Abstract + notes…

Make IT 2022

When Capacity is Limited.

We got confronted with old hardware and notably very slow storage.

The job we had to do involved, unavoidably, a large amount of Writing and reading.
We'll present and discuss some of the measures we tried / failed / succeeded to survive until we get our new+shiny hardware…

It was a bit like pointing firehoses at a sink, and then try not to let it overflow.

Next slide is screen+intro,
Put abstract here, in case you need it.

Make IT
2022

When Capacity is Limited

Piet de Visser

The Simple Oracle DBA

Combining Microservices and Databases can work,
and I have a few Ideas on how to do that Efficiently.

Make IT
2022

Capacity of a DWH system. Load jobs over night, Processing time way into the morning.
Infrastructure unclear, system-admin … ?? But users definitely not happy..

Agenda (approx 45 minutes)

Make IT 2022

Agenda. What you need to know. Sometimes, I told this story in 10min.
You can now go listen next door..

ETL type jobs: collect data …

Make IT 2022

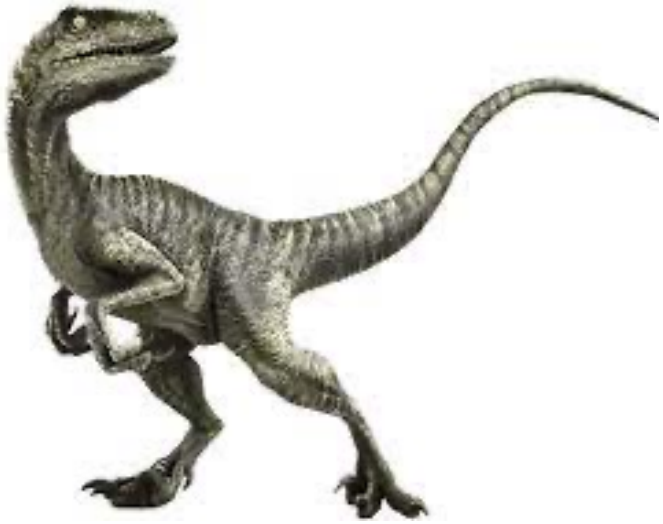
20+ source systems

Millions of records per source.

Read + Write + Process + R + W + R …; 1TB redo/night

Batch-jobs start “uncontrolled”

IaaS: Little contact with “infra- and admin teams”



ETL type system. Control mostly by source systems.
Little contact with admin / storage. Infra is I-aaS. (gov…)

I aa S: Admin… (not much info..)

Make IT 2022

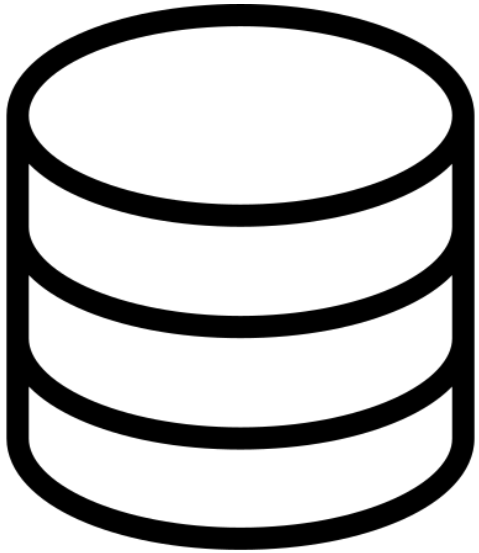
Servers are Linux on ESX.
Storage is Hitachi, but “unknown”

“System is not Busy” ??? (1TB archives?)

Two small steps:

1. **Finally SQL-Access to V$ and AWR**
2. **Some info and … “Do you want more CPUs?”**

Eeehhh?



Info: Server ws 12cpu + 40G memory. Connection to storage was fiberchannel
Give them extra CPU + memory, maybe they go away….

