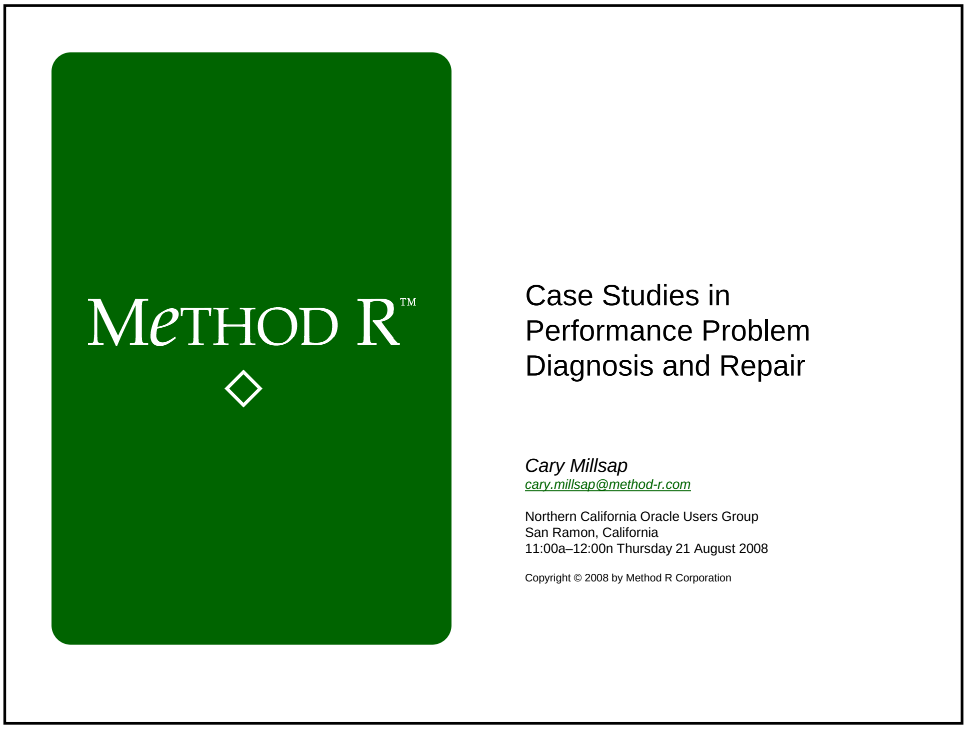
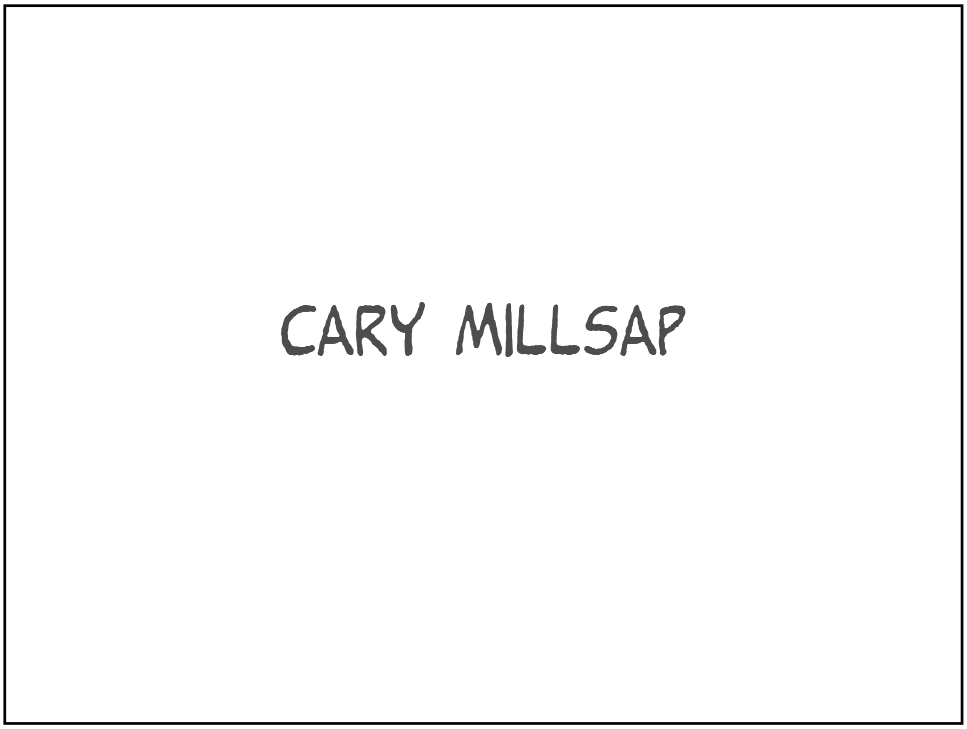


METHOD R™

Case Studies in Performance Problem Diagnosis and Repair

Cary Millsap
cary.millsap@method-r.com

Northern California Oracle Users Group
San Ramon, California
11:00a–12:00n Thursday 21 August 2008



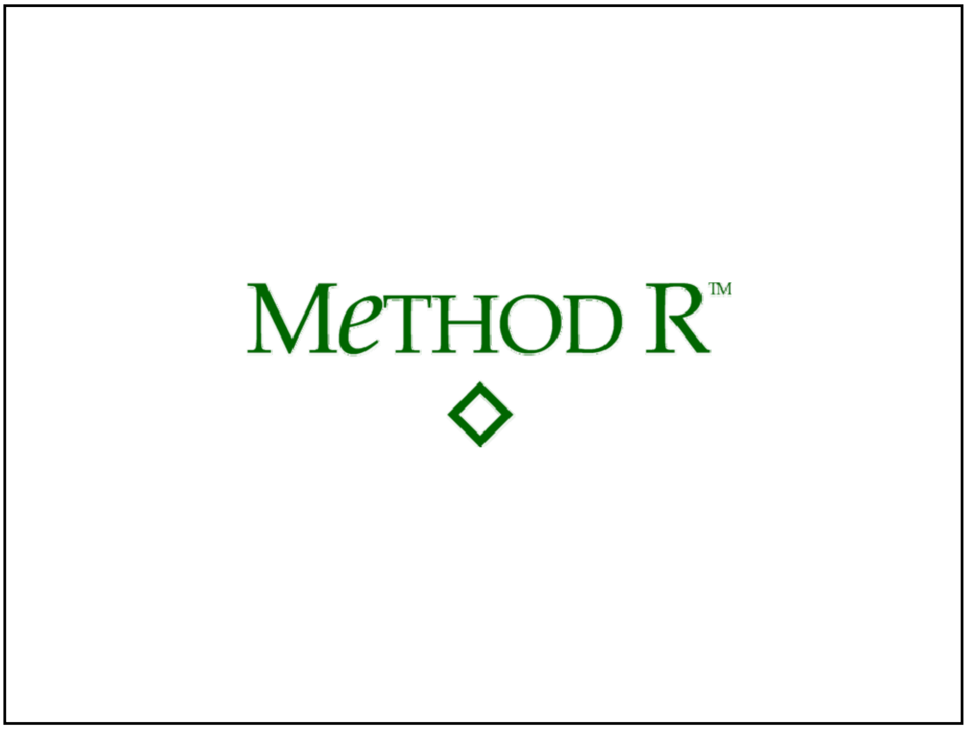
CARY MILLSAP

http://method-r.com

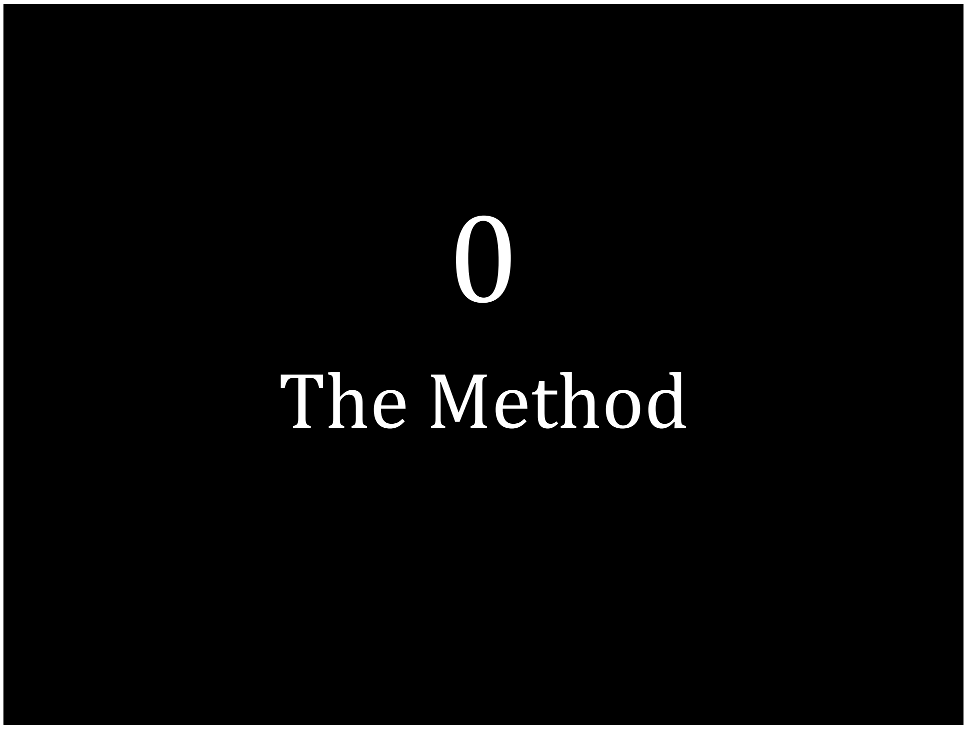
http://carymillsap.blogspot.com

http://karenmorton.blogspot.com

cary.millsap@method-r.com

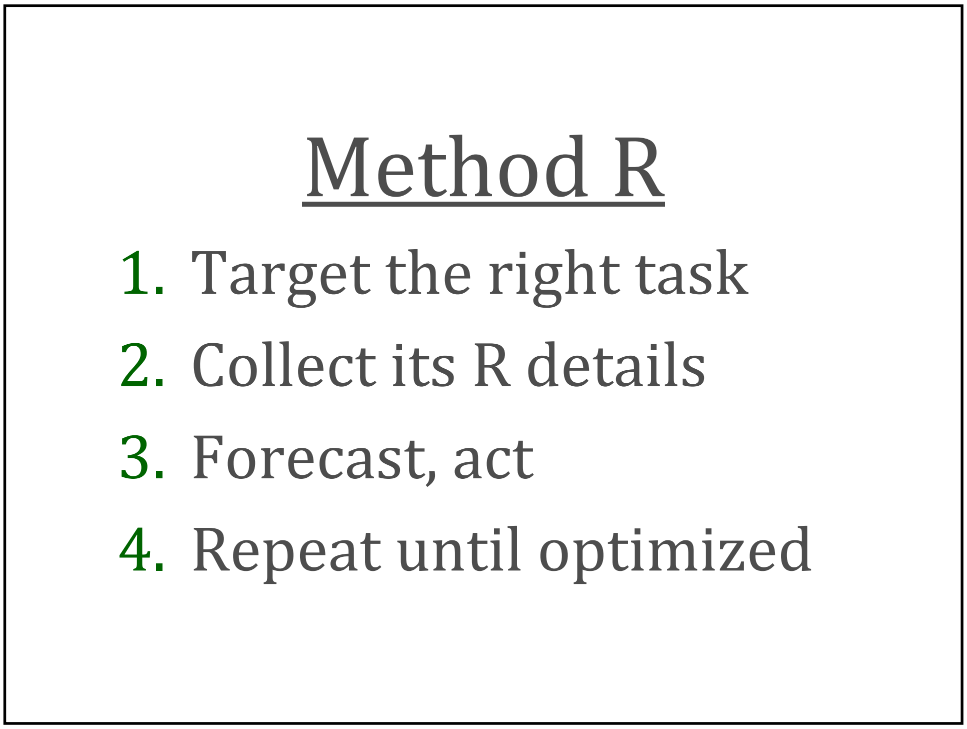


METHOD R™



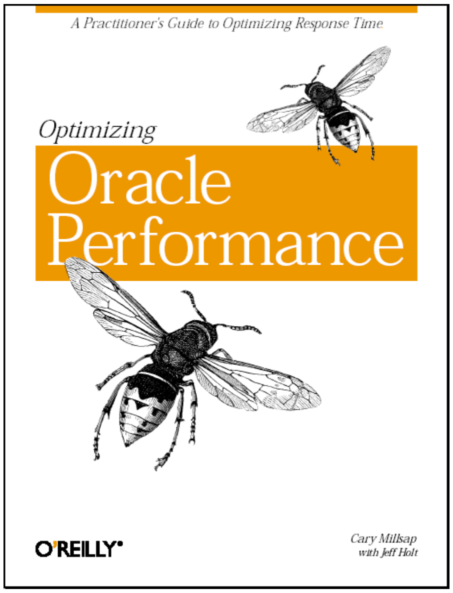
0

The Method



Method R

1. Target the right task
2. Collect its R details
3. Forecast, act
4. Repeat until optimized



1

“Idle” events

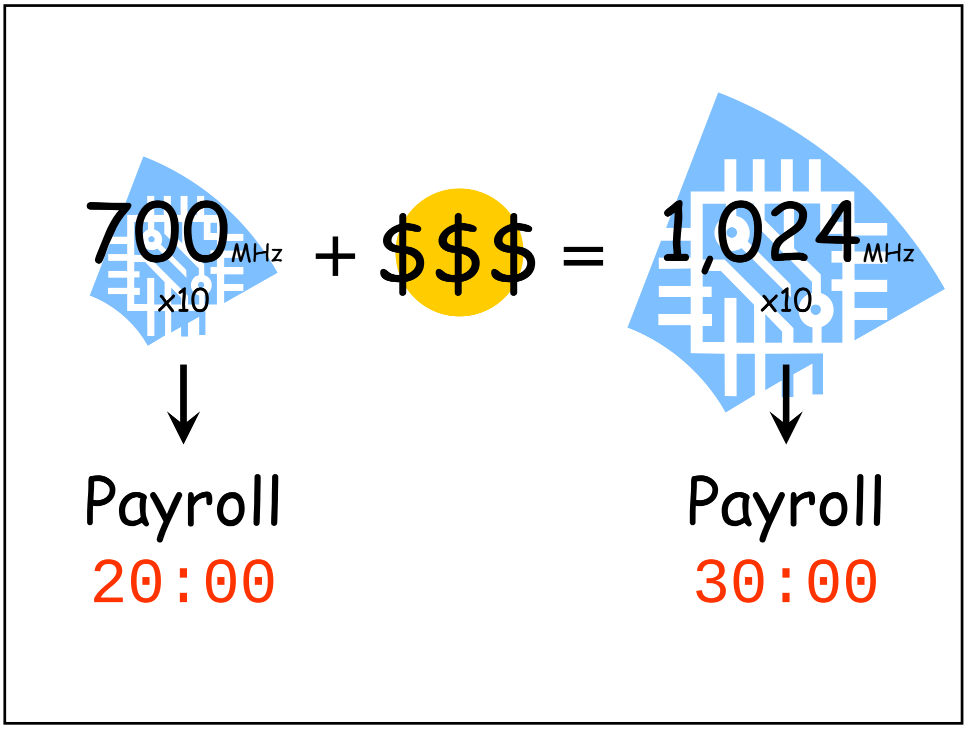
700 MHz

×10

↓

Payroll

20:00

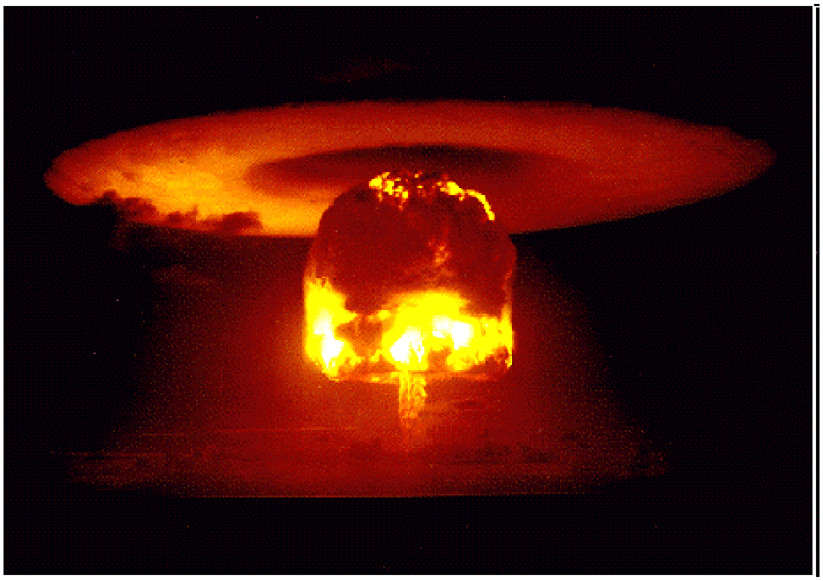


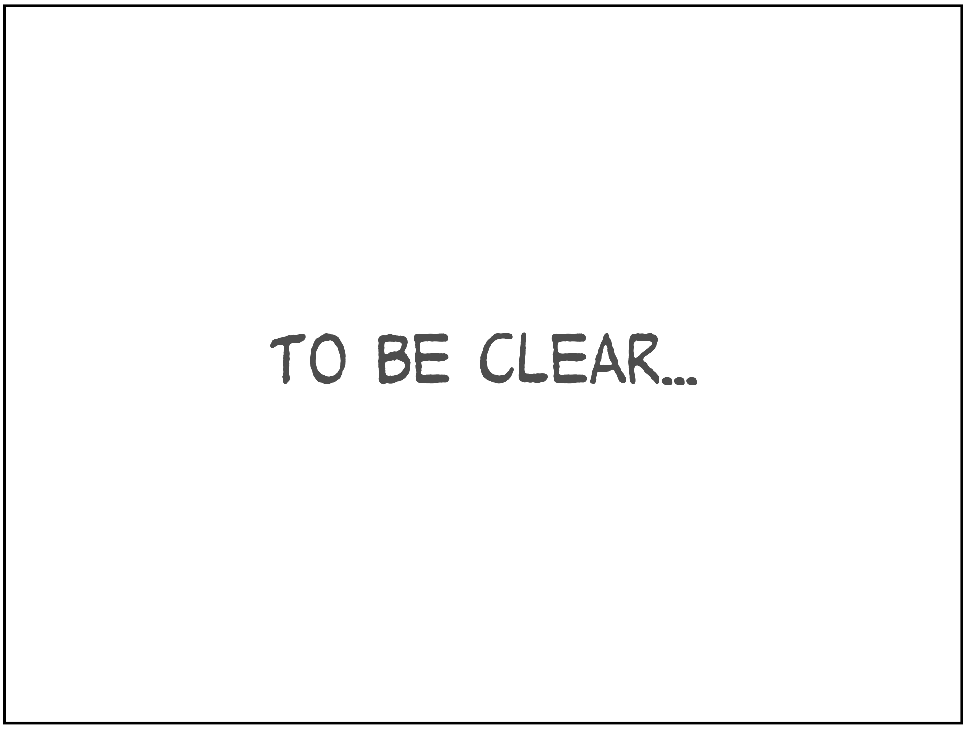
700 MHz ×10 + $$$ = 1,024 MHz ×10

↓ ↓

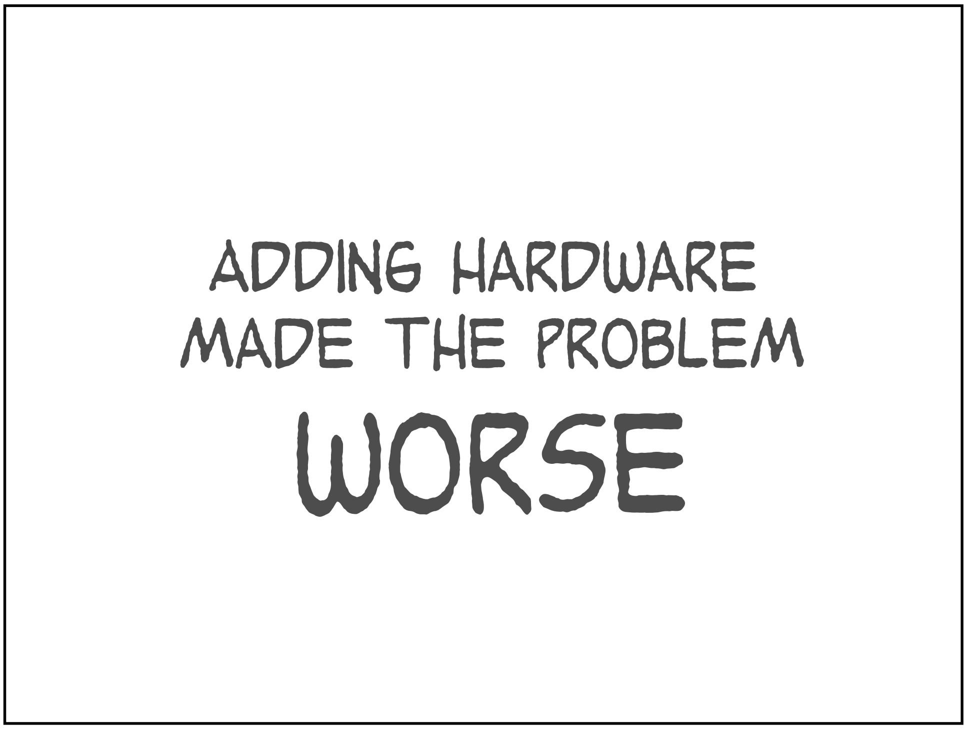
Payroll 20:00

Payroll 30:00



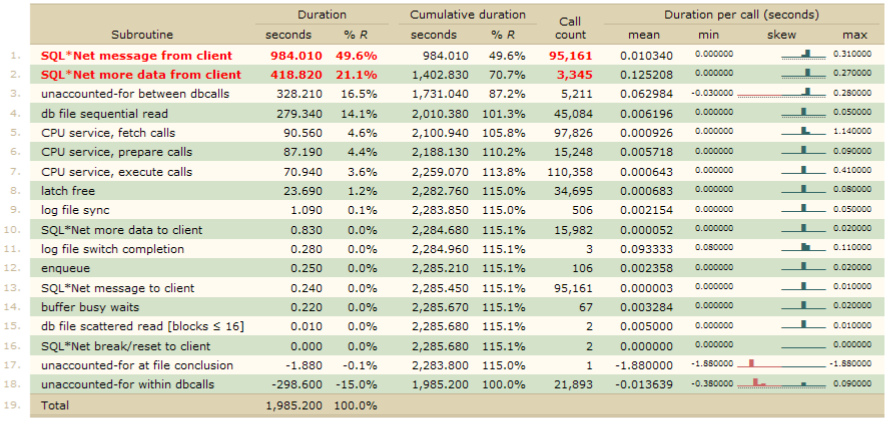


To be clear...

