;
begin
x := testeo(40);
dbms_output.put_line(x);
end;
/
```

Performance Tuning: How to run correctly

Example

```
[oracle@lab1 admin]$ sqlplus hr/hr@PDB1

SQL*Plus: Release 12.1.0.1.0 Production on Wed Apr 23 17:23:43 2014

Copyright (c) 1982, 2013, Oracle.  All rights reserved.

Last Successful login time: Wed Jan 15 2014 12:14:36 -06:00

Connected to:

Oracle Database 12c Enterprise Edition Release 12.1.0.1.0 - 64bit
Production

With the Partitioning, OLAP, Advanced Analytics and Real Application
Testing options

SQL>
```

Performance Tuning: How to run correctly

Example

```
SQL> @p1.pl
Function created.

SQL> alter session set plsql_code_type = 'INTERPRETED';
Session altered.

SQL> alter function testeo compile;
Function altered.
```

Performance Tuning: How to run correctly

Example

```
SQL> set linesize 200
SQL> @ver_parametro.pl
```

OBJECT_NAME	PARAM_NAME	PARAM_VALUE
TESTEO	plsql_optimize_level	2
TESTEO	plsql_code_type	INTERPRETED
TESTEO	plsql_debug	FALSE
TESTEO	nls_length_semantics	BYTE
TESTEO	plsql_warnings	DISABLE:ALL
TESTEO	plsql_ccflags	
TESTEO	plscope_settings	IDENTIFIERS:NONE
TESTEO	plsql_compiler_flags	INTERPRETED,NON_DEBUG

8 rows selected.

Performance Tuning: How to run correctly

Example

SQL> @ej1.pl
102334155

PL/SQL procedure successfully completed.
Elapsed: **00:00:52.22**

SQL> alter session set plsql_code_type = '**NATIVE**';
Session altered.

SQL> alter function testeo compile;
Function altered.

SQL> @ej1.pl
102334155

PL/SQL procedure successfully completed.
Elapsed: **00:00:27.73**

Performance Tuning: How to run correctly

Example (Do not forget: COMPILE)

SQL> @ej1.pl R/ 102334155

PL/SQL procedure successfully completed.
Elapsed: **00:00:56.43**

SQL> alter session set plsql_code_type = '**NATIVE**';
Session altered.

SQL> @ej1.pl R/ 102334155

PL/SQL procedure successfully completed.
Elapsed: **00:00:57.40**

SQL> **alter function testeo compile**;
Function altered.

SQL> @ej1.pl R/ 102334155

PL/SQL procedure successfully completed.

Elapsed: **00:00:29.28**
SQL>

Performance Tuning: How to run correctly

Example 2 (Compiled='**NATIVE**')

```
create or replace procedure insertar
as
begin
for i in 1 .. 100000
loop
        execute immediate
        'insert into t(x,y)
        values (' || i ||',''x'')';
end loop;
end;
/
```

```
SQL> execute insertar;
PL/SQL procedure successfully completed.
                                Elapsed: 00:01:12.61
```

Performance Tuning: How to run correctly

Example 2, (Compiled =‘**NATIVE**’)

```
create or replace procedure insertarb
as
begin
for i in 1 .. 100000
loop
        execute immediate
        'insert into t(x,y)
        values (:i,''x'')'
        using i;
end loop;
end;
/
```

```
SQL> execute insertarb;
PL/SQL procedure successfully completed.
Elapsed: 00:00:05.05
```

Performance Tuning: How to run correctly

Result_Cache

The Result Cache is a new memory structure in the SGA.

By adding the **RESULT_CACHE** clause to your function, Oracle will:

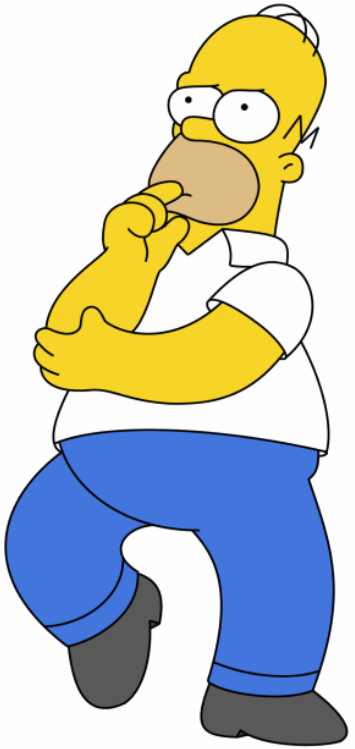
- Cache the values returned by the function, along with the input values.
- Return the cached values if the same input values are provided – that is, not execute the function.
- Share this cache among all sessions in the instance.

Performance Tuning: How to run correctly

Result_Cache

```
SQL> select count(*) from hr.employees2;

  COUNT(*)
----------
  10700214

Elapsed: 00:00:00.46
SQL> host
[oracle@lab1 ~]$ more cache2_result.sql

SELECT count(*)
  FROM ( SELECT  department_id, manager_id,
           count(*) count
           FROM hr.employees
           GROUP BY department_id, manager_id )
           view1
 WHERE department_id <> 30;
```

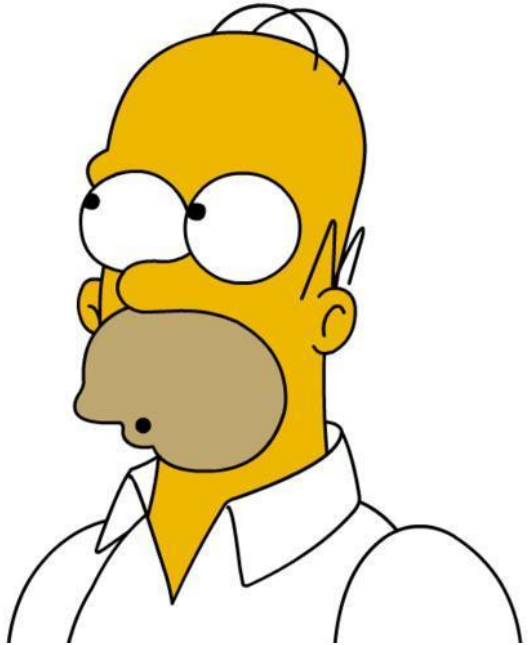
```
SQL> connect hr/hr@pdb1
Connected.
SQL> @cache2_result

