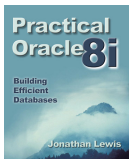
The Database gets better, but the Metrics look worse.



Jonathan Lewis

www.jlcomp.demon.co.uk

Who am I

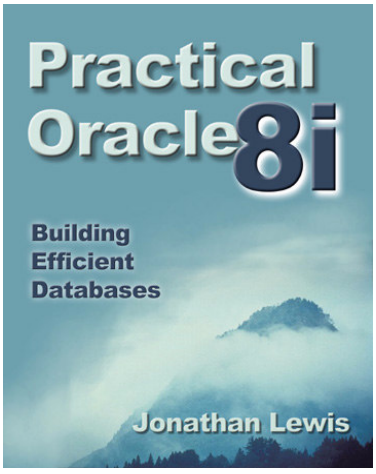
Freelance consultant

17+ years experience

Design, Strategy, and Trouble-shooting

Seminars / Tutorials

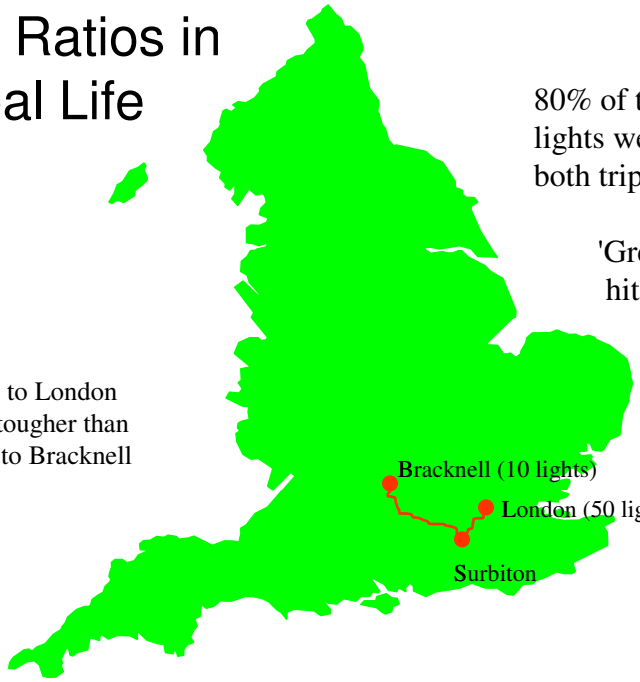
www.jlcomp.demon.co.uk



Highlights

A real life example
What is a ratio
What is a logical I/O
Can you trust the statistics anyway
A relevant ratio
Q and A

Hit Ratios in Real Life



80% of the traffic lights were red on both trips

'Green light' hit ratio 20%

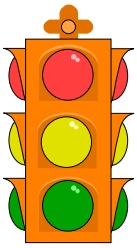
Driving to London is a lot tougher than driving to Bracknell

Hit Ratios in Real Life

A failed attempt to improve the definition.

Held up	7
Straight through	3
Total places	10

Green Light Hit ratio = 30%



Saw red	19
Saw green	15
Total events	34

Long queues at some traffic lights

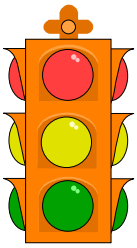
Green Light Hit ratio = 15/34 = 44%

Hit Ratios in Real Life

A better attempt to improve the definition.

Some green lights are less green than others.

Real Reds	19
Good greens	3
Bad greens	12 (= 6 reds)
Total Score	3 + 19 + 6 = 28



Pass Through Hit ratio = 3/28 = 10.7%

Two key problems

- Ratios always lose information
  - is London easier to reach than Bracknell ?
- The meaning of the underlying data may be badly defined
  - When is a green light not green enough ?

What is a ratio ?

X/Z	= 75/95	= 0.7895
(Z-Y)/Z	= (95 - 20)/(95)	= 0.7895

How can we ***'improve'*** the ratio ?

Increase Z by 5: (100-20)/100 = 0.80

Decrease Y by 1: (95-19)/95 = 0.80

One of these changes *might* be good for your system

Misleading ratios (1)

We do less work, but the cache hit ratio has dropped.