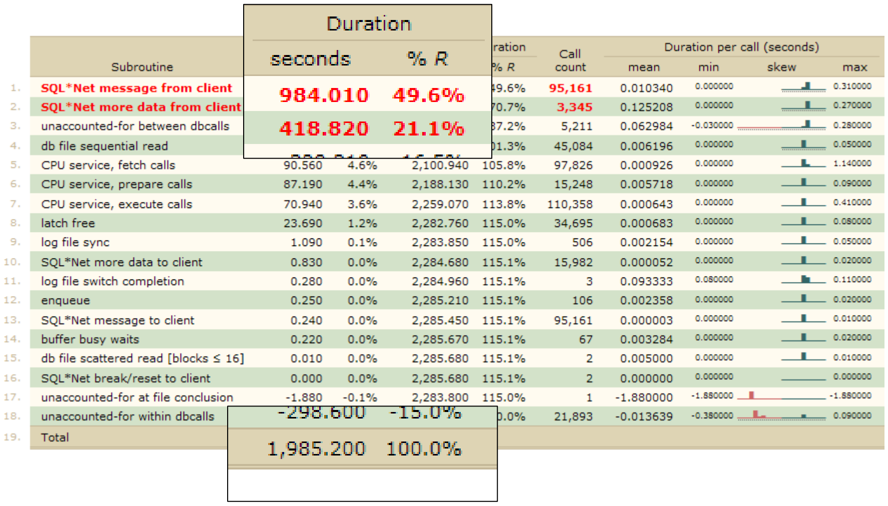


Adding hardware made the problem

WORSE



	Subroutine	Duration		Cumulative duration		Call	Duration per call (seconds)			
		seconds	% *R*	seconds	% *R*	count	mean	min	skew	max
1.	**SQL*Net message from client**	**984.010**	**49.6%**	984.010	49.6%	**95,161**	0.010340	0.000000		0.310000
2.	**SQL*Net more data from client**	**418.820**	**21.1%**	1,402.830	70.7%	**3,345**	0.125208	0.000000		0.270000
3.	unaccounted-for between dbcalls	328.210	16.5%	1,731.040	87.2%	5,211	0.062984	-0.030000		0.280000
4.	db file sequential read	279.340	14.1%	2,010.380	101.3%	45,084	0.006196	0.000000		0.050000
5.	CPU service, fetch calls	90.560	4.6%	2,100.940	105.8%	97,826	0.000926	0.000000		1.140000
6.	CPU service, prepare calls	87.190	4.4%	2,188.130	110.2%	15,248	0.005718	0.000000		0.090000
7.	CPU service, execute calls	70.940	3.6%	2,259.070	113.8%	110,358	0.000643	0.000000		0.410000
8.	latch free	23.690	1.2%	2,282.760	115.0%	34,695	0.000683	0.000000		0.080000
9.	log file sync	1.090	0.1%	2,283.850	115.0%	506	0.002154	0.000000		0.050000
10.	SQL*Net more data to client	0.830	0.0%	2,284.680	115.1%	15,982	0.000052	0.000000		0.020000
11.	log file switch completion	0.280	0.0%	2,284.960	115.1%	3	0.093333	0.080000		0.110000
12.	enqueue	0.250	0.0%	2,285.210	115.1%	106	0.002358	0.000000		0.020000
13.	SQL*Net message to client	0.240	0.0%	2,285.450	115.1%	95,161	0.000003	0.000000		0.010000
14.	buffer busy waits	0.220	0.0%	2,285.670	115.1%	67	0.003284	0.000000		0.020000
15.	db file scattered read [blocks ≤ 16]	0.010	0.0%	2,285.680	115.1%	2	0.005000	0.000000		0.010000
16.	SQL*Net break/reset to client	0.000	0.0%	2,285.680	115.1%	2	0.000000	0.000000		0.000000
17.	unaccounted-for at file conclusion	-1.880	-0.1%	2,283.800	115.0%	1	-1.880000	-1.880000		-1.880000
18.	unaccounted-for within dbcalls	-298.600	-15.0%	1,985.200	100.0%	21,893	-0.013639	-0.380000		0.090000
19.	Total	1,985.200	100.0%							



Duration

seconds % *R*

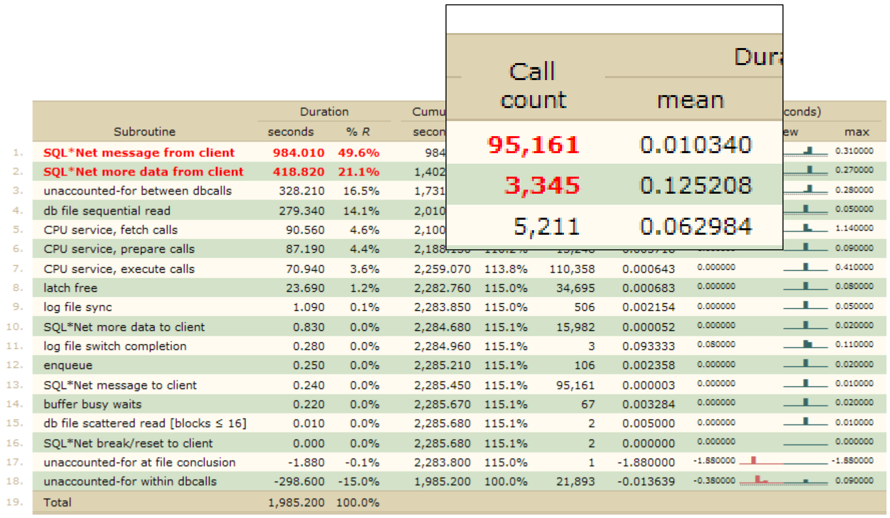
984.010 49.6%

418.820 21.1%

-298.600 -15.0%

1,985.200 100.0%

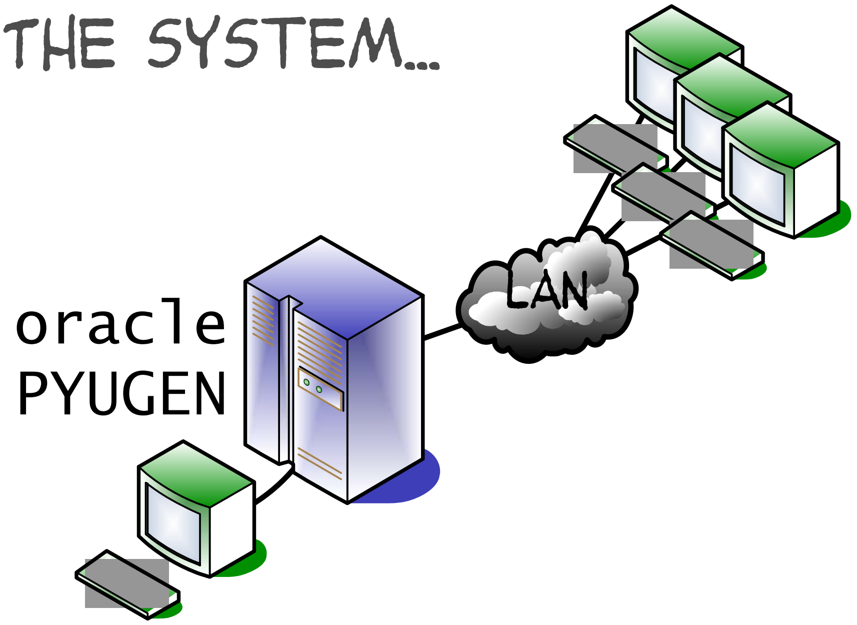
	Subroutine	Duration		Cumulative duration		Call	Duration per call (seconds)			
		seconds	% *R*	seconds	% *R*	count	mean	min	skew	max
1.	**SQL*Net message from client**	**984.010**	**49.6%**	984.010	49.6%	**95,161**	0.010340	0.000000		0.310000
2.	**SQL*Net more data from client**	**418.820**	**21.1%**	1,402.830	70.7%	**3,345**	0.125208	0.000000		0.270000
3.	unaccounted-for between dbcalls	328.210	16.5%	1,731.040	87.2%	5,211	0.062984	-0.030000		0.280000
4.	db file sequential read	279.340	14.1%	2,010.380	101.3%	45,084	0.006196	0.000000		0.050000
5.	CPU service, fetch calls	90.560	4.6%	2,100.940	105.8%	97,826	0.000926	0.000000		1.140000
6.	CPU service, prepare calls	87.190	4.4%	2,188.130	110.2%	15,248	0.005718	0.000000		0.090000
7.	CPU service, execute calls	70.940	3.6%	2,259.070	113.8%	110,358	0.000643	0.000000		0.410000
8.	latch free	23.690	1.2%	2,282.760	115.0%	34,695	0.000683	0.000000		0.080000
9.	log file sync	1.090	0.1%	2,283.850	115.0%	506	0.002154	0.000000		0.050000
10.	SQL*Net more data to client	0.830	0.0%	2,284.680	115.1%	15,982	0.000052	0.000000		0.020000
11.	log file switch completion	0.280	0.0%	2,284.960	115.1%	3	0.093333	0.080000		0.110000
12.	enqueue	0.250	0.0%	2,285.210	115.1%	106	0.002358	0.000000		0.020000
13.	SQL*Net message to client	0.240	0.0%	2,285.450	115.1%	95,161	0.000003	0.000000		0.010000
14.	buffer busy waits	0.220	0.0%	2,285.670	115.1%	67	0.003284	0.000000		0.020000
15.	db file scattered read [blocks ≤ 16]	0.010	0.0%	2,285.680	115.1%	2	0.005000	0.000000		0.010000
16.	SQL*Net break/reset to client	0.000	0.0%	2,285.680	115.1%	2	0.000000	0.000000		0.000000
17.	unaccounted-for at file conclusion	-1.880	-0.1%	2,283.800	115.0%	1	-1.880000	-1.880000		-1.880000
18.	unaccounted-for within dbcalls				0.0%	21,893	-0.013639	-0.380000		0.090000
19.	Total									



		Duration		Cumu						conds)
	Subroutine	seconds	% *R*	secon					ew	max
1.	**SQL*Net message from client**	**984.010**	**49.6%**	984						0.310000
2.	**SQL*Net more data from client**	**418.820**	**21.1%**	1,402						0.270000
3.	unaccounted-for between dbcalls	328.210	16.5%	1,731						0.280000
4.	db file sequential read	279.340	14.1%	2,010						0.050000
5.	CPU service, fetch calls	90.560	4.6%	2,100						1.140000
6.	CPU service, prepare calls	87.190	4.4%	2,188						0.090000
7.	CPU service, execute calls	70.940	3.6%	2,259.070	113.8%	110,358	0.000643	0.000000		0.410000
8.	latch free	23.690	1.2%	2,282.760	115.0%	34,695	0.000683	0.000000		0.080000
9.	log file sync	1.090	0.1%	2,283.850	115.0%	506	0.002154	0.000000		0.050000
10.	SQL*Net more data to client	0.830	0.0%	2,284.680	115.1%	15,982	0.000052	0.000000		0.020000
11.	log file switch completion	0.280	0.0%	2,284.960	115.1%	3	0.093333	0.080000		0.110000
12.	enqueue	0.250	0.0%	2,285.210	115.1%	106	0.002358	0.000000		0.020000
13.	SQL*Net message to client	0.240	0.0%	2,285.450	115.1%	95,161	0.000003	0.000000		0.010000
14.	buffer busy waits	0.220	0.0%	2,285.670	115.1%	67	0.003284	0.000000		0.020000
15.	db file scattered read [blocks ≤ 16]	0.010	0.0%	2,285.680	115.1%	2	0.005000	0.000000		0.010000
16.	SQL*Net break/reset to client	0.000	0.0%	2,285.680	115.1%	2	0.000000	0.000000		0.000000
17.	unaccounted-for at file conclusion	-1.880	-0.1%	2,283.800	115.0%	1	-1.880000	-1.880000		-1.880000
18.	unaccounted-for within dbcalls	-298.600	-15.0%	1,985.200	100.0%	21,893	-0.013639	-0.380000		0.090000
19.	Total	1,985.200	100.0%							

Call count	mean
95,161	0.010340
3,345	0.125208
5,211	0.062984

THE SYSTEM...

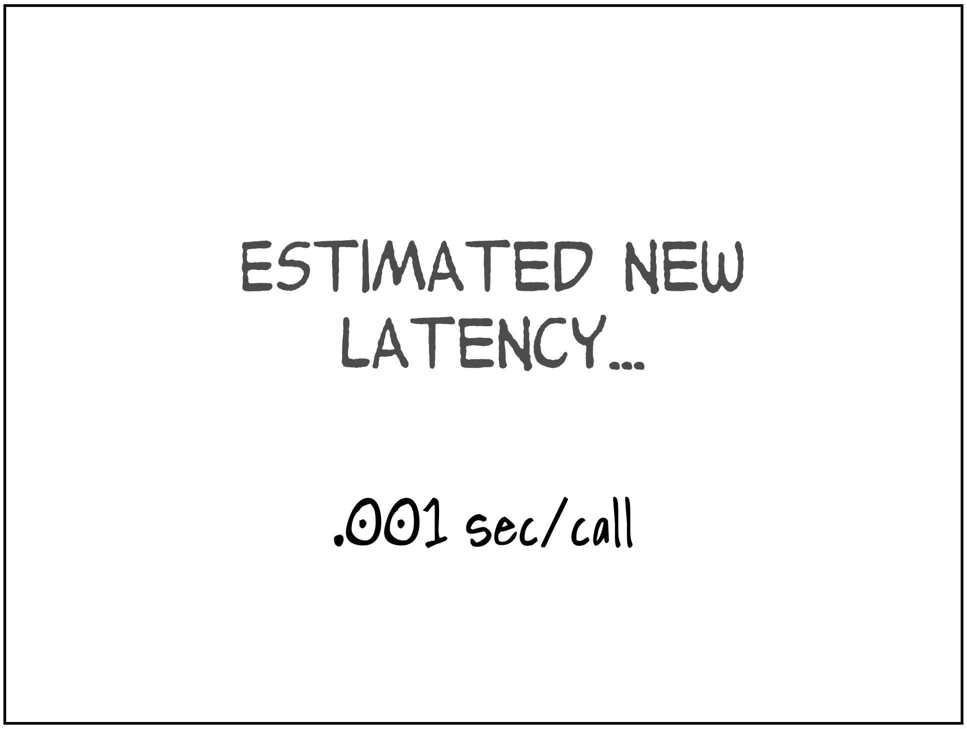


tnsnames.ora

```
oracle =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS =
        (PROTOCOL = TCP)
        (HOST = SYS1)
        (PORT = 1521)
      )
    )
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = oracle.xyz)
    )
  )
```

TEST…

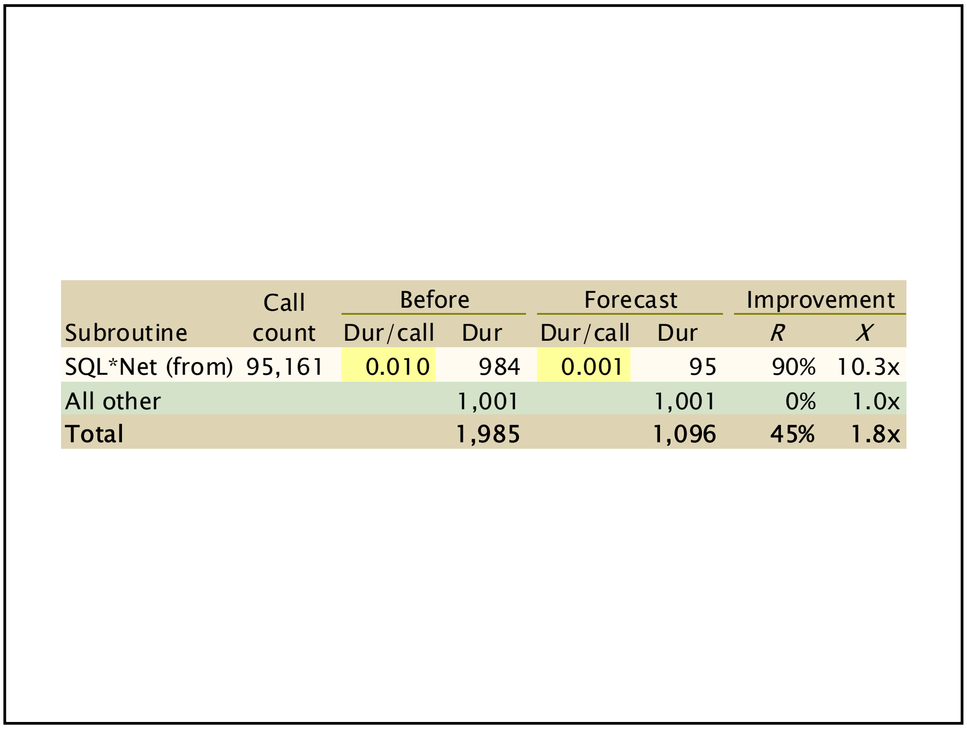
```
oracle_local =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS =
        (PROTOCOL = BEQ)
        (HOST = SYS1)
        (PORT = 1521)
      )
    )
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = oracle_local.xyz)
    )
  )
```



ESTIMATED NEW LATENCY...

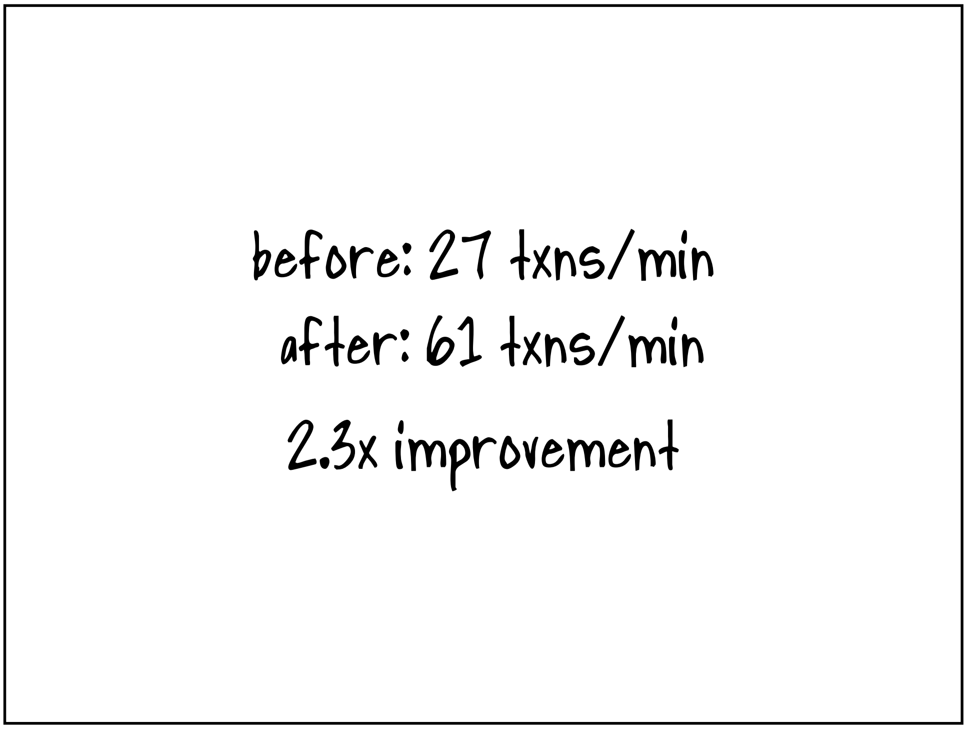
.001 sec/call

FORECAST...