AWR…

Top 10 Foreground Events by Total Wait Time

Event	Waits	Total Wait Time (sec)	Wait Avg(ms)	% DB time	Wait Class
direct path read	831,119	19,1K	22.92	37.7	User I/O
direct path read temp	647,704	10,2K	15.76	20.2	User I/O
direct path write temp	594,692	5126,2	8.62	10.1	User I/O
DB CPU		3797		7.5	
db file parallel read	5,416	1623,4	299.74	3.2	User I/O
db file scattered read	56,030	1091,3	19.48	2.2	User I/O
direct path write	169,400	907,3	5.36	1.8	User I/O
db file sequential read	51,655	447,4	8.66	.9	User I/O
read by other session	13,127	254,9	19.42	.5	User I/O
control file sequential read	31,604	206,6	6.54	.4	System I/O

Wait Classes by Total Wait Time

Wait Class	Waits	Total Wait Time (sec)	Avg Wait (ms)	% DB time	Avg Active Sessions
User I/O	2,414,851	38,789	16.06	76.7	42.9
DB CPU		3,797		7.5	4.2
Other	38,666	2,040	52.76	4.0	2.3

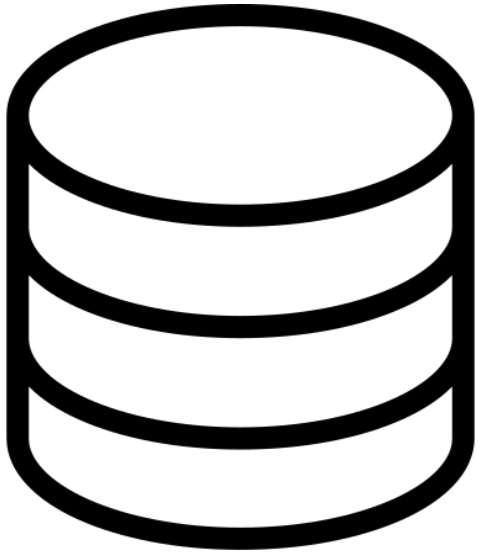
Getting to the Problem

Make IT 2022

Your tools: SQL, AWR + ASH ! **(only sql-access, hm..)**

We Observe: IO **(e.g. not CPU or memory)**

“Ingesting” and “checking” a TB of data is just ... IO.



All night: IO at 70% ++ of db-time.
(the “server was quiet”, but waiting for IO)

Also: quite Slow : 20+ ms for reads..
(this DB is not happy with storage…)

We find: an IO bottleneck.
Hence, the offer of more CPU+memr may not help..

Infra-team: we have CPUs ...

Make IT 2022

Them: CPU + Mem is ”available”

We: need IO !

Manager: hm...

Our “server” got doubled in size.

Now 24 CPUs + 80G Memory.
Adjusted spfile…

Zero Effect.

The “available” solution was more CPU, and you cannot stop a manager from “taking what is available”
The effect was … No Change… : jobs still too slow, still IO problem in AWR.

Pause needed…

Make IT 2022

“Bigger Server doesnt help”…. (who would have thought)

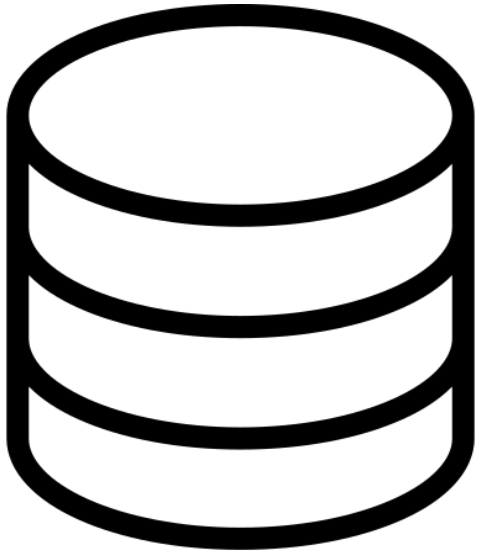
Them: You must have software problem…

We : IO problem… !

Next:

a. **Examine “the process”**
b. **Examine more AWRs**
c. **Benchmarks.**

No Progress yet.