Cache Hit Ratio =
100 * (1 - (physical reads / logical reads))
100 * (1 - 10,000 / 1,000,000) = 99%

Let's tune the SQL to do a quarter of the work:

1,000,000 -> 250,000

CHR = 100 * (1 - 10,000 / 250,000) = 96%

Misleading ratios (2)

The cache hit ratio has gone up, but is it a good trade.

Cache Hit Ratio =
100 * (1 - (physical reads / logical reads))
100 * (1 - 10,000 / 100,000) = 90%

Let's trade physical I/O against logical I/O:

10,000 -> 5,000, 100,000 -> 500,000

CHR = 100 * (1 - 5,000 / 500,000) = 99%

Stress Test (v8)

Be careful how many logicals you trade for a physical

```
create table kill_cpu (n, primary key(n))
organization index as
select rownum n from all_objects
where rownum <= 23;

select count(*) X from kill_cpu
connect by n > prior n
start with n = 1;
```


Results

Extreme logical I/O can result in extreme latching

HP D370 - 2 x 160MHz CPU - HPUX 11.00 64 bit.

```
  CPU time          3 mins 20 seconds
  Elapsed time      3 mins 20 seconds
```

top showed one CPU idle, one at 100%.

Two copies

```
  Elapsed 1:  3:27    Wait time:  6.12 secs
  Elapsed 2:  3:57    Wait time: 31.70 secs
```

top showed a total idle time of 8%.

Observations 1

Tuning ***for*** a cache hit ratio is not good

Tuning for a cache hit ratio can:

- increase logical I/O - which is not free
- increase contention

Rate this instance.

This is three design errors hiding each other

```
            Buffer Nowait %:  100.00
            Buffer  Hit   %:  100.00
            Library Hit   %:   99.29
         Execute to Parse %:   98.02
Parse CPU to Parse Elapsd %:   97.06

              Redo NoWait %:  100.00
            In-memory Sort %:  100.00
                Soft Parse %:   98.96
                 Latch Hit %:  100.00
             % Non-Parse CPU:   98.68
```

Missing logical I/O

How should you count logical I/O

select small_vc from pin_c where id > 0;

Run cycle 1	Set arraysize 1
consistent gets	504
buffer is pinned count	498
buffer is not pinned count	752

Run cycle 2	Set arraysize 500
consistent gets	23
buffer is pinned count	980
buffer is not pinned count	26

How does logical I/O work

Should you worry about buffer accesses or latch gets

```
select    count(t.padding)
from      driver d,  target t
where     t.id = d.xref ;
```

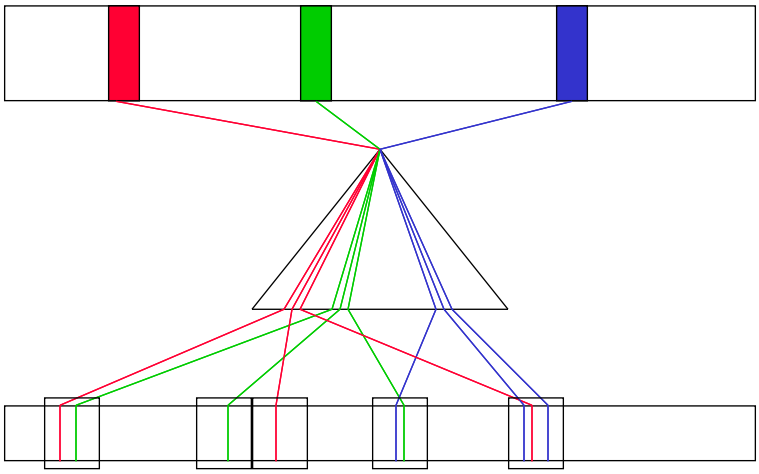
Scan DRIVER

Unique index into TARGET

Version	CR Gets	Buffer pinned	Latches
8.1.7.0	9,000	0	18,000
8.1.7.3	6,000	3,000	12,000
9.0.1.2	6,000	3,000	9,000
9.0.1.3 (9.2)	3,000	9,000	6,000