Subroutine	Call count	Before		Forecast		Improvement	
		Dur/call	Dur	Dur/call	Dur	*R*	*X*
SQL*Net (from)	95,161	0.010	984	0.001	95	90%	10.3x
All other			1,001		1,001	0%	1.0x
Total			**1,985**		**1,096**	**45%**	**1.8x**

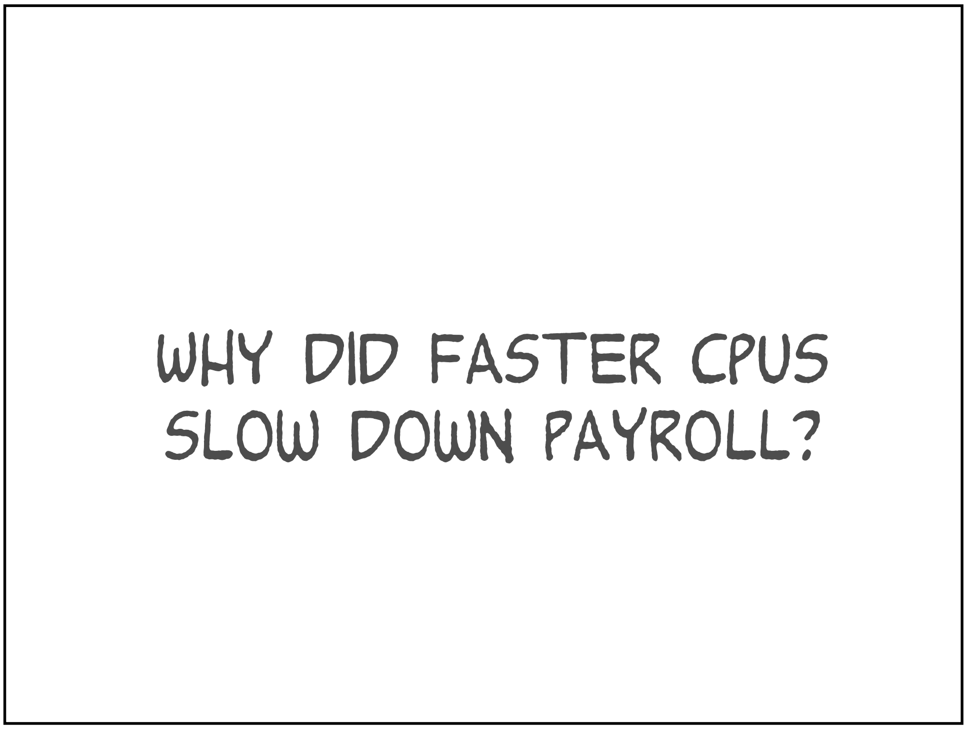
OUTCOME…



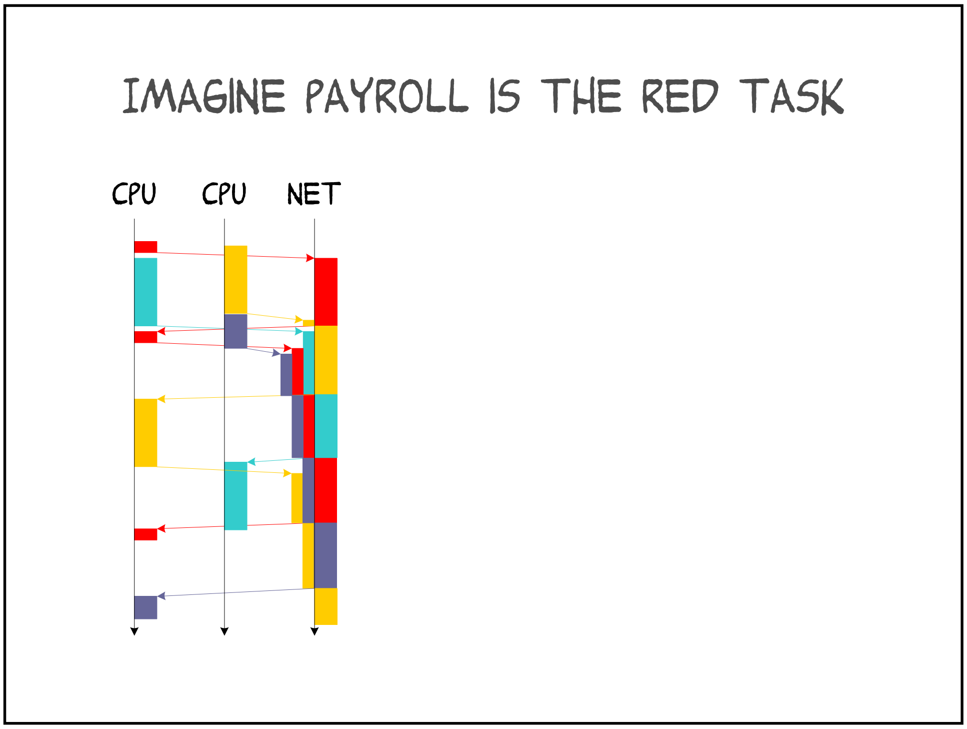
before: 27 txns/min

after: 61 txns/min

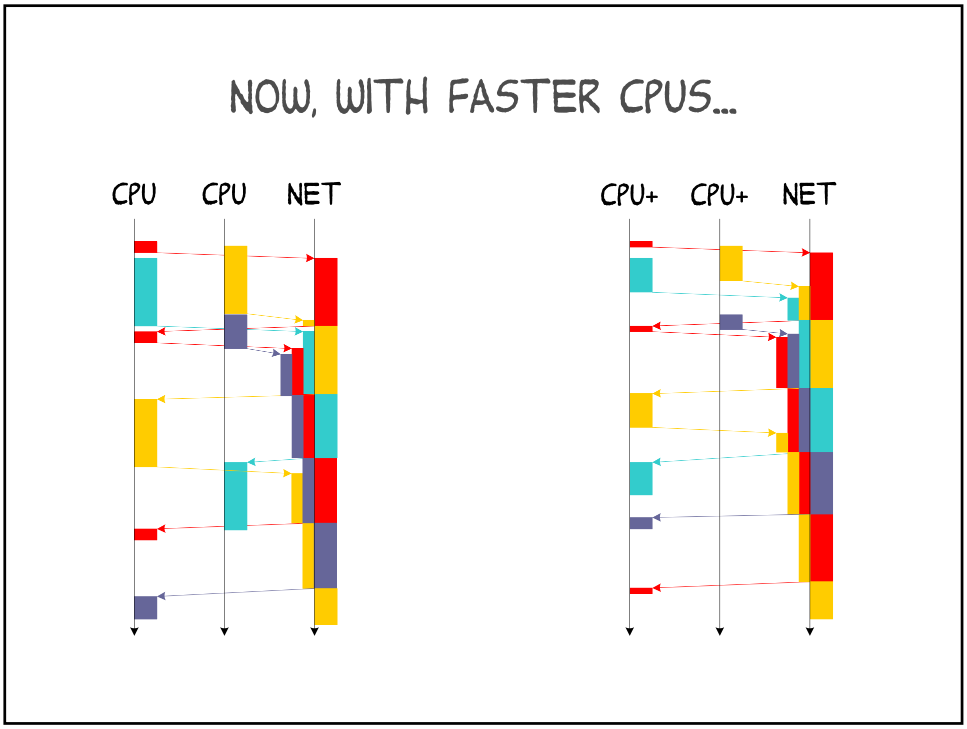
2.3x improvement



WHY DID FASTER CPUS SLOW DOWN PAYROLL?

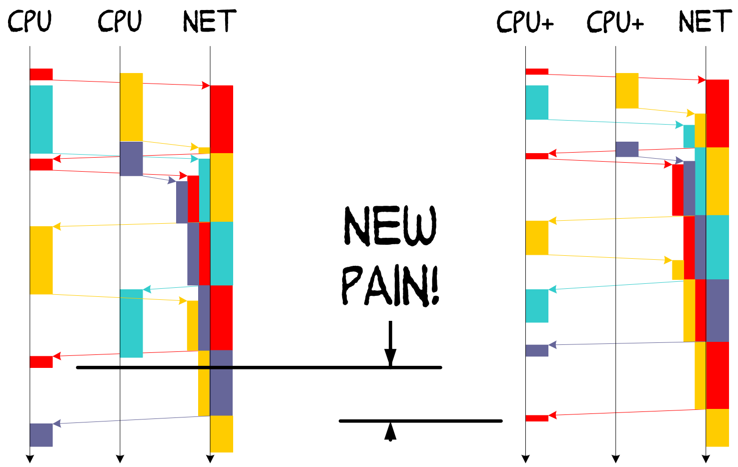


IMAGINE PAYROLL IS THE RED TASK

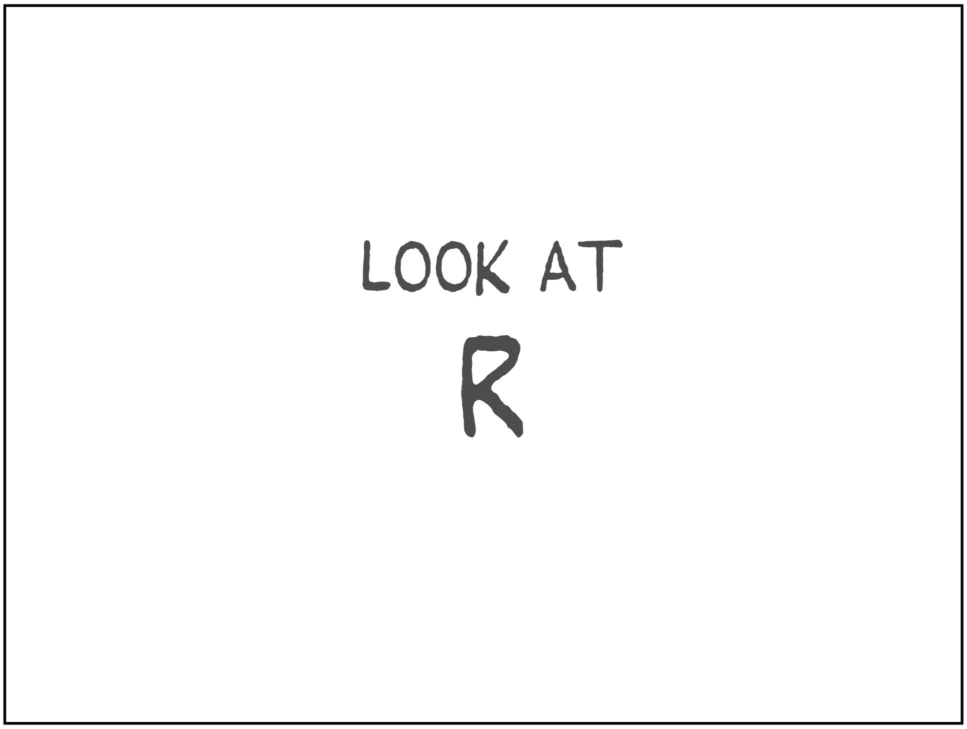


NOW, WITH FASTER CPUS...

NET EFFECT... PAYROLL IS *WORSE*



MORAL...



LOOK AT

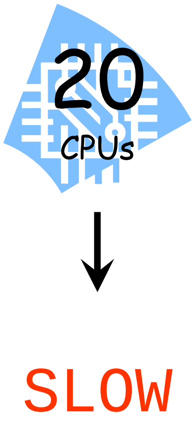
R

APPLY REMEDIES THAT WILL MAKE THE DESIRED DIFFERENCE

Questions?

2

Killing me softly

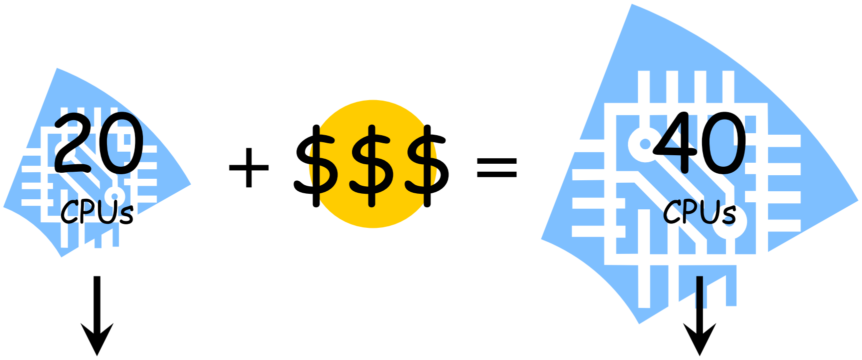


20

CPUs

↓

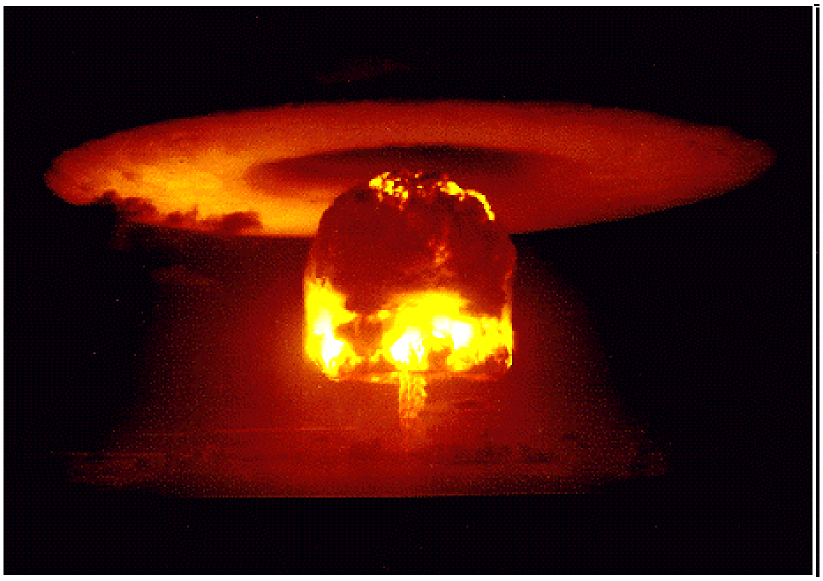
SLOW

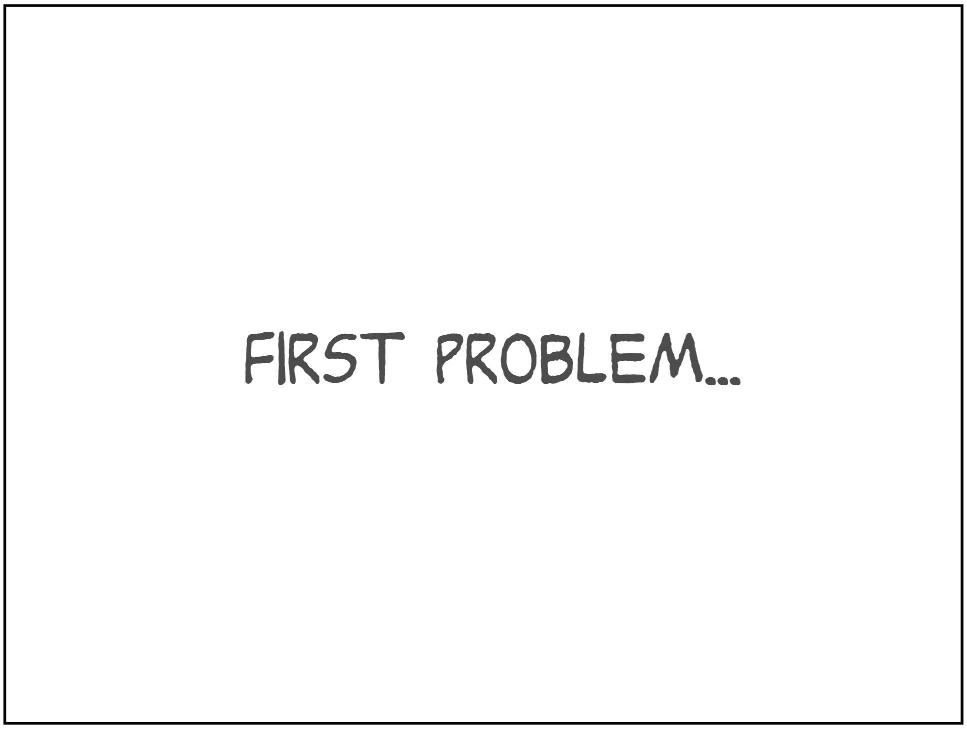


20 CPUs + $$$ = 40 CPUs

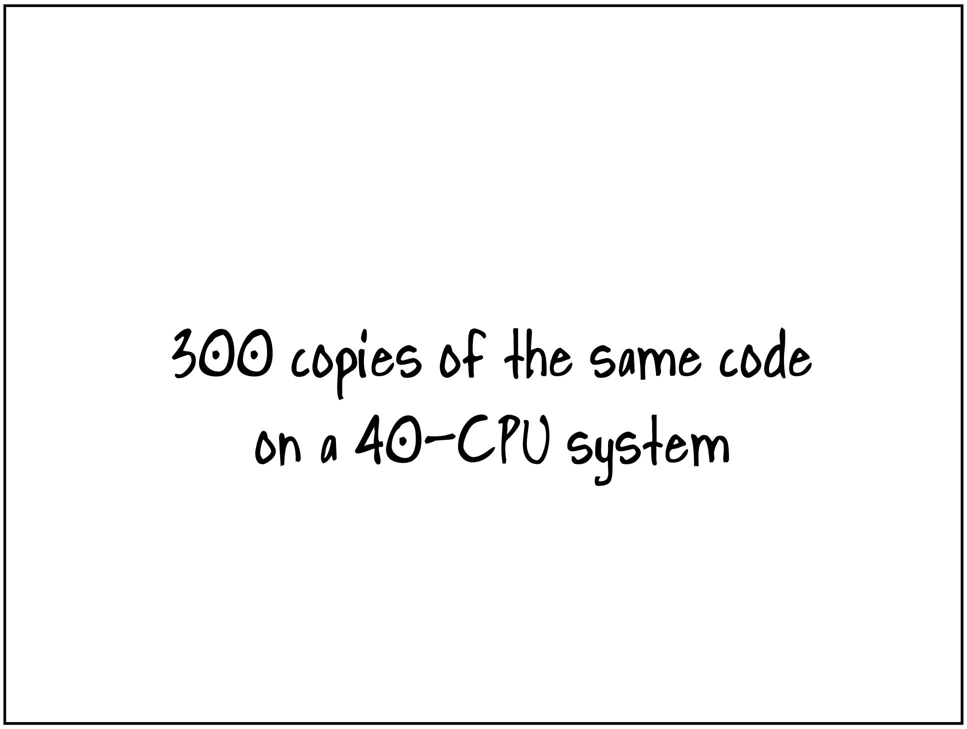
SLOW

STILL SLOW



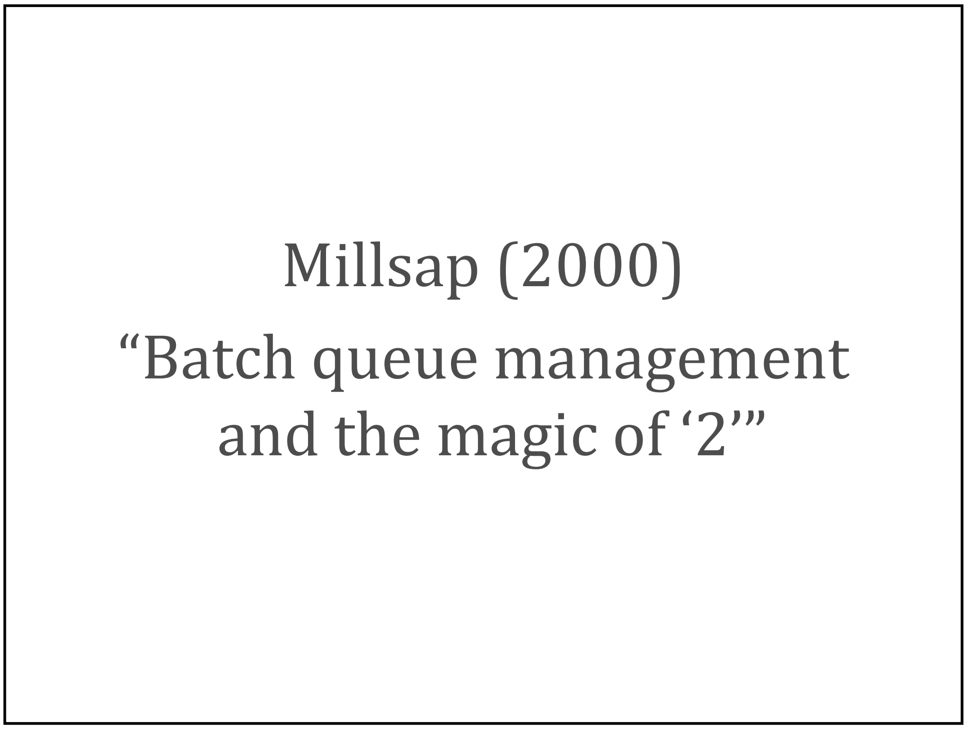


FIRST PROBLEM...