  COUNT(*)
----------
        25

Elapsed: 00:00:02.04
```



Performance Tuning: How to run correctly

Result_Cache

```
SQL> connect system/oracle@pdb1
Connected.
SQL> alter system flush buffer_cache;

System altered.

Elapsed: 00:00:00.03

SQL> connect hr/hr@pdb1
Connected.
SQL> @cache2_result

  COUNT(*)
----------
        25

Elapsed: 00:00:02.11
```



Performance Tuning: How to run correctly

Result_Cache

```
SQL> host
[oracle@lab1 ~]$ more cache2_result.sql

CREATE OR REPLACE FUNCTION f1 RETURN number is
x number;
begin
SELECT count(*)
INTO x
  FROM ( SELECT  /*+ RESULT_CACHE */
           department_id, manager_id, count(*)
           count
           FROM hr.employees2
           GROUP BY department_id, manager_id )
                                  view1
           WHERE department_id <> 30;
           return x;
end f1;
```

```
SQL> @cache2_result
Function created.
```

Performance Tuning: How to run correctly

Result_Cache

```
SQL> select f1 from dual;

        F1
----------
        25

Elapsed: 00:00:03.45

SQL> select f1 from dual;

        F1
----------
        25

Elapsed: 00:00:00.00
```

```
SQL> connect system/oracle@pdb1
Connected.

SQL> alter system flush buffer_cache;

System altered.

Elapsed: 00:00:00.38
```

Performance Tuning: How to run correctly

Qué es más eficiente ?

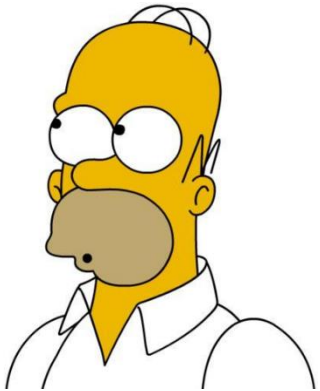
```
FOR item IN (
   SELECT SQRT(department_id) col_alias
   FROM (SELECT DISTINCT department_id FROM empleados)
   ORDER BY col_alias
 )
 LOOP
```

```
FOR item IN (
   SELECT DISTINCT(SQRT(department_id)) col_alias
   FROM empleados
   ORDER BY col_alias
 )
```

Performance Tuning: How to run correctly

Hacer magia

- EXEC DBMS_STATS.gather_table_stats('HR', 'EMPLOYEES');
- Alter table tabla_magica result_cache (mode force);



Performance Tuning: How to run correctly

Result_Cache

```
SQL> connect hr/hr@pdb1
Connected.
SQL> select f1 from dual;

        F1
----------
        25

Elapsed: 00:00:00.01
```

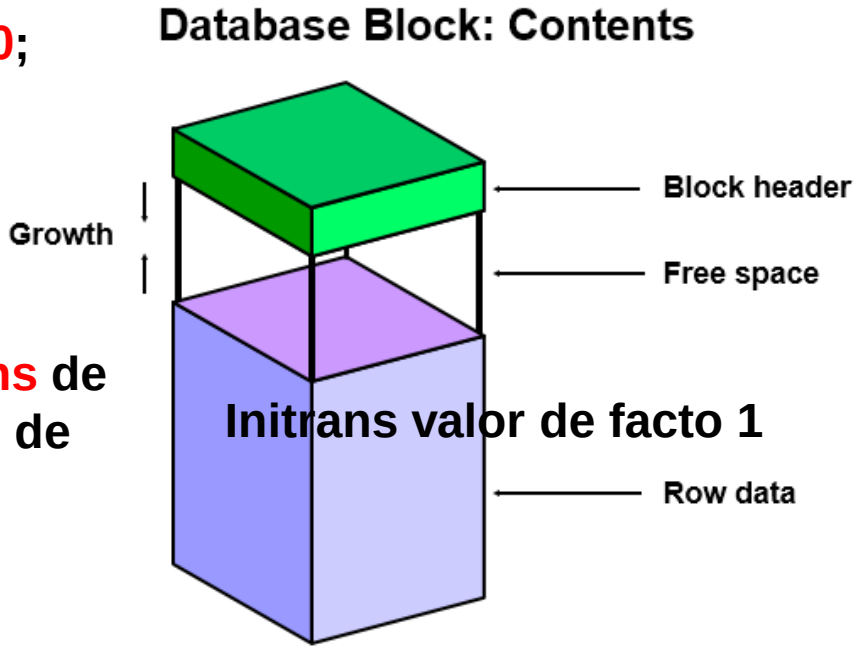
```
SQL> connect system/oracle@pdb1
Connected.
SQL> select hr.f1 from dual;

        F1
----------
        25

Elapsed: 00:00:00.02
```

Performance Tuning: How to run correctly

Concurrencia: Evitando bloqueos

Alter table tabla1 initrans 10;

Tip: Cambiar initrans de los índices al doble de la tabla



@rovaque
oracledbacr.blogspot.com
rvargas@netsoftca.com