Indexed Nested Loop v9

Table pre-fetching - stops at the index and sorts by block



Updates and rollback

Should we try to cater for undo blocks in the calculation ?

```
update t1 set id = id + 1;
3000 rows updated.
```

T1 is not indexed

```
db block gets                                3,213
consistent gets                                221
db block changes                             6,100
redo entries                                 3,015
redo size                                  750,324
rollback changes - undo records applied          0
buffer is not pinned count                     203
```

Updates and rollback

Rolling back gives a better BCHR than doing the update !

```
rollback;
Rollback complete.
```

```
db block gets                                6,080
consistent gets                                 30
db block changes                             6,001
redo entries                                 3,001
redo size                                  526,588
rollback changes - undo records applied      3,000
buffer is not pinned count                       0
```


Commit Cleanout

```
update   /*+ index(t1) */ t1
set      small_vc = small_vc + 1;
commit;
```

```
session logical reads                        1
db block gets                                1
db block changes                             1

commit cleanouts successfully completed    180

cache buffers chains (gets)                183
```

Read Consistency

Mixing reports with OLTP can make the BCHR look good

Session 1	Session 2
Set TX read only	
	update row **R** x 2000
	(no commit needed)
select row **R**	

```
consistent gets                          2,003
consistent changes                       2,000
data blks cons reads - undo recs app     2,000
consistent gets - examination            2,000  v9
```


Sysdate, User and DUAL

Some system suffer from lots of redundant logical I/O

```
declare
        m_date          date;
        m_user          varchar2(32);
begin
        for i in 1..10000 loop
                m_date := sysdate;
                m_user := user;
        end loop;
end;
```

Which stats do you trust

Different reports on the pl/sql from the previous page.

Oracle 9.2.0.1

v$sysstat results

consistent gets	30,000
no work - consistent read gets	10,000

v$buffer_pool_statistics

DB_BLOCK_GETS	30,000
CONSISTENT_GETS	30,000

(fixed in 9.2.0.2)

Observations 2

- There are plenty of inefficient ways of getting a high cache hit ratio
- There are plenty of errors in the raw input to the cache hit ratio.
- The cache hit ratio does not tell you about the efficiency of your system.

The FAN hit ratio

```
100 * (
  1 - least (1,
                calculated response time /
                actual response time
            )
) = FHR
```


The Fan Hit Ratio

Examples of General Usage

Calculated	Actual	FHR
3 seconds	12 secs	75%
4 minutes	5 mins	20%
0.12 sec	0.1 sec	0%

For the FHR, zero is best, 100% is worst.

The Fan Hit Ratio

But any ratio can be misleading.

Calculated	Actual	FHR
0.001 sec	0.01 sec	90%
100 min	1,000 min	90%
4 hours	5 hours	20%

Warnings aside -

What does the FAN Hit Ratio represent ?

What should the FHR be ?

Typical Systems - 50%

Big Systems - need a B/S factor

Relevant considerations:

Ignorance of infrastructure

Redundant work

Rows needed vs. Rows visited

Quality of indexes / Timing of data elimination

Scope for contention

The FP Hit Ratio

100 * (
  1 - least (1,
    desired response time /
    actual response time
)
)

The FP Hit Ratio

Required	**Calculated**	**Actual**
< 1 sec	31 seconds	33 seconds

```
FHR      = 100*(1 - 31/33) = 6%
FPHR     = 100*(1 - 1/33)  = 97%
```

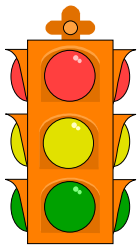
Sometimes the best way to tune the system is to tell the users how much work the database is doing - in business terms.

Traffic Lights revisited

Because 'stalled' green lights last half the time of red lights.

Some green lights are less green than others.

Real Reds	19
Good greens	3
Bad greens	12 (= 6 reds)
Total Score	3 + 19 + 6 = 28



Pass Through Hit ratio = 3/28 = 10.7%

Conclusions

- If you can't ***predict*** response times, you can't keep the users happy.
- Despite the choice of name, the FAN hit ratio is not a joke.