300 copies of the same code
on a 40-CPU system

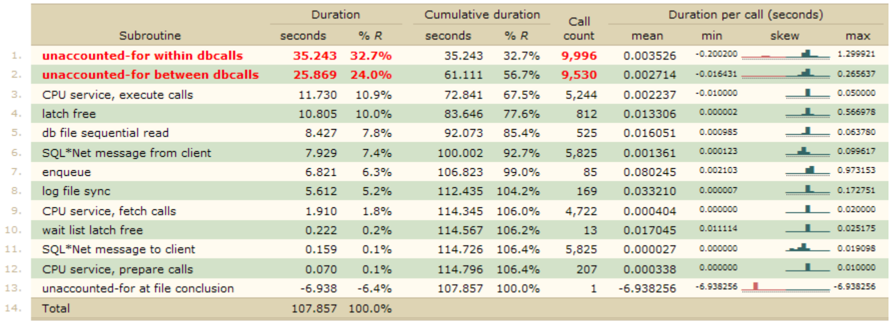
THAT'S TOO MANY CONCURRENT BATCH JOBS



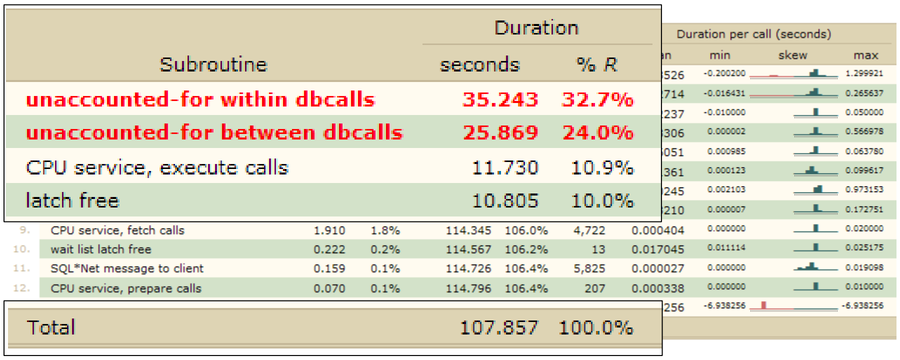
Millsap (2000)

“Batch queue management and the magic of ‘2’”

HERE'S THE PROBLEM IT CAUSES



		Duration		Cumulative duration		Call	Duration per call (seconds)			
	Subroutine	seconds	% *R*	seconds	% *R*	count	mean	min	skew	max
1.	**unaccounted-for within dbcalls**	**35.243**	**32.7%**	35.243	32.7%	**9,996**	0.003526	-0.200200		1.299921
2.	**unaccounted-for between dbcalls**	**25.869**	**24.0%**	61.111	56.7%	**9,530**	0.002714	-0.016431		0.265637
3.	CPU service, execute calls	11.730	10.9%	72.841	67.5%	5,244	0.002237	-0.010000		0.050000
4.	latch free	10.805	10.0%	83.646	77.6%	812	0.013306	0.000002		0.566978
5.	db file sequential read	8.427	7.8%	92.073	85.4%	525	0.016051	0.000985		0.063780
6.	SQL*Net message from client	7.929	7.4%	100.002	92.7%	5,825	0.001361	0.000123		0.099617
7.	enqueue	6.821	6.3%	106.823	99.0%	85	0.080245	0.002103		0.973153
8.	log file sync	5.612	5.2%	112.435	104.2%	169	0.033210	0.000007		0.172751
9.	CPU service, fetch calls	1.910	1.8%	114.345	106.0%	4,722	0.000404	0.000000		0.020000
10.	wait list latch free	0.222	0.2%	114.567	106.2%	13	0.017045	0.011114		0.025175
11.	SQL*Net message to client	0.159	0.1%	114.726	106.4%	5,825	0.000027	0.000000		0.019098
12.	CPU service, prepare calls	0.070	0.1%	114.796	106.4%	207	0.000338	0.000000		0.010000
13.	unaccounted-for at file conclusion	-6.938	-6.4%	107.857	100.0%	1	-6.938256	-6.938256		-6.938256
14.	Total	107.857	100.0%							

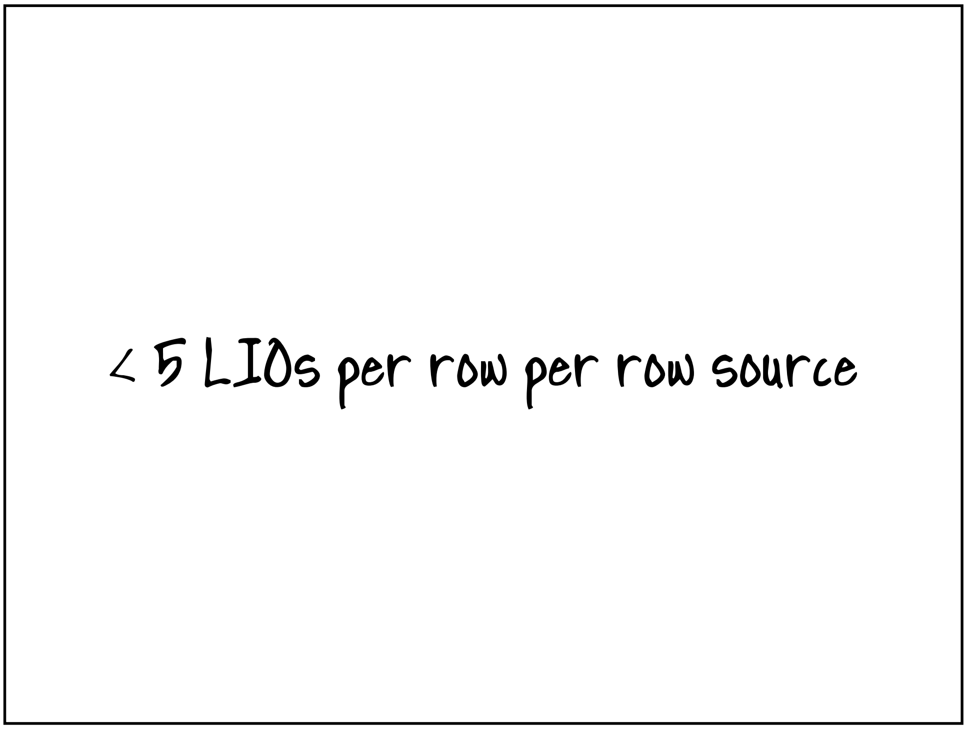


Subroutine	Duration seconds	 % *R*
unaccounted-for within dbcalls	**35.243**	**32.7%**
unaccounted-for between dbcalls	**25.869**	**24.0%**
CPU service, execute calls	11.730	10.9%
latch free	10.805	10.0%
Total	107.857	100.0%

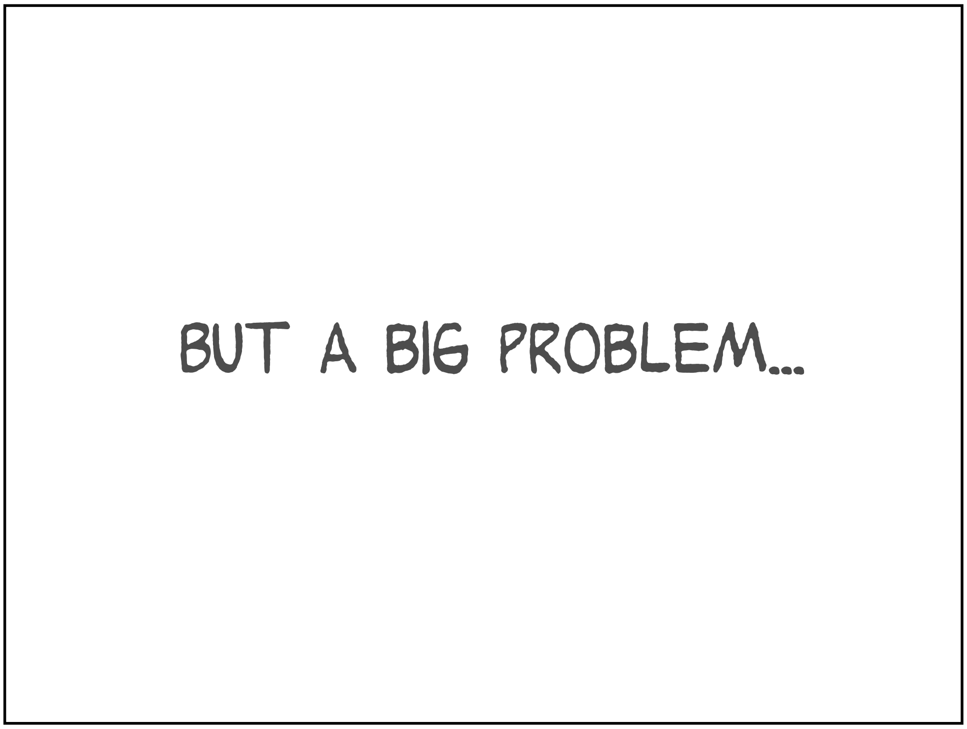
							Duration per call (seconds)			
							an	min	skew	max
							526	-0.200200		1.299921
							714	-0.016431		0.265637
							237	-0.010000		0.050000
							306	0.000002		0.566978
							051	0.000985		0.063780
							361	0.000123		0.099617
							245	0.002103		0.973153
							210	0.000007		0.172751
9.	CPU service, fetch calls	1.910	1.8%	114.345	106.0%	4,722	0.000404	0.000000		0.020000
10.	wait list latch free	0.222	0.2%	114.567	106.2%	13	0.017045	0.011114		0.025175
11.	SQL*Net message to client	0.159	0.1%	114.726	106.4%	5,825	0.000027	0.000000		0.019098
12.	CPU service, prepare calls	0.070	0.1%	114.796	106.4%	207	0.000338	0.000000		0.010000
							256	-6.938256		-6.938256

THIS IS WHAT COMPETITION FOR CPU CAPACITY LOOKS LIKE

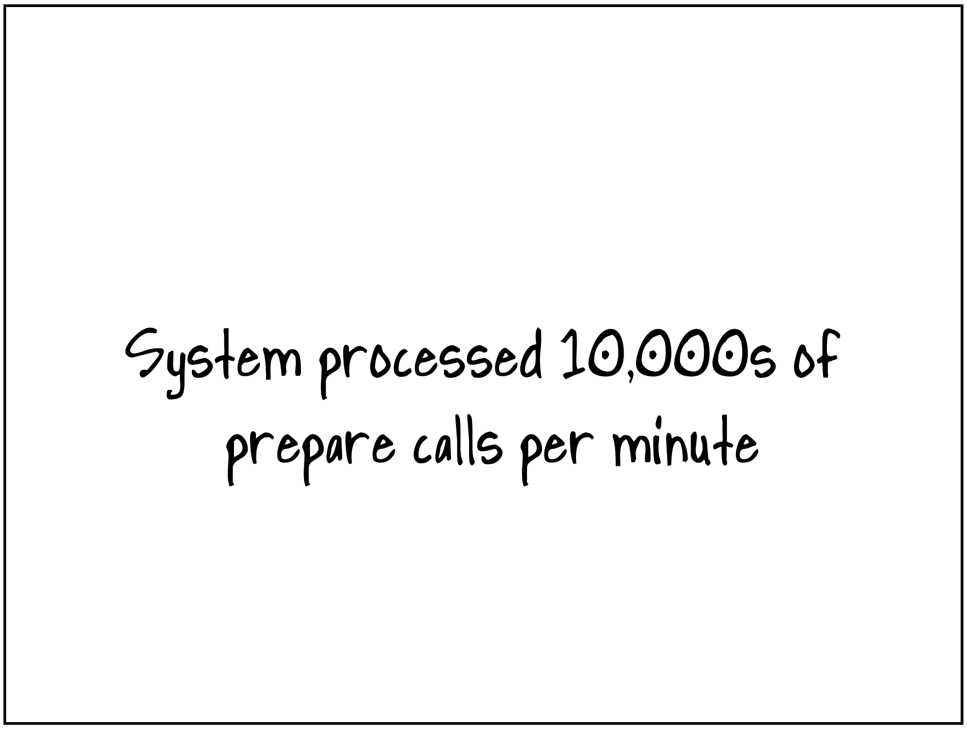
SQL EFFICIENCY WAS
NOT A PROBLEM...



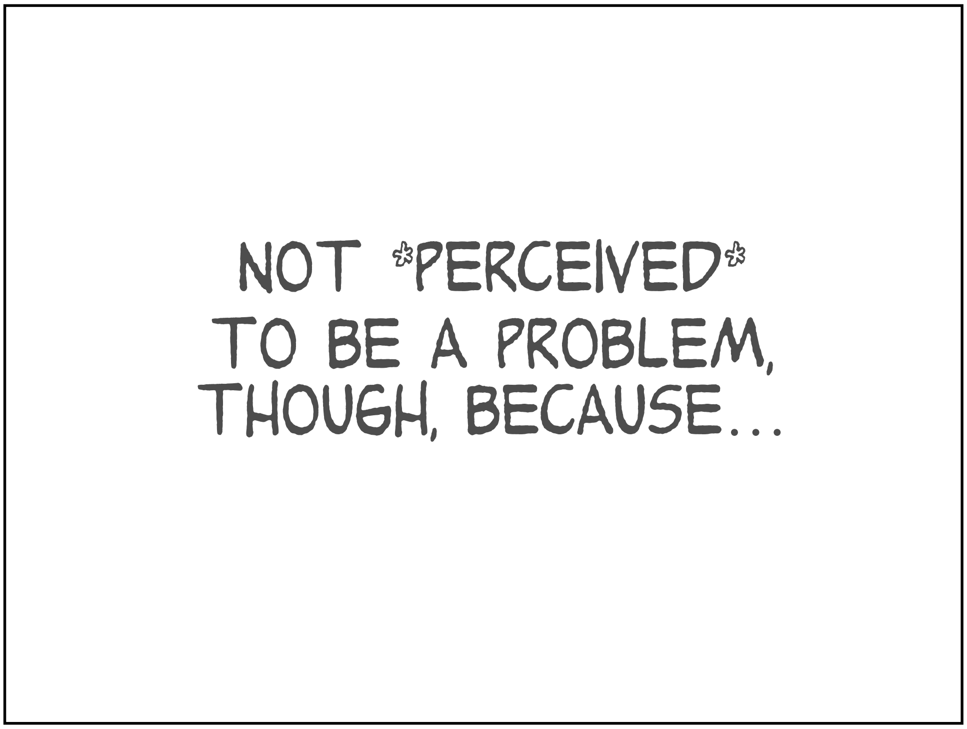
< 5 LIOs per row per row source



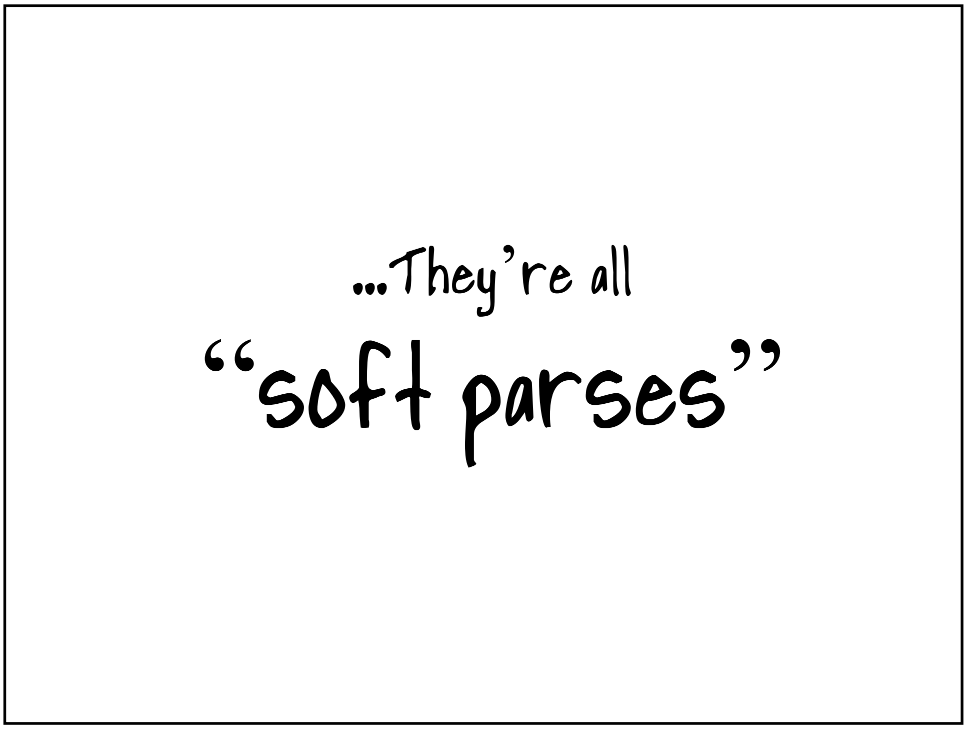
BUT A BIG PROBLEM...



System processed 10,000s of
prepare calls per minute



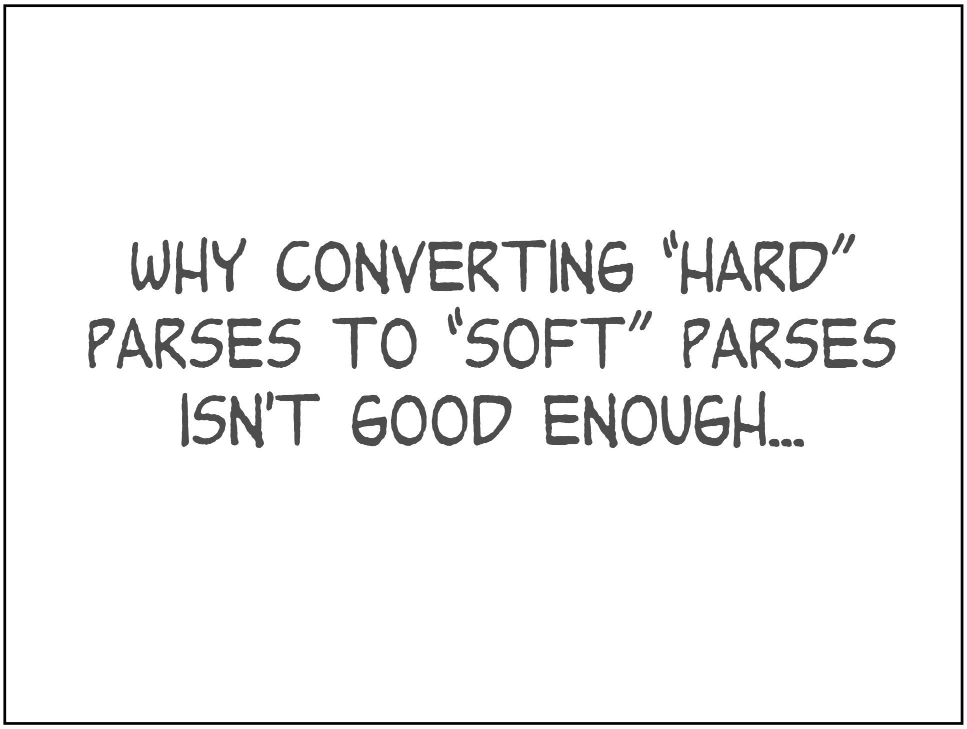
NOT *PERCEIVED*
TO BE A PROBLEM,
THOUGH, BECAUSE...



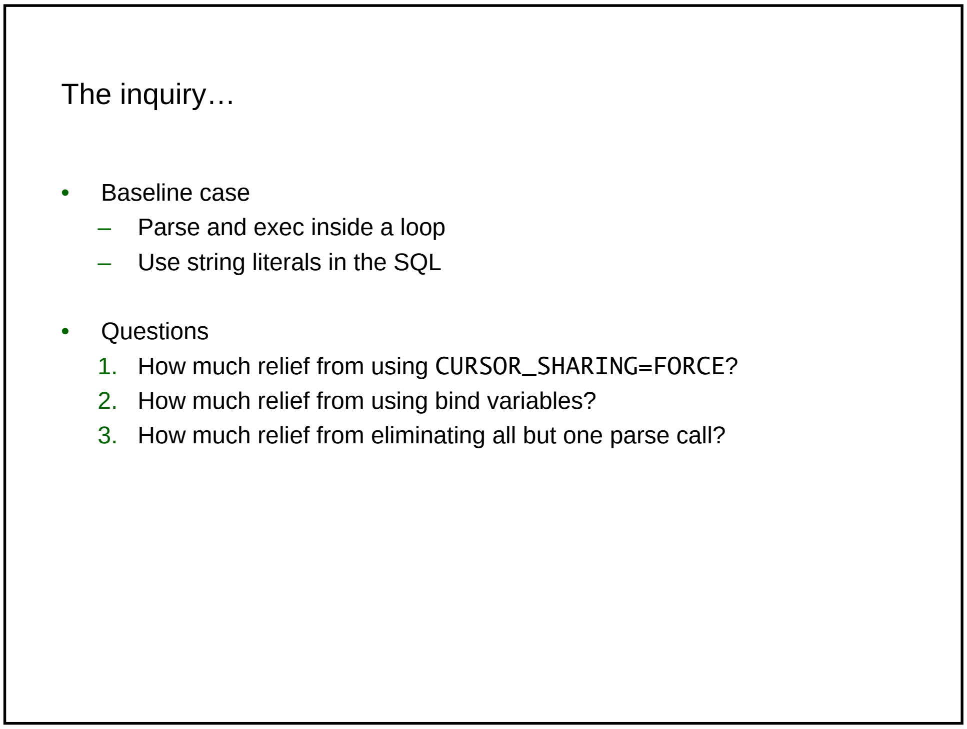
...They're all

"soft parses"

AARGH!

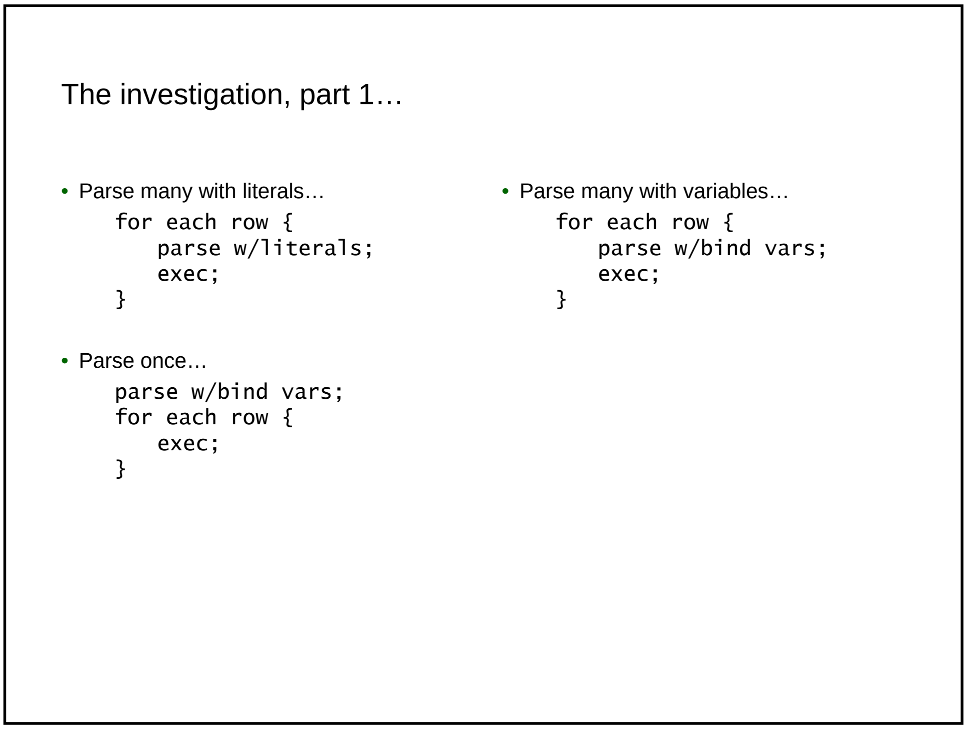


WHY CONVERTING "HARD" PARSES TO "SOFT" PARSES ISN'T GOOD ENOUGH...



The inquiry…

- Baseline case
  - Parse and exec inside a loop
  - Use string literals in the SQL
- Questions
  1. How much relief from using `CURSOR_SHARING=FORCE`?
  2. How much relief from using bind variables?
  3. How much relief from eliminating all but one parse call?



The investigation, part 1…

- Parse many with literals…

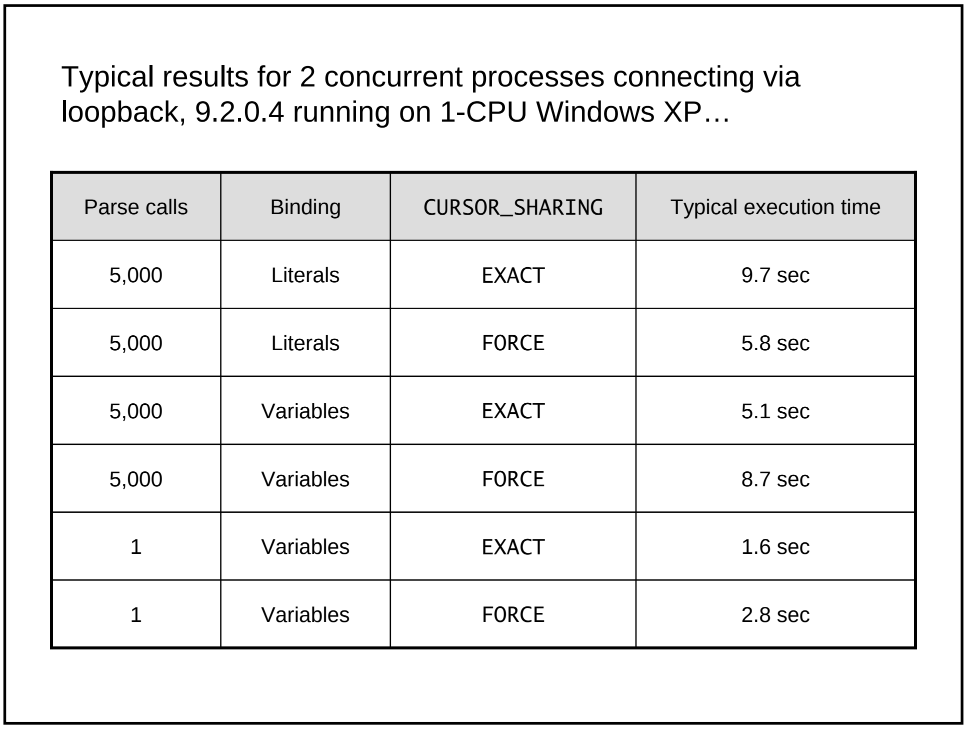
```
for each row {
    parse w/literals;
    exec;
}
```

- Parse many with variables…

```
for each row {
    parse w/bind vars;
    exec;
}
```

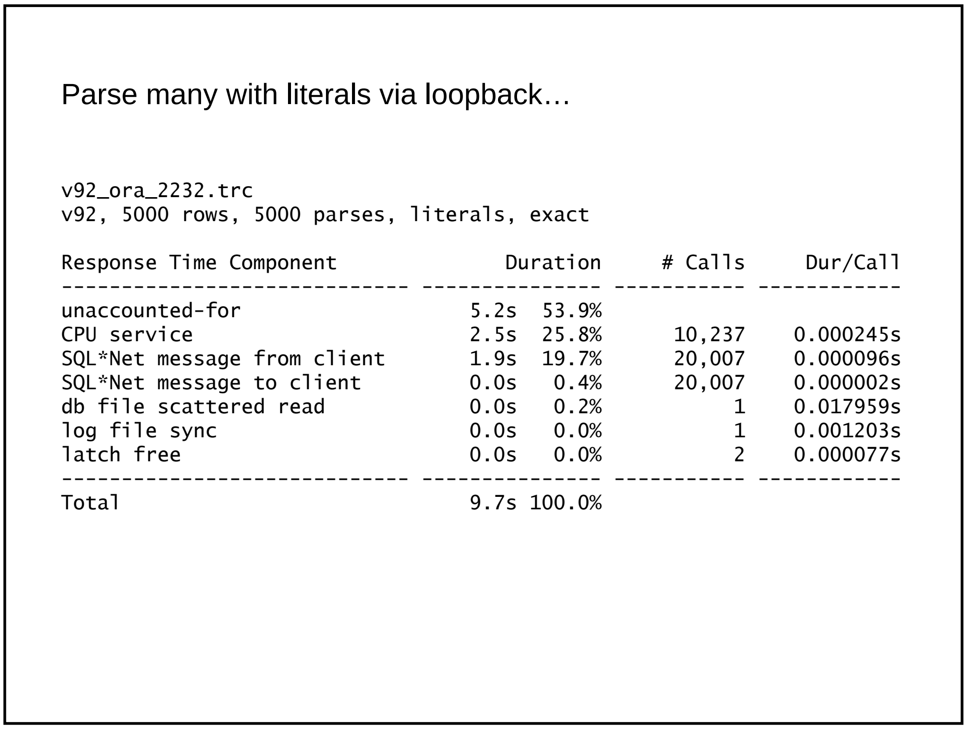
- Parse once…

```
parse w/bind vars;
for each row {
    exec;
}
```



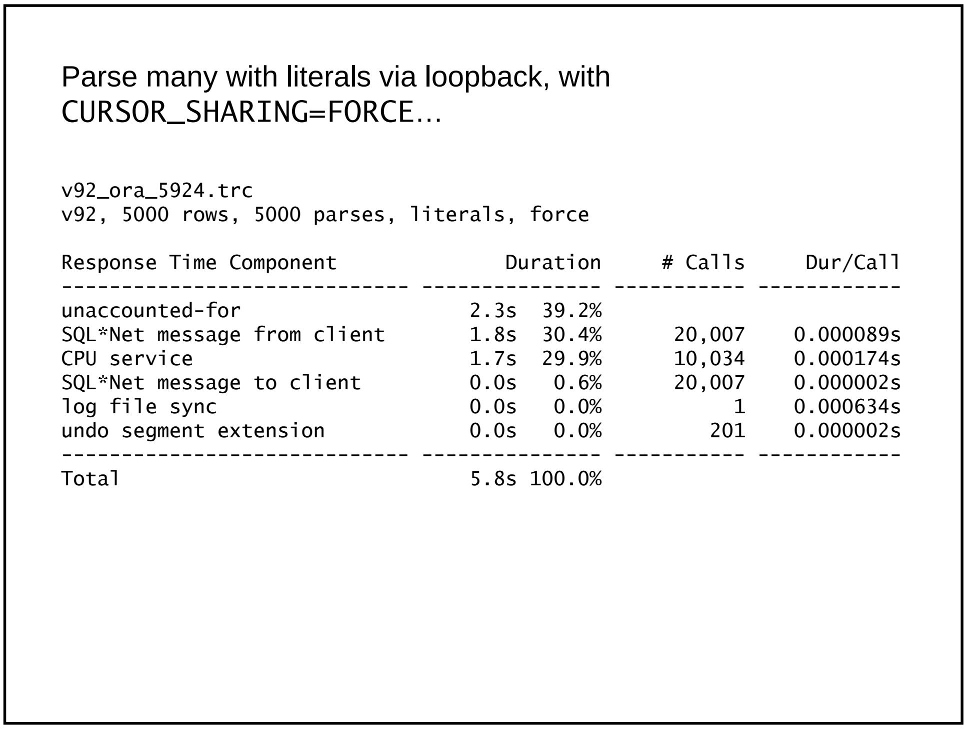
Typical results for 2 concurrent processes connecting via loopback, 9.2.0.4 running on 1-CPU Windows XP…

Parse calls	Binding	CURSOR_SHARING	Typical execution time
5,000	Literals	EXACT	9.7 sec
5,000	Literals	FORCE	5.8 sec
5,000	Variables	EXACT	5.1 sec
5,000	Variables	FORCE	8.7 sec
1	Variables	EXACT	1.6 sec
1	Variables	FORCE	2.8 sec



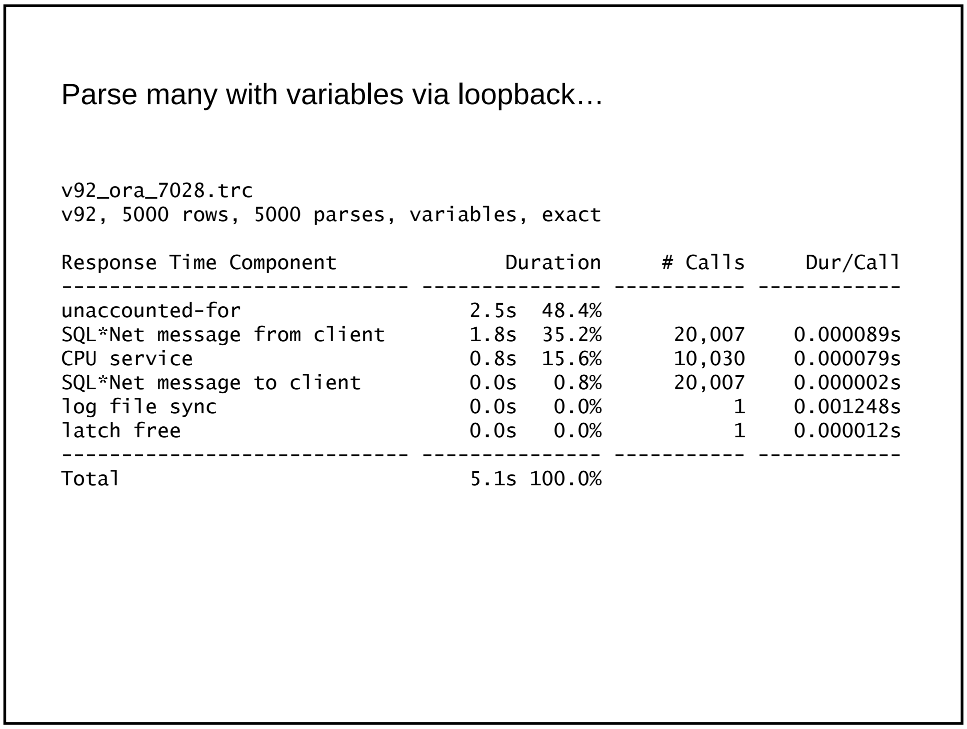
Parse many with literals via loopback…

```
v92_ora_2232.trc
v92, 5000 rows, 5000 parses, literals, exact

Response Time Component              Duration     # Calls     Dur/Call
----------------------------- --------------- ----------- ------------
unaccounted-for                  5.2s  53.9%
CPU service                      2.5s  25.8%      10,237    0.000245s
SQL*Net message from client      1.9s  19.7%      20,007    0.000096s
SQL*Net message to client        0.0s   0.4%      20,007    0.000002s
db file scattered read           0.0s   0.2%           1    0.017959s
log file sync                    0.0s   0.0%           1    0.001203s
latch free                       0.0s   0.0%           2    0.000077s
----------------------------- --------------- ----------- ------------
Total                            9.7s 100.0%
```



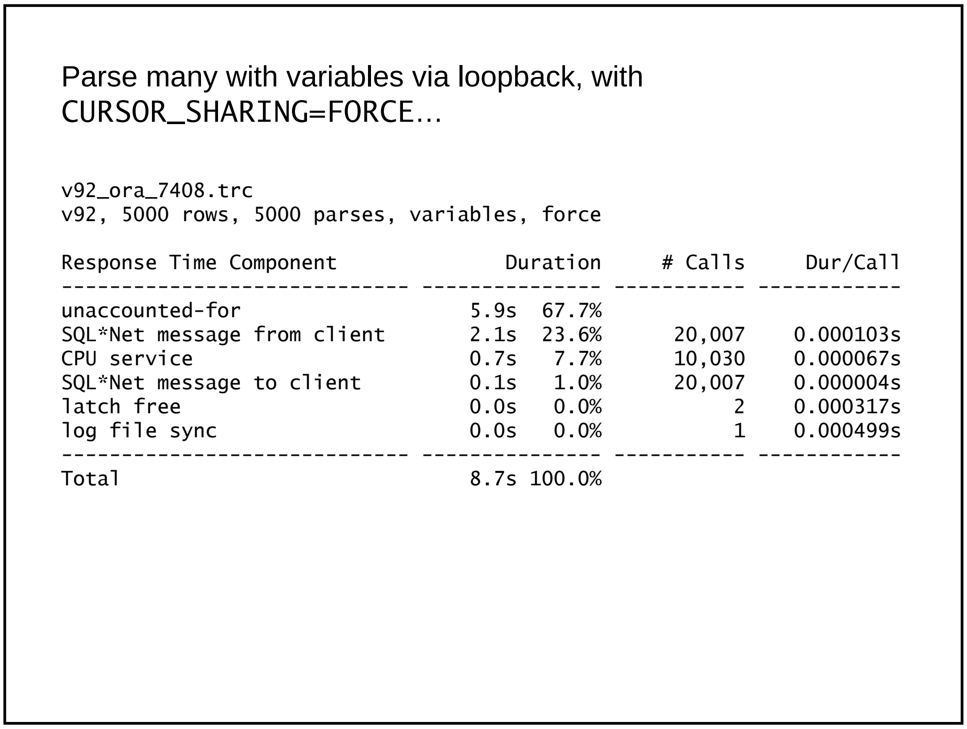
Parse many with literals via loopback, with `CURSOR_SHARING=FORCE`…

```
v92_ora_5924.trc
v92, 5000 rows, 5000 parses, literals, force

Response Time Component              Duration     # Calls     Dur/Call
----------------------------- --------------- ----------- ------------
unaccounted-for                   2.3s  39.2%
SQL*Net message from client       1.8s  30.4%      20,007    0.000089s
CPU service                       1.7s  29.9%      10,034    0.000174s
SQL*Net message to client         0.0s   0.6%      20,007    0.000002s
log file sync                     0.0s   0.0%           1    0.000634s
undo segment extension            0.0s   0.0%         201    0.000002s
----------------------------- --------------- ----------- ------------
Total                             5.8s 100.0%
```



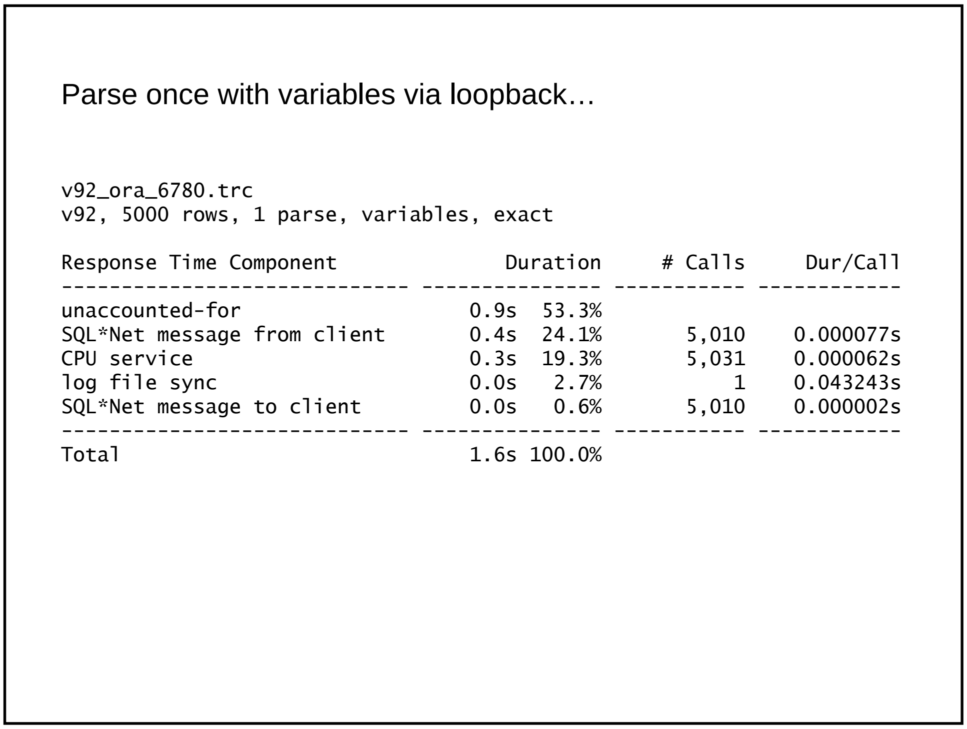
Parse many with variables via loopback…

```
v92_ora_7028.trc
v92, 5000 rows, 5000 parses, variables, exact

Response Time Component              Duration     # Calls     Dur/Call
----------------------------- --------------- ----------- ------------
unaccounted-for                   2.5s  48.4%
SQL*Net message from client       1.8s  35.2%      20,007    0.000089s
CPU service                       0.8s  15.6%      10,030    0.000079s
SQL*Net message to client         0.0s   0.8%      20,007    0.000002s
log file sync                     0.0s   0.0%           1    0.001248s
latch free                        0.0s   0.0%           1    0.000012s
----------------------------- --------------- ----------- ------------
Total                             5.1s 100.0%
```



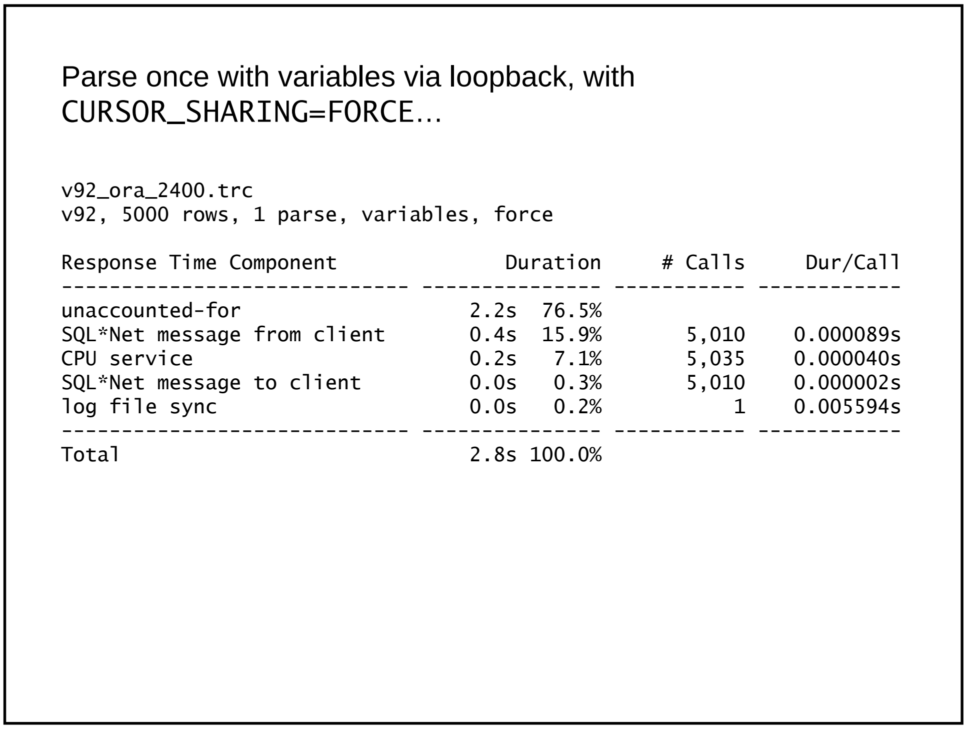
Parse many with variables via loopback, with `CURSOR_SHARING=FORCE`…

```
v92_ora_7408.trc
v92, 5000 rows, 5000 parses, variables, force

Response Time Component               Duration     # Calls     Dur/Call
----------------------------- --------------- ----------- ------------
unaccounted-for                   5.9s  67.7%
SQL*Net message from client       2.1s  23.6%      20,007    0.000103s
CPU service                       0.7s   7.7%      10,030    0.000067s
SQL*Net message to client         0.1s   1.0%      20,007    0.000004s
latch free                        0.0s   0.0%           2    0.000317s
log file sync                     0.0s   0.0%           1    0.000499s
----------------------------- --------------- ----------- ------------
Total                             8.7s 100.0%
```



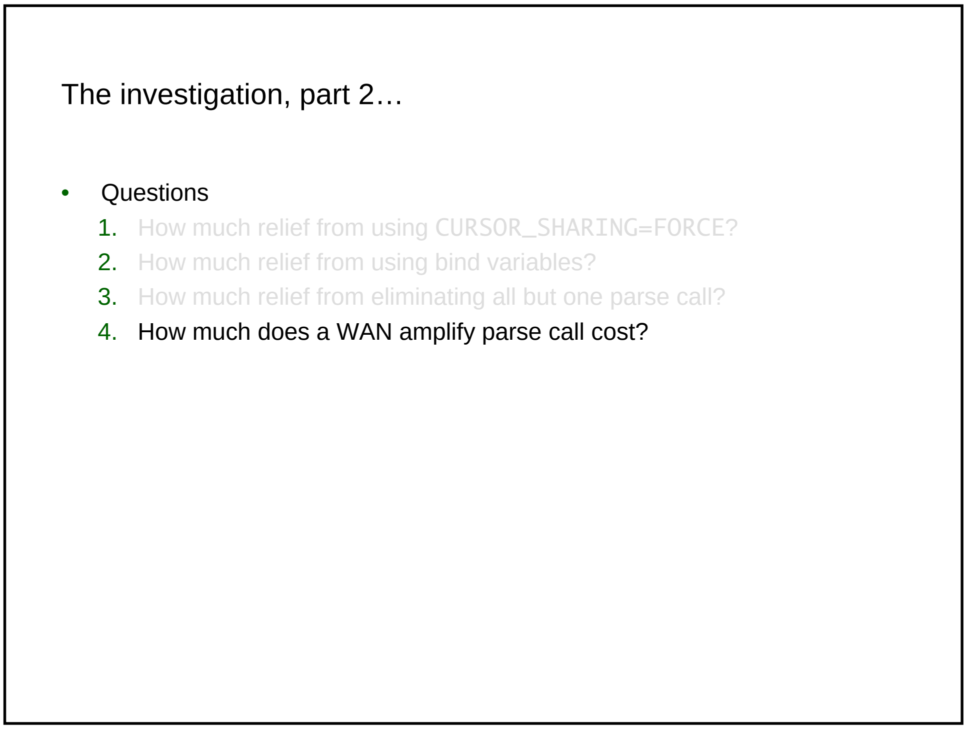
Parse once with variables via loopback…

```
v92_ora_6780.trc
v92, 5000 rows, 1 parse, variables, exact

Response Time Component               Duration     # Calls     Dur/Call
----------------------------- --------------- ----------- ------------
unaccounted-for                   0.9s  53.3%
SQL*Net message from client       0.4s  24.1%       5,010    0.000077s
CPU service                       0.3s  19.3%       5,031    0.000062s
log file sync                     0.0s   2.7%           1    0.043243s
SQL*Net message to client         0.0s   0.6%       5,010    0.000002s
----------------------------- --------------- ----------- ------------
Total                             1.6s 100.0%
```



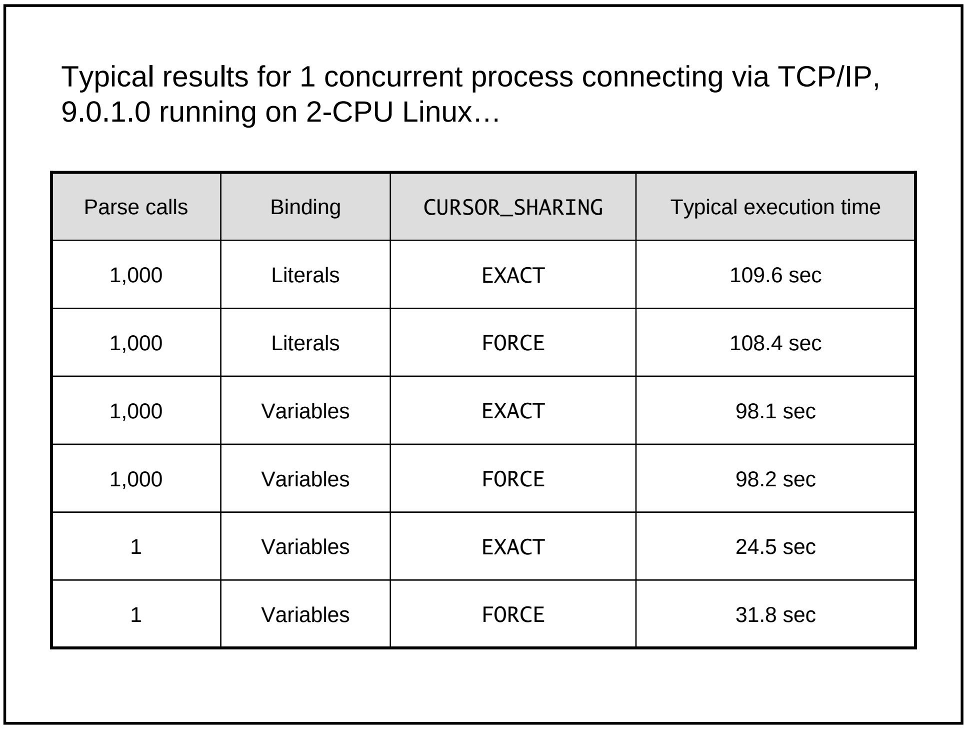
Parse once with variables via loopback, with CURSOR_SHARING=FORCE…

```
v92_ora_2400.trc
v92, 5000 rows, 1 parse, variables, force

Response Time Component               Duration     # Calls     Dur/Call
----------------------------- --------------- ----------- ------------
unaccounted-for                  2.2s  76.5%
SQL*Net message from client      0.4s  15.9%       5,010    0.000089s
CPU service                      0.2s   7.1%       5,035    0.000040s
SQL*Net message to client        0.0s   0.3%       5,010    0.000002s
log file sync                    0.0s   0.2%           1    0.005594s
----------------------------- --------------- ----------- ------------
Total                            2.8s 100.0%
```



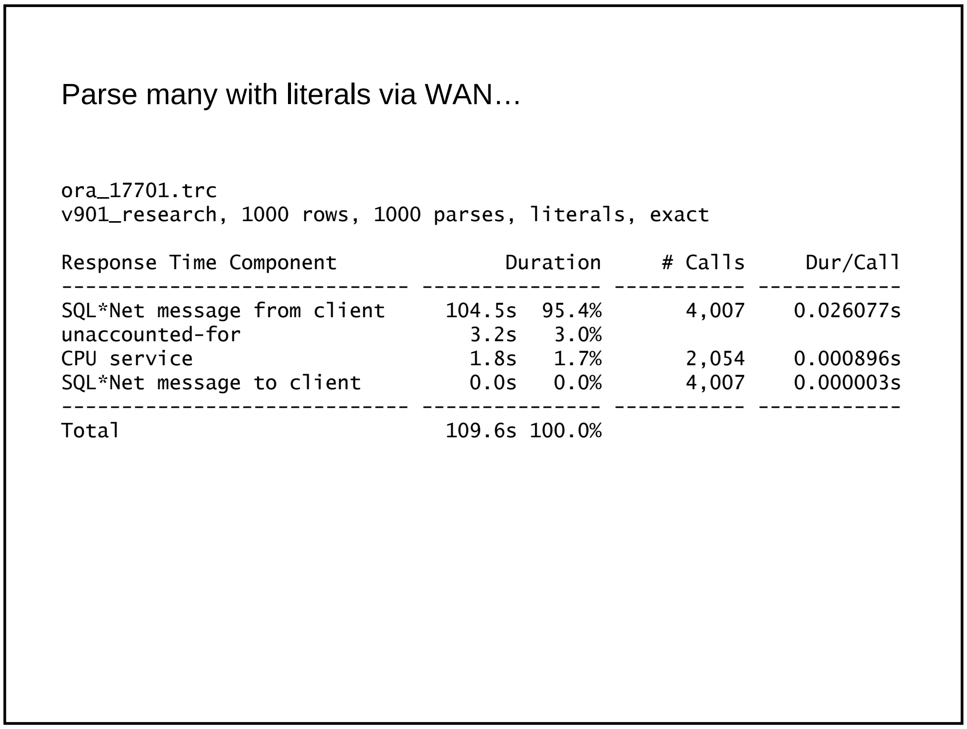
The investigation, part 2…

- Questions
  1. How much relief from using CURSOR_SHARING=FORCE?
  2. How much relief from using bind variables?
  3. How much relief from eliminating all but one parse call?
  4. How much does a WAN amplify parse call cost?



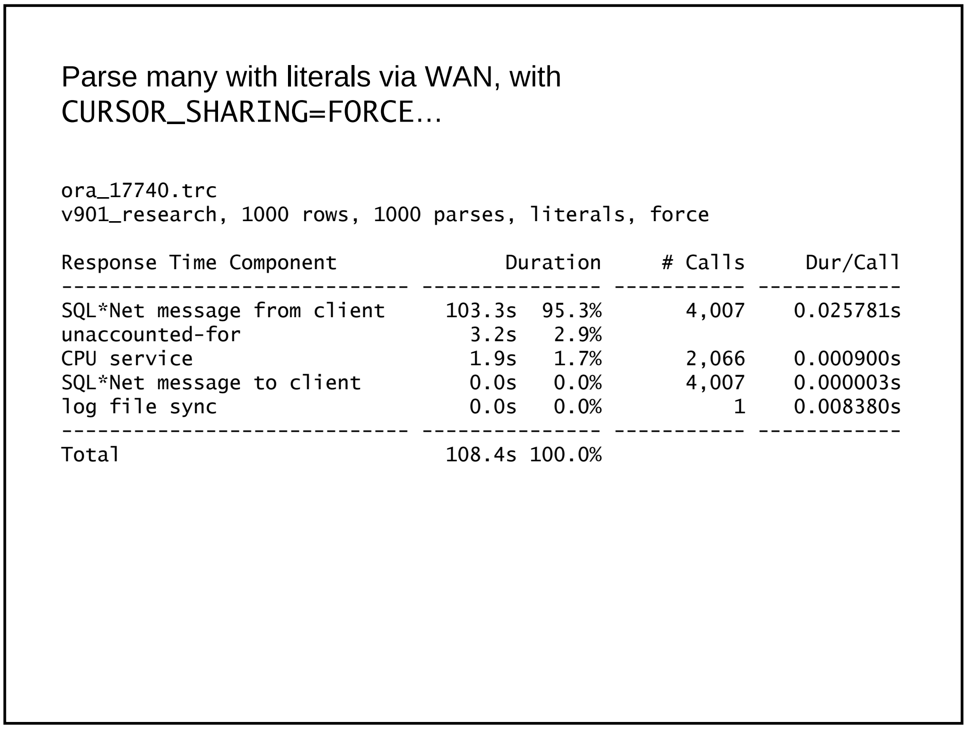
Typical results for 1 concurrent process connecting via TCP/IP, 9.0.1.0 running on 2-CPU Linux…

Parse calls	Binding	CURSOR_SHARING	Typical execution time
1,000	Literals	EXACT	109.6 sec
1,000	Literals	FORCE	108.4 sec
1,000	Variables	EXACT	98.1 sec
1,000	Variables	FORCE	98.2 sec
1	Variables	EXACT	24.5 sec
1	Variables	FORCE	31.8 sec



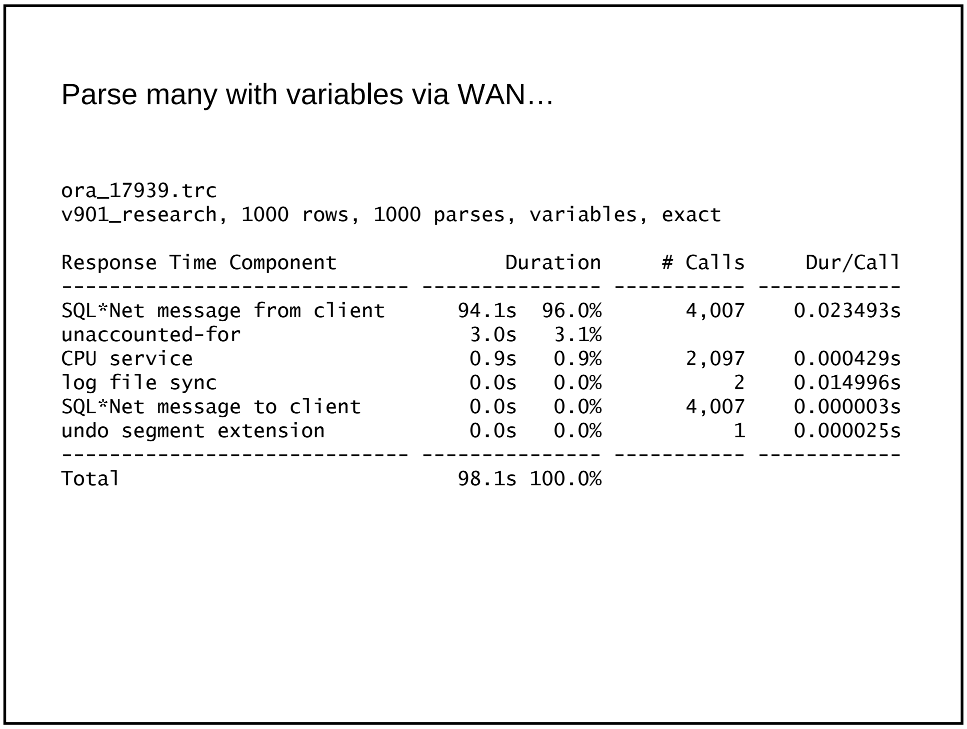
Parse many with literals via WAN…

```
ora_17701.trc
v901_research, 1000 rows, 1000 parses, literals, exact

Response Time Component                Duration     # Calls     Dur/Call
----------------------------- --------------- ----------- ------------
SQL*Net message from client     104.5s  95.4%       4,007    0.026077s
unaccounted-for                   3.2s   3.0%
CPU service                       1.8s   1.7%       2,054    0.000896s
SQL*Net message to client         0.0s   0.0%       4,007    0.000003s
----------------------------- --------------- ----------- ------------
Total                           109.6s 100.0%
```



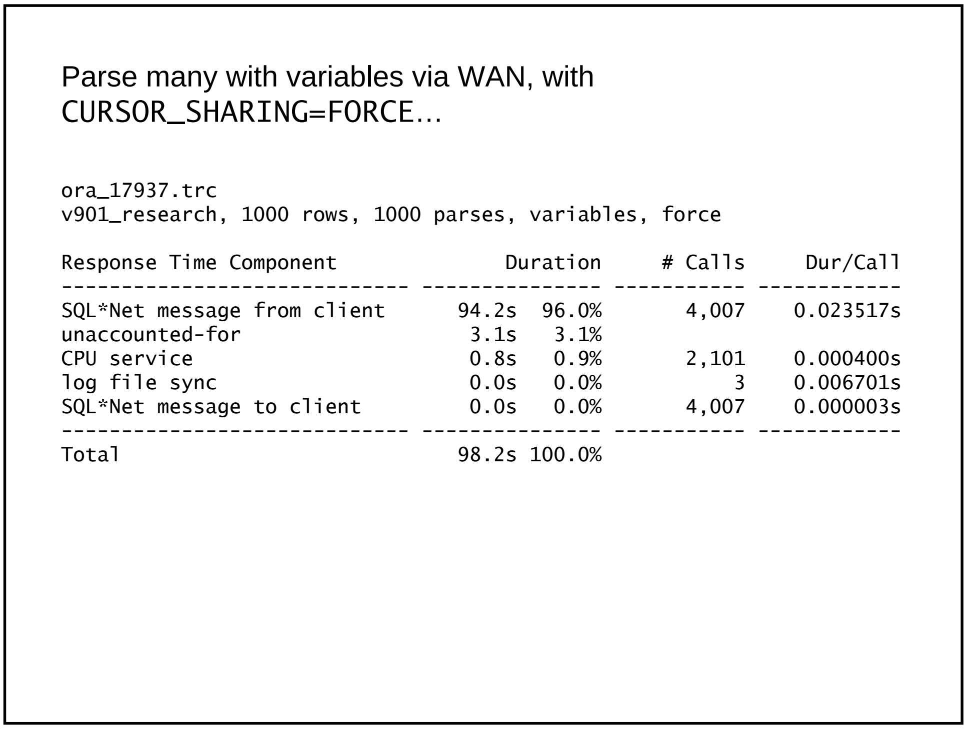
Parse many with literals via WAN, with CURSOR_SHARING=FORCE…

```
ora_17740.trc
v901_research, 1000 rows, 1000 parses, literals, force

Response Time Component             Duration     # Calls     Dur/Call
----------------------------- --------------- ----------- ------------
SQL*Net message from client     103.3s  95.3%       4,007    0.025781s
unaccounted-for                   3.2s   2.9%
CPU service                       1.9s   1.7%       2,066    0.000900s
SQL*Net message to client         0.0s   0.0%       4,007    0.000003s
log file sync                     0.0s   0.0%           1    0.008380s
----------------------------- --------------- ----------- ------------
Total                           108.4s 100.0%
```



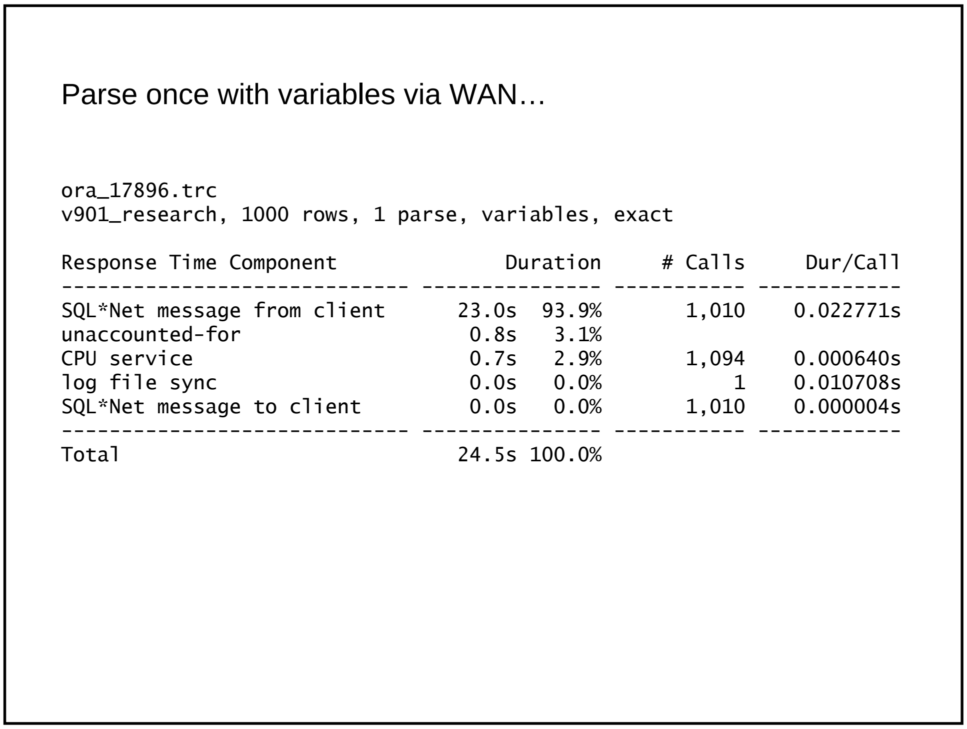
Parse many with variables via WAN…

```
ora_17939.trc
v901_research, 1000 rows, 1000 parses, variables, exact

Response Time Component             Duration     # Calls     Dur/Call
----------------------------- --------------- ----------- ------------
SQL*Net message from client      94.1s  96.0%       4,007    0.023493s
unaccounted-for                   3.0s   3.1%
CPU service                       0.9s   0.9%       2,097    0.000429s
log file sync                     0.0s   0.0%           2    0.014996s
SQL*Net message to client         0.0s   0.0%       4,007    0.000003s
undo segment extension            0.0s   0.0%           1    0.000025s
----------------------------- --------------- ----------- ------------
Total                            98.1s 100.0%
```



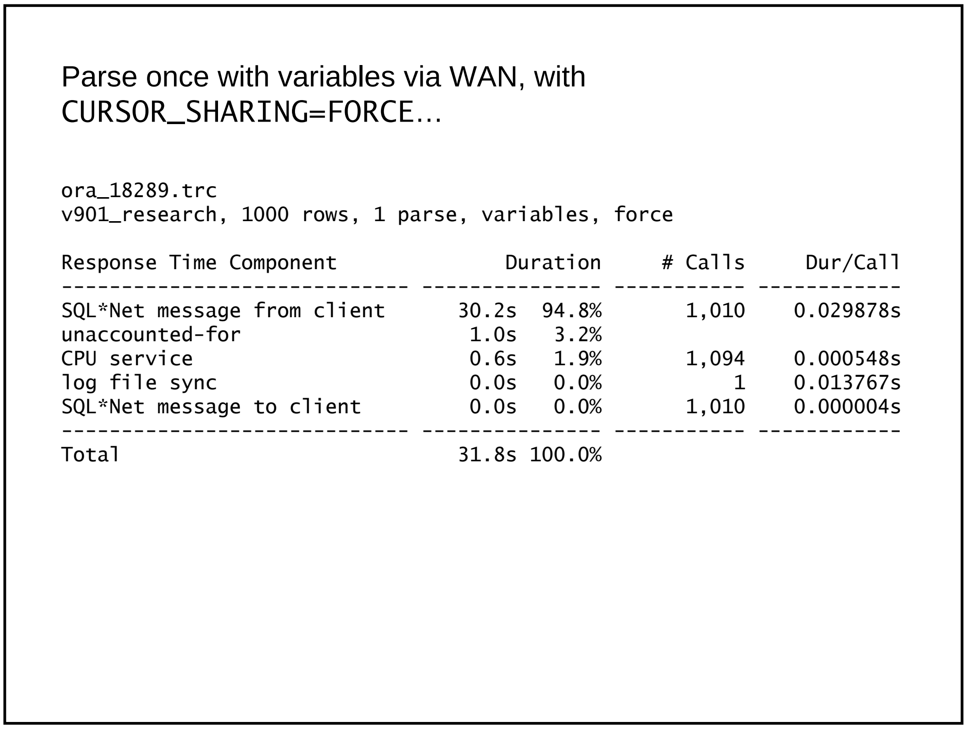
Parse many with variables via WAN, with CURSOR_SHARING=FORCE…

```
ora_17937.trc
v901_research, 1000 rows, 1000 parses, variables, force

Response Time Component              Duration     # Calls     Dur/Call
----------------------------- --------------- ----------- ------------
SQL*Net message from client      94.2s  96.0%       4,007    0.023517s
unaccounted-for                   3.1s   3.1%
CPU service                       0.8s   0.9%       2,101    0.000400s
log file sync                     0.0s   0.0%           3    0.006701s
SQL*Net message to client         0.0s   0.0%       4,007    0.000003s
----------------------------- --------------- ----------- ------------
Total                            98.2s 100.0%
```



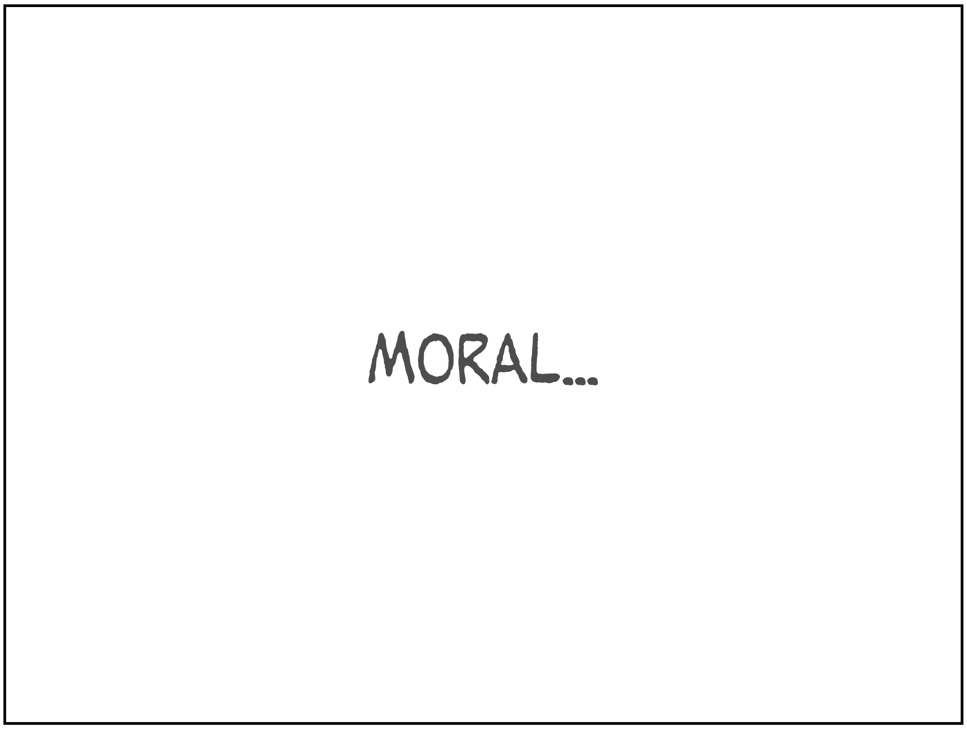
Parse once with variables via WAN…

```
ora_17896.trc
v901_research, 1000 rows, 1 parse, variables, exact

Response Time Component              Duration     # Calls     Dur/Call
----------------------------- --------------- ----------- ------------
SQL*Net message from client      23.0s  93.9%       1,010    0.022771s
unaccounted-for                   0.8s   3.1%
CPU service                       0.7s   2.9%       1,094    0.000640s
log file sync                     0.0s   0.0%           1    0.010708s
SQL*Net message to client         0.0s   0.0%       1,010    0.000004s
----------------------------- --------------- ----------- ------------
Total                            24.5s 100.0%
```



Parse once with variables via WAN, with CURSOR_SHARING=FORCE…

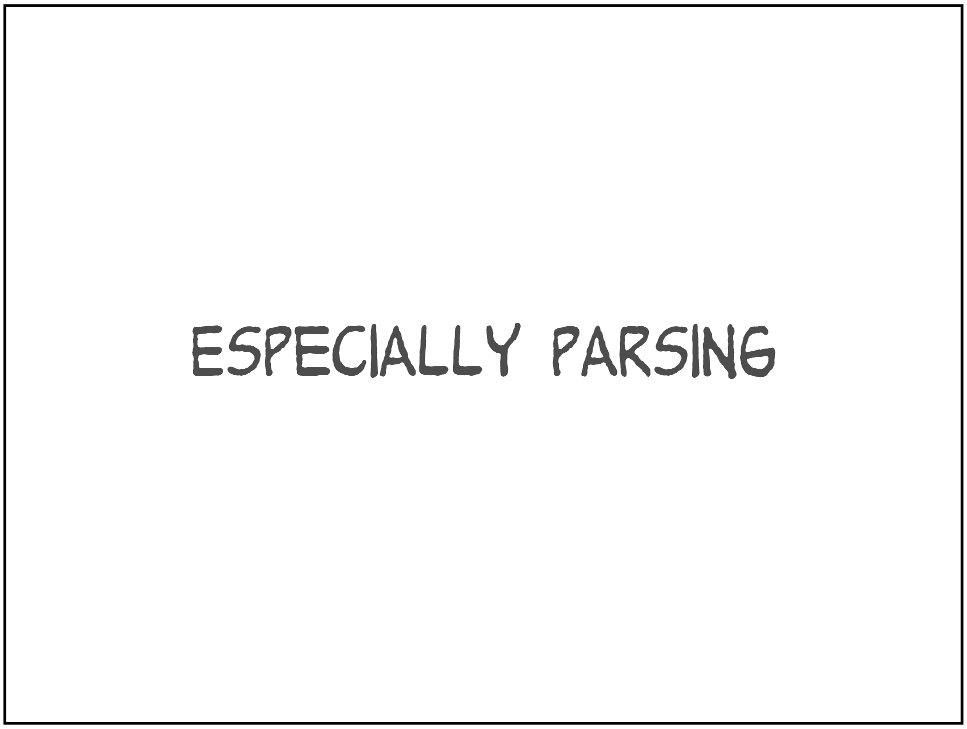
```
ora_18289.trc
v901_research, 1000 rows, 1 parse, variables, force

Response Time Component             Duration     # Calls     Dur/Call
----------------------------- --------------- ----------- ------------
SQL*Net message from client      30.2s  94.8%       1,010    0.029878s
unaccounted-for                   1.0s   3.2%
CPU service                       0.6s   1.9%       1,094    0.000548s
log file sync                     0.0s   0.0%           1    0.013767s
SQL*Net message to client         0.0s   0.0%       1,010    0.000004s
----------------------------- --------------- ----------- ------------
Total                            31.8s 100.0%
```



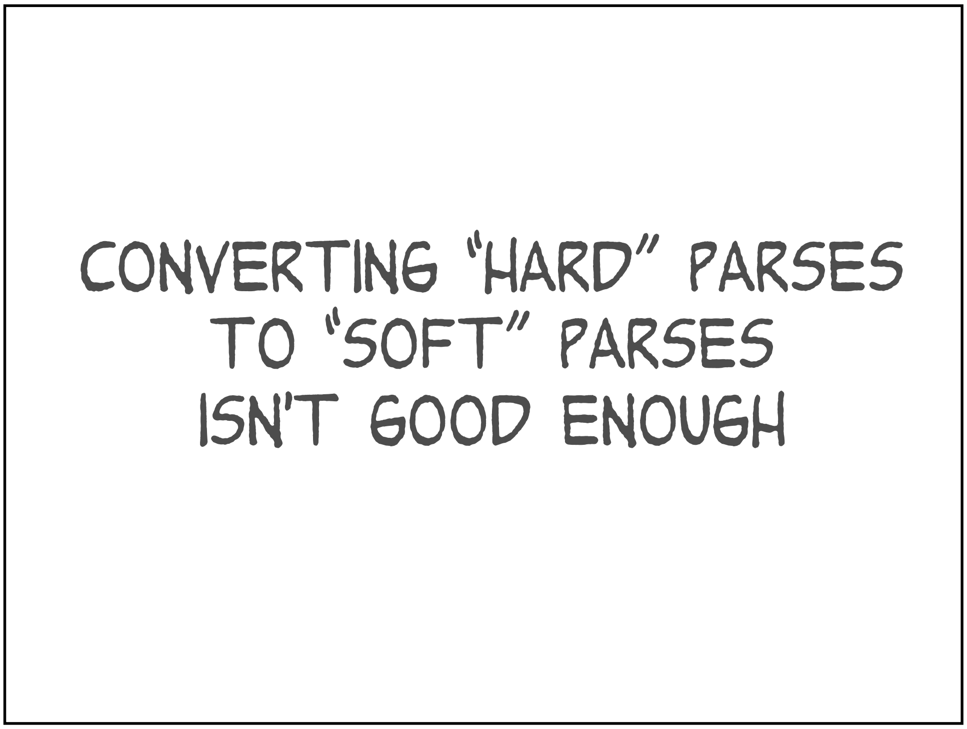
MORAL…

UNNECESSARY PARSING
IS BAD

UNNECESSARY
ANYTHING IS BAD



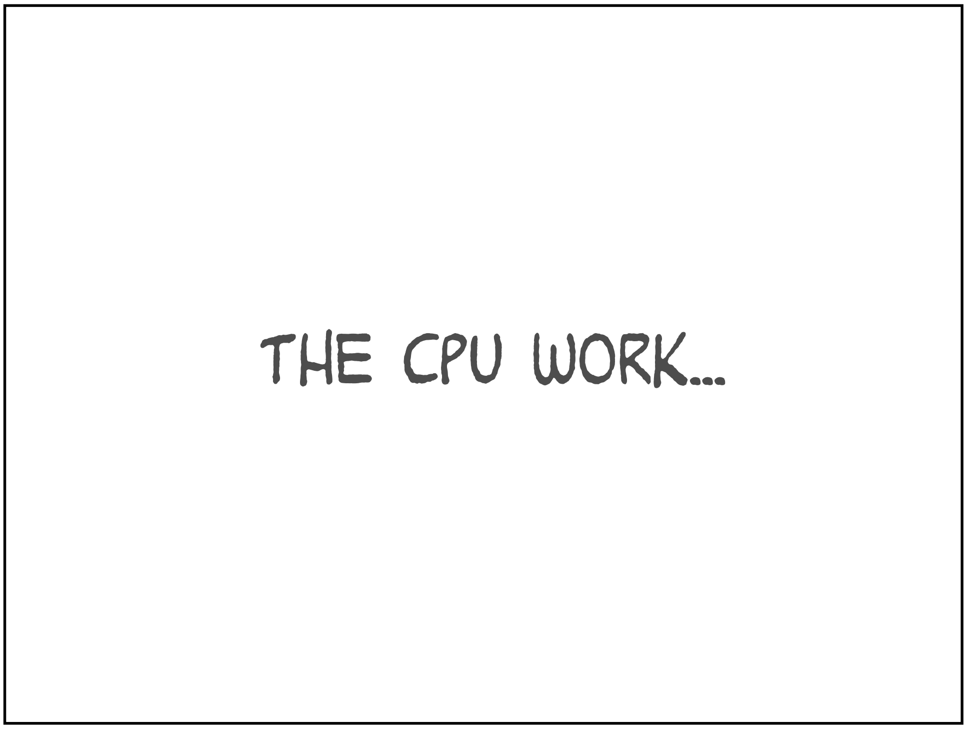
ESPECIALLY PARSING



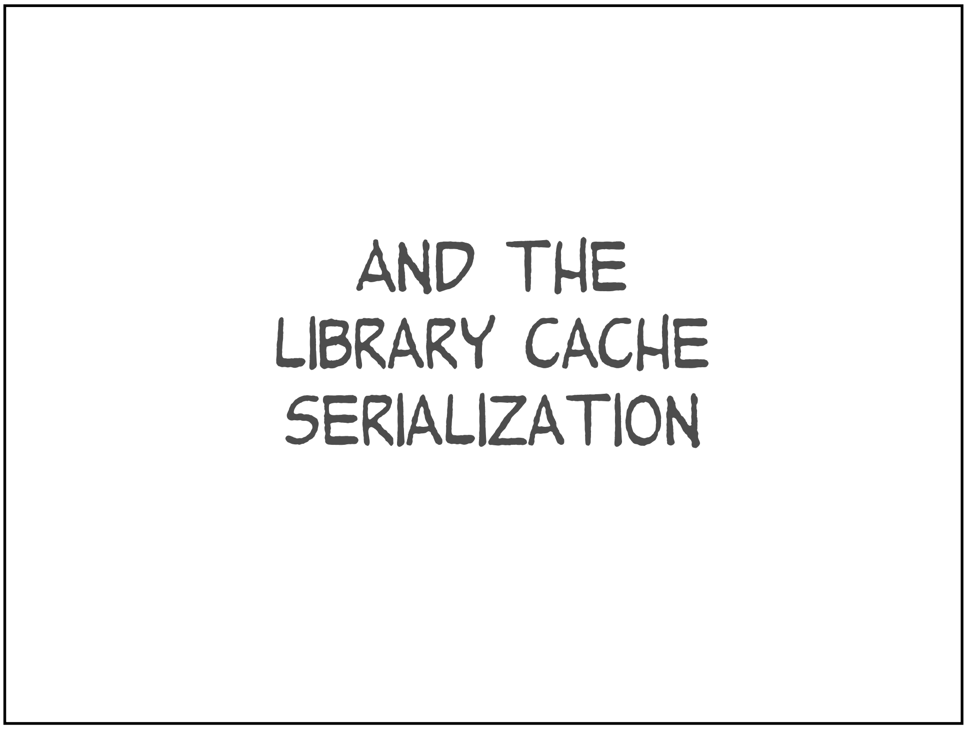
CONVERTING "HARD" PARSES
TO "SOFT" PARSES
ISN'T GOOD ENOUGH

SAVING THE SHARED POOL WORK IS NICE, BUT IT'S NOT ENOUGH

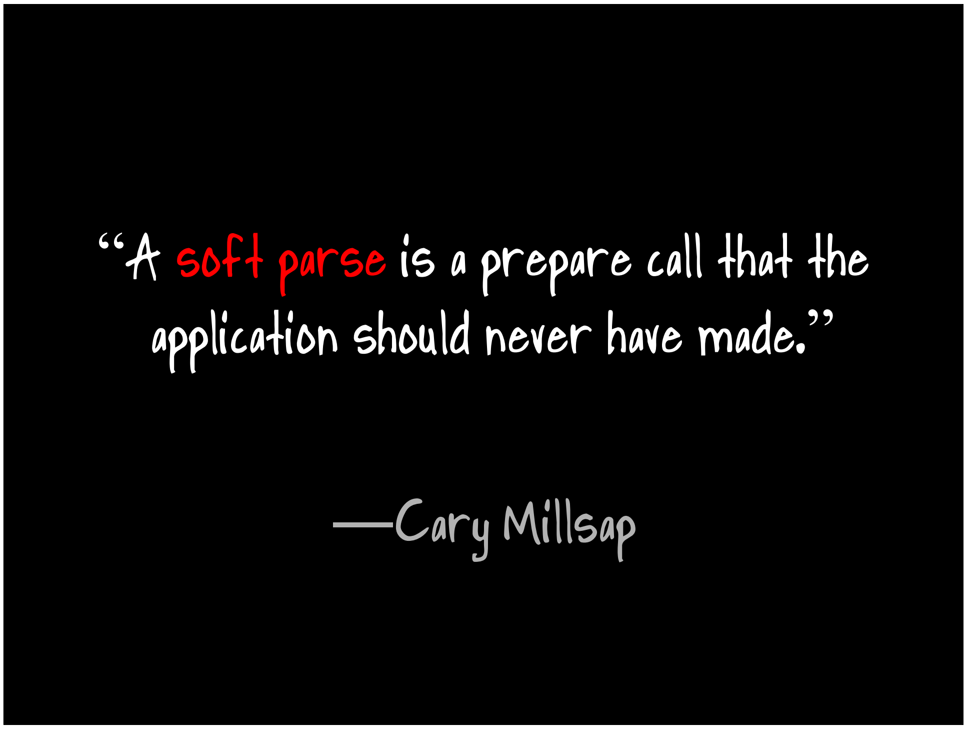
THERE'S ALSO THE NETWORK I/O...



THE CPU WORK...

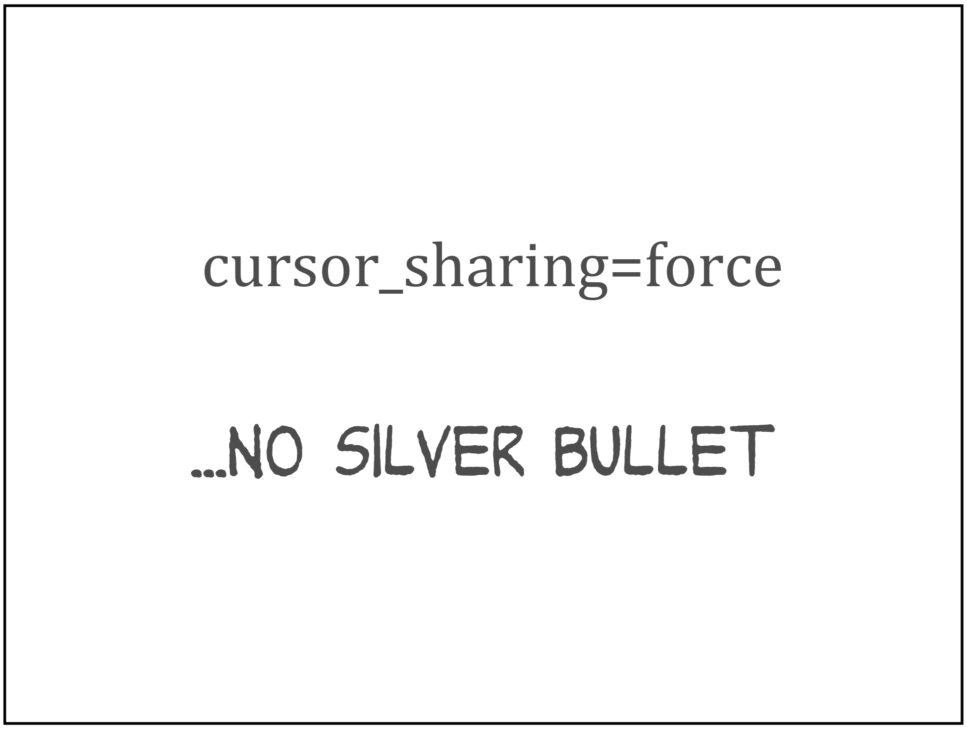


AND THE
LIBRARY CACHE
SERIALIZATION



"A soft parse is a prepare call that the application should never have made."

—Cary Millsap



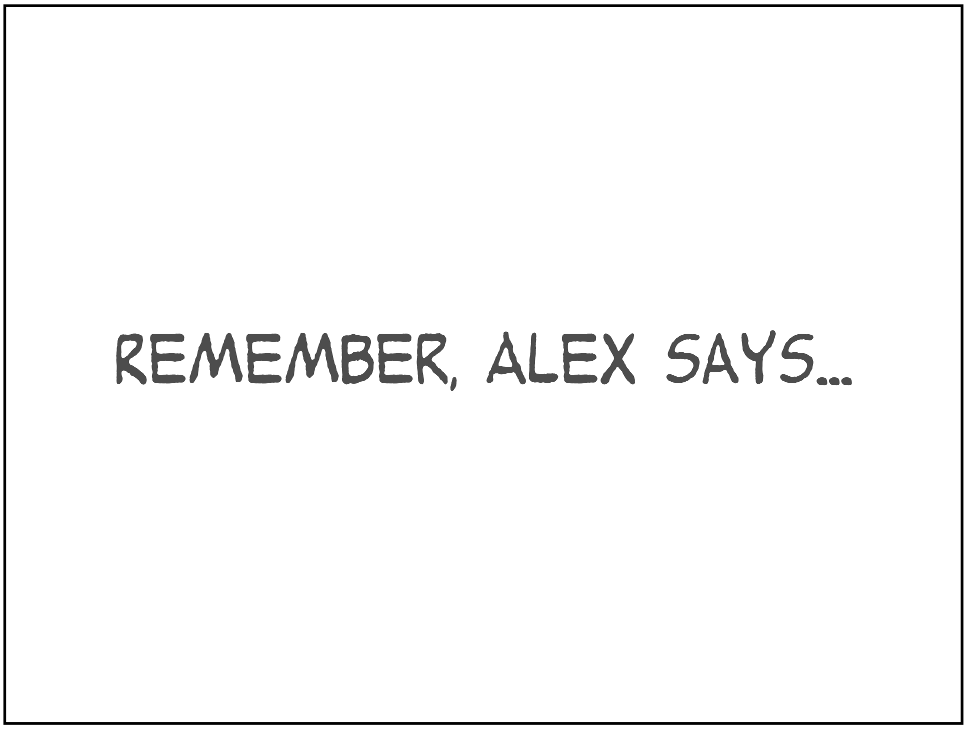
cursor_sharing=force

...NO SILVER BULLET

IT HURTS PERFORMANCE
OF *GOOD*
APPLICATION CODE

SEE FOR YOURSELF

SQL TRACE



REMEMBER, ALEX SAYS...

Questions?

3

Skew

THE DEAL WITH SKEW...

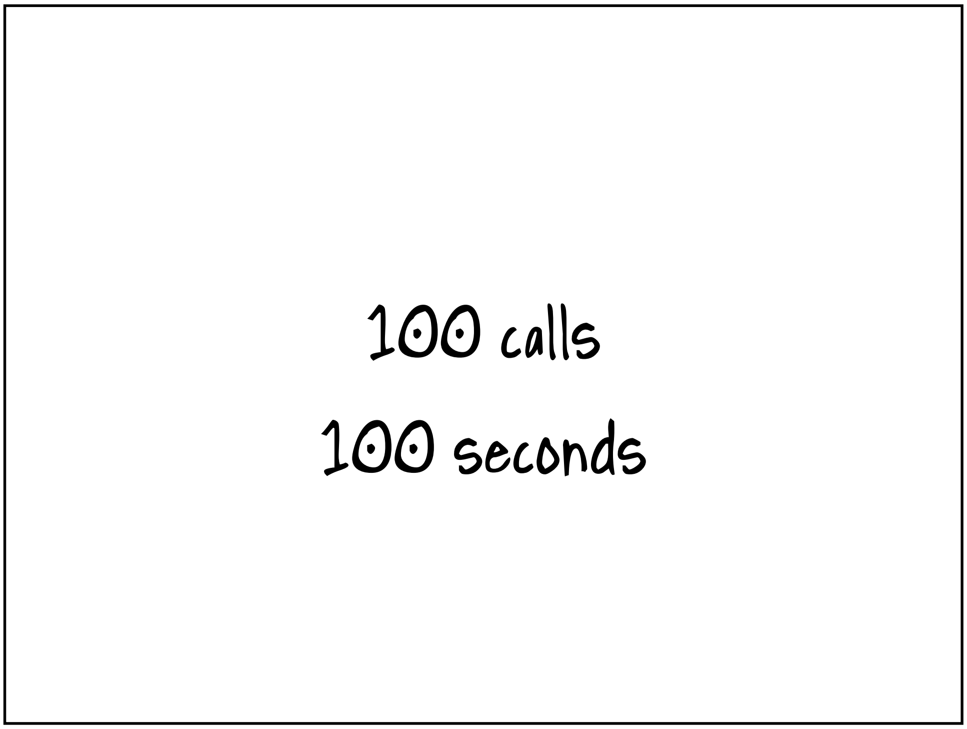
skew

=

non-uniformity

EXAMPLE...

HOW MUCH TIME
WILL YOU SAVE...



100 calls

100 seconds

...IF YOU ELIMINATE
HALF THE CALLS?

If

100 calls

100 seconds

...then

50 calls

50 seconds

Right?

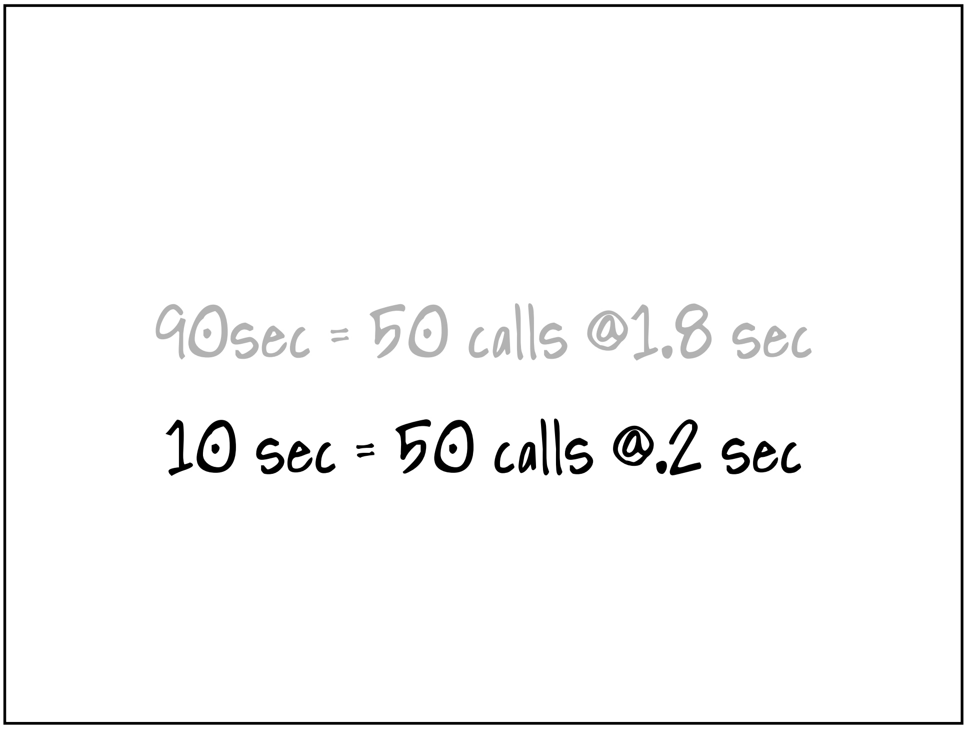
ANSWER…
YOU CAN'T KNOW.

EXAMPLE…

90sec = 50 calls @1.8 sec

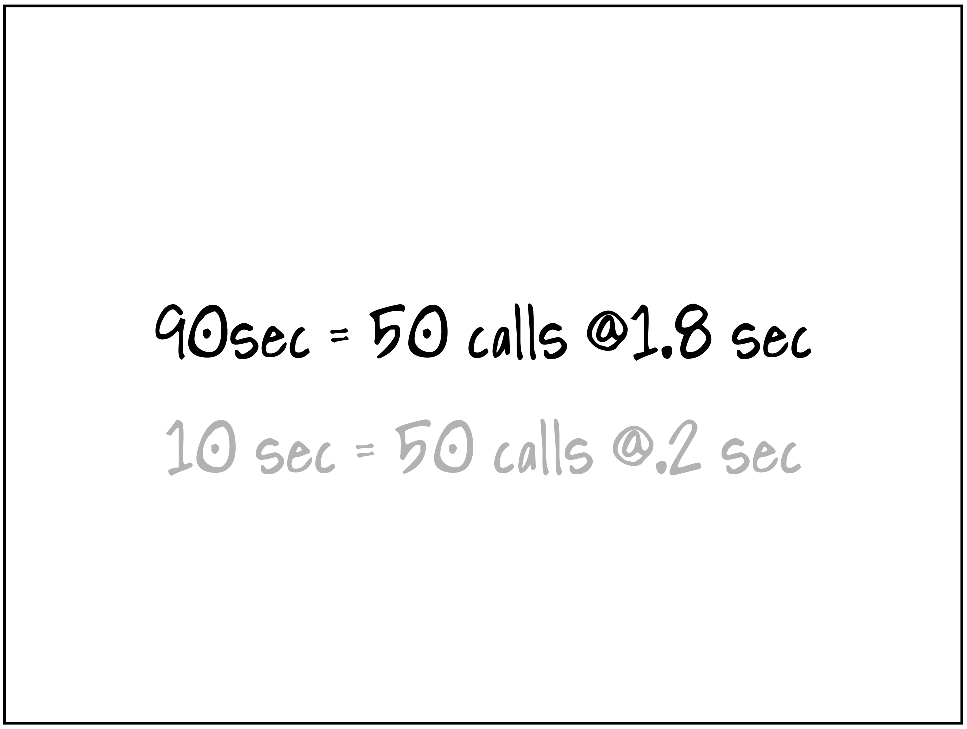
10 sec = 50 calls @.2 sec

DEPENDS WHICH 50 YOU ELIMINATE...



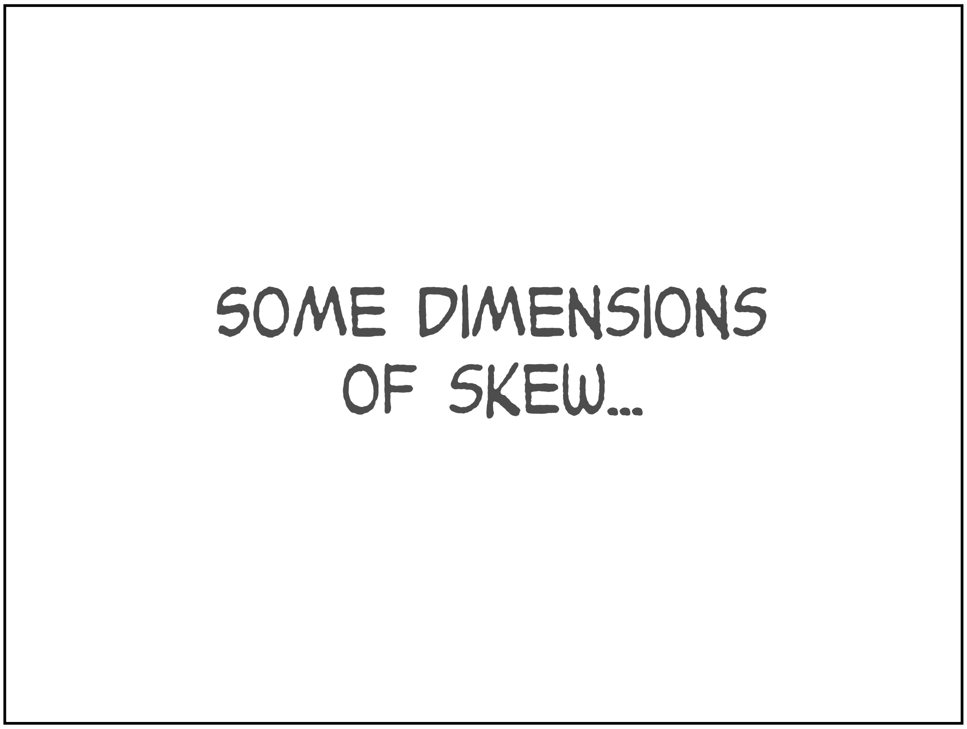
90sec = 50 calls @1.8 sec

10 sec = 50 calls @.2 sec

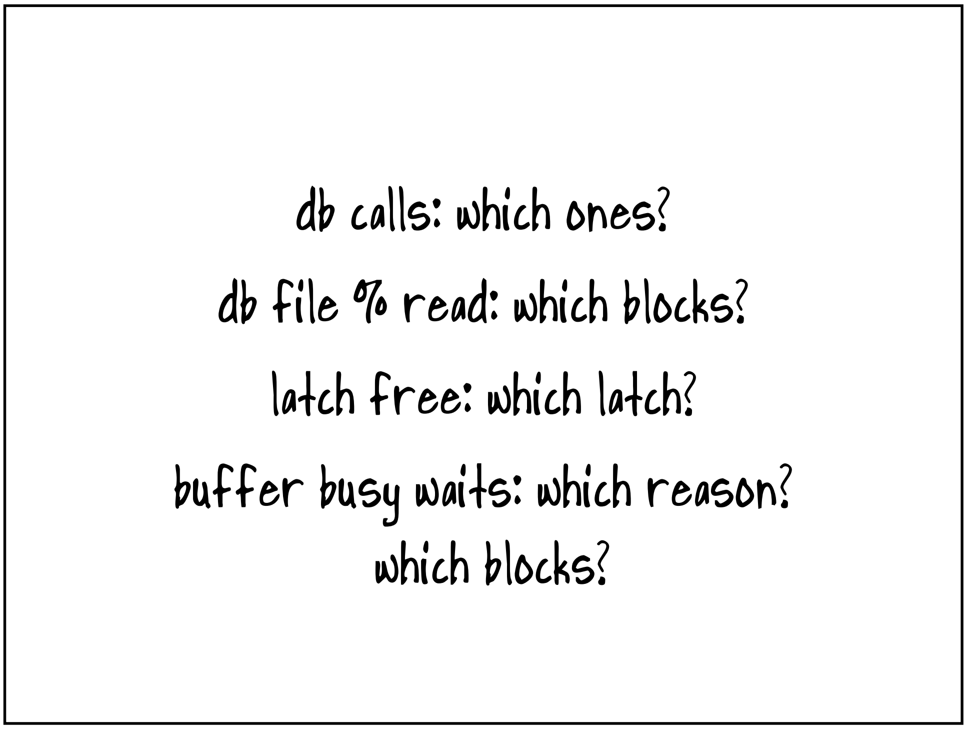


90sec = 50 calls @1.8 sec

10 sec = 50 calls @.2 sec



SOME DIMENSIONS OF SKEW...



db calls: which ones?

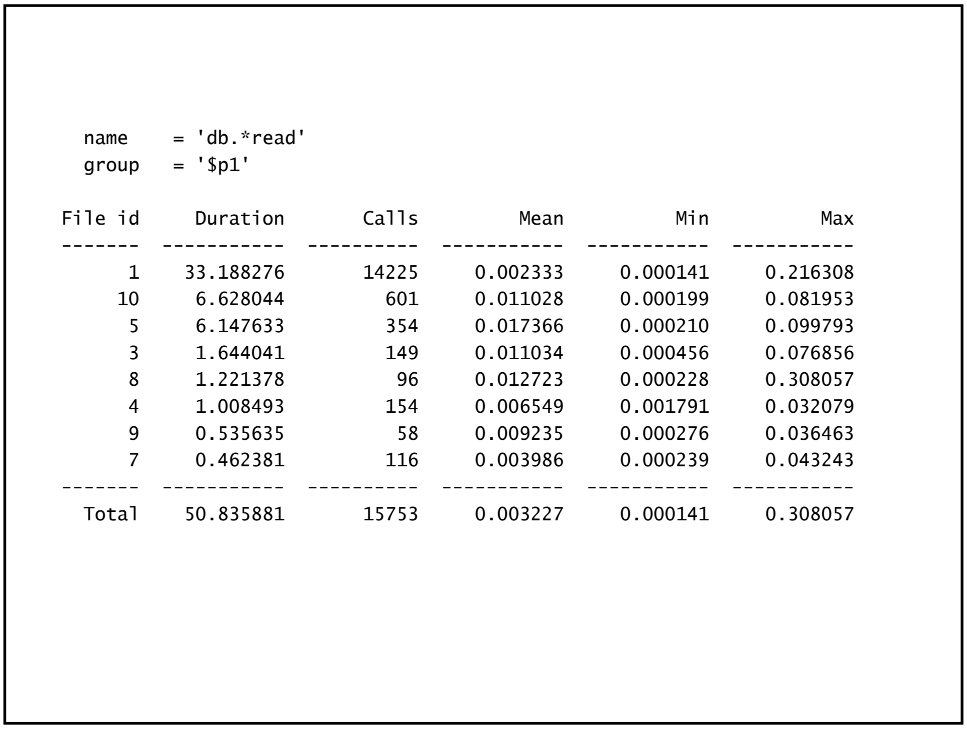
db file % read: which blocks?

latch free: which latch?

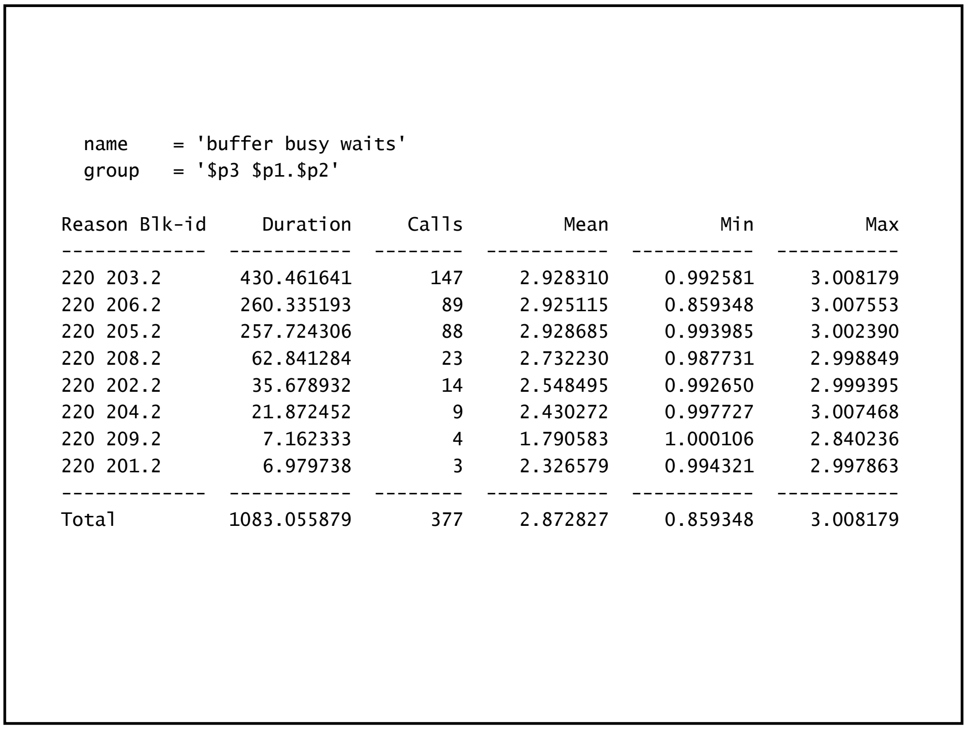
buffer busy waits: which reason?
which blocks?

SOME EXAMPLES...

```
   name    = 'db.*read'
   group   = '$p3'

Blks/read     Duration       Calls        Mean         Min         Max
---------  -----------  ----------  ----------- ----------- -----------
        1    48.068934       15556     0.003090    0.000141    0.308057
       16     1.468489          84     0.017482    0.001752    0.082294
        8     0.702916          52     0.013518    0.006832    0.044180
        5     0.147912          12     0.012326    0.000664    0.047341
        2     0.142277          20     0.007114    0.000264    0.039686
       10     0.080358           4     0.020090    0.004538    0.042498
        3     0.071654           8     0.008957    0.000436    0.029179
        4     0.069486           9     0.007721    0.000559    0.041011
        7     0.052081           4     0.013020    0.002649    0.032218
        6     0.031774           4     0.007943    0.002262    0.013016
---------  -----------  ----------  ----------- ----------- -----------
    Total    50.835881       15753     0.003227    0.000141    0.308057
```



```
  name    = 'db.*read'
  group   = '$p1'

File id     Duration       Calls         Mean          Min          Max
-------  -----------  ----------  -----------  -----------  -----------
      1    33.188276       14225     0.002333     0.000141     0.216308
     10     6.628044         601     0.011028     0.000199     0.081953
      5     6.147633         354     0.017366     0.000210     0.099793
      3     1.644041         149     0.011034     0.000456     0.076856
      8     1.221378          96     0.012723     0.000228     0.308057
      4     1.008493         154     0.006549     0.001791     0.032079
      9     0.535635          58     0.009235     0.000276     0.036463
      7     0.462381         116     0.003986     0.000239     0.043243
-------  -----------  ----------  -----------  -----------  -----------
  Total    50.835881       15753     0.003227     0.000141     0.308057
```



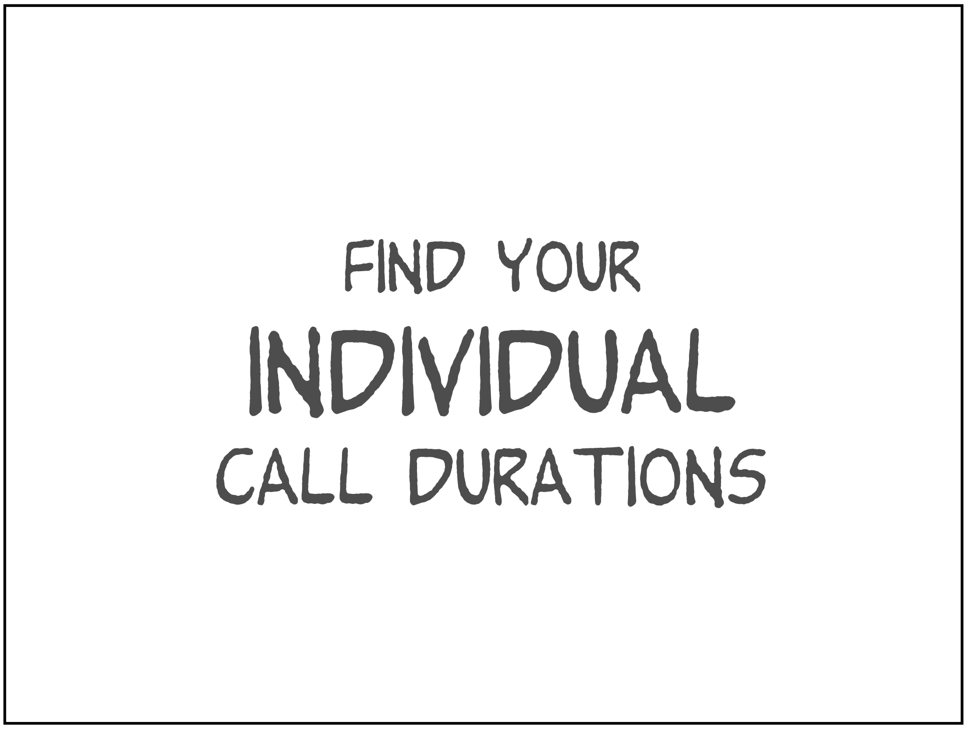
```
  name    = 'buffer busy waits'
  group   = '$p3 $p1.$p2'

Reason Blk-id     Duration     Calls         Mean          Min          Max
-------------  -----------  --------  -----------  -----------  -----------
220 203.2       430.461641       147     2.928310     0.992581     3.008179
220 206.2       260.335193        89     2.925115     0.859348     3.007553
220 205.2       257.724306        88     2.928685     0.993985     3.002390
220 208.2        62.841284        23     2.732230     0.987731     2.998849
220 202.2        35.678932        14     2.548495     0.992650     2.999395
220 204.2        21.872452         9     2.430272     0.997727     3.007468
220 209.2         7.162333         4     1.790583     1.000106     2.840236
220 201.2         6.979738         3     2.326579     0.994321     2.997863
-------------  -----------  --------  -----------  -----------  -----------
Total          1083.055879       377     2.872827     0.859348     3.008179
```

MORAL...

DURATION OF EACH CALL

≠

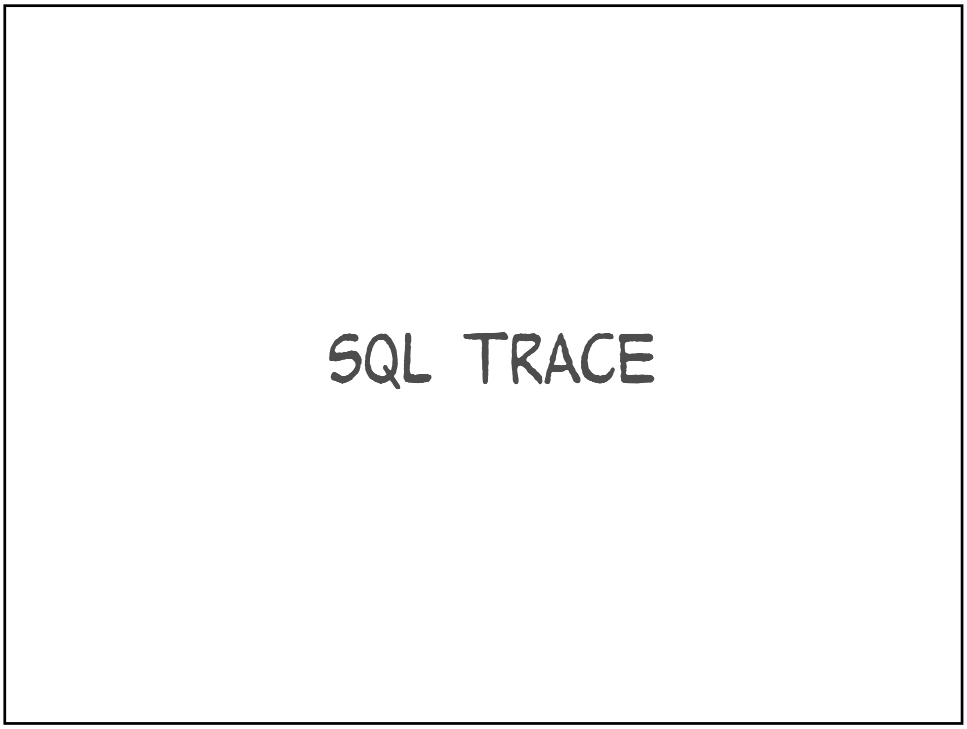
AVERAGE CALL DURATION



FIND YOUR

INDIVIDUAL

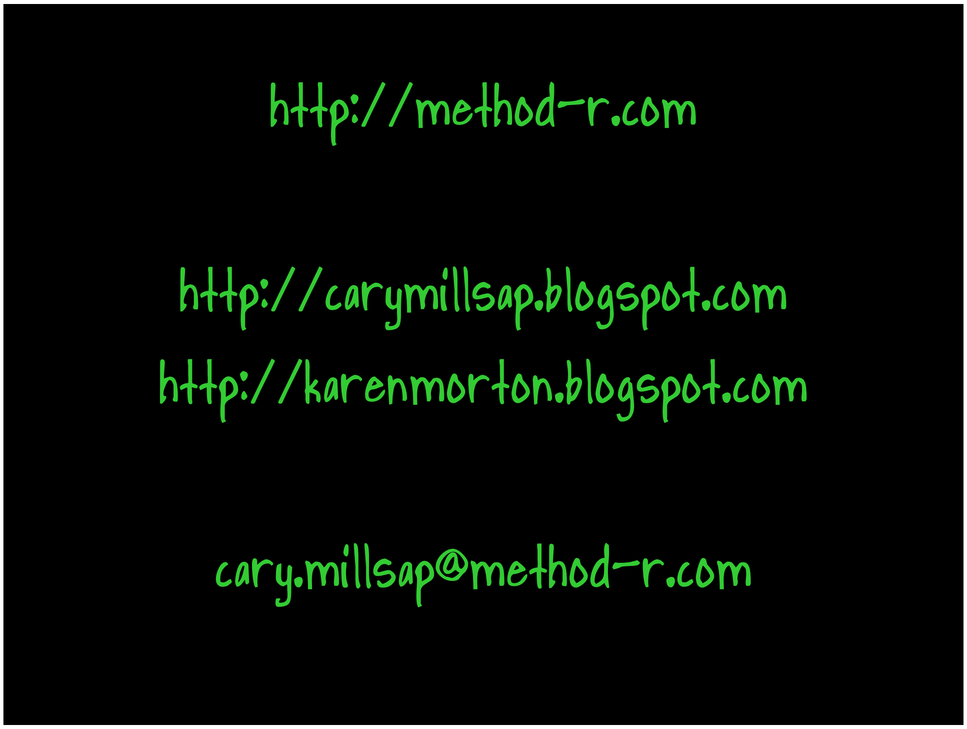
CALL DURATIONS



SQL TRACE

Questions?

Thank you



http://method-r.com

http://carymillsap.blogspot.com

http://karenmorton.blogspot.com

cary.millsap@method-r.com