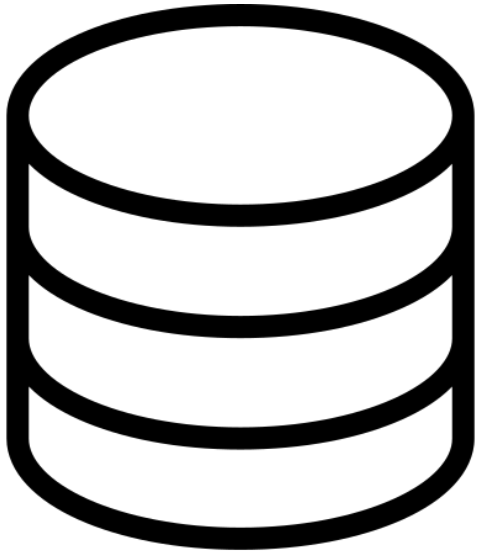
The “available” solution was more CPU, and you cannot stop a manager from “taking what is available”
The effect was … No Change… : jobs still too slow, still IO problem in AWR.

A. The process..

Make IT 2022

Ingesting + classifying data.
Ingest via DB-links (good, no waits here - yet)
Software was all PL/SQL (#SmartDB, good)

Data-team could demonstrate:
- Process already quite “optimal”, Efficient.
- Read-once, process + store + verify (and Stats!)



Problem 1: Timing. Overload.
Problem 2: Data-Volume “Search through 10TB”

Not much Progress yet.

The process of Ingesting was well thought out. Not much IO could be “saved” here.
Programmers had smart temp-tables, but data volume was too large for cache.

B. The AWRs..

Make IT 2022

System was Busy all night…

IO: Direct Path Reads.
IO: Temp (also dp-reads)

Average Active Sessions : >CPUs
Many Parallel-hints (64 !?)

Problem 3: Too much Reading...
Problem 4: Reading too slow… (20ms??)

Some Progress: we suspect overload!

From the AWR it was clear: we do too much IO, Many Parr-hints, "because that helped…)
AND our IO is too slow. We begin to guess.. : Overload.

C. Benchmarks…

Make IT 2022

So… What _Can_ this system do ?

IO times of 20ms ??
(nb: SLOB – Kevin Closson, but we didnt have server-access)

If we “test” : dp-reads = 4.5 ms (still slow, but better)

If we test “multiple”: max 8 session…
If we test Parallel hints: max DOP 8 or 16

Aha: IO Can be better, IF… we dont do so much of it.

Testing : read a large table, by quiet daytime.. Better IO! (testing isn’t simple.. Many factors, notably cache…)
More testing: we can do about 8 processes, or DOP-16 before IO deteriorates.

We Begin to see the light….

Make IT 2022

Need to “reduce the load” …

Reduce the DOP-hints: max 8

Even if we have 24 CPU’s…

Prevent multiple jobs.. Max 2 jobs at any time.

=> Difficulty is “coordination of source systems”

Also:

- Gather-Stats by each job, not by 22:00..

We now begin to know our problem and possible solusitons.
We need to prevent overload – (and the gathering of stats, as a sde-note, shluld not be automatic…)

Reduce load…: Control Jobs-starting.

Make IT 2022

Run only 1 or 2 jobs simultaneously.

Two benefits:

1. Better “managing” of IO load.

2. Focus on 1 “set of data” => Cache.

Main challenge:

Organisation (this problem not fixed yet…)

Later:

Probe + Start: Measure load before next job…

Prevent new (heavy) jobs from starting : is not always easy. And even small jobs with DoP=64 are problems.
Future : measure load before start of next job, AWR or function. But … “organisational problems”

Reduce load…: Less PX

Make IT 2022

Limit the DOP to what our IO can handle (e.g. 8)

Main benefit:

1. **Better IO times. (e.g faster).**

Main challenge:

Dev: ”but the hints helped”

Testing is now “slower”.

Note: PQ/PX only really work on a quiet system.

Note: Resmgr in future ?

The Hints that help in Dev, can be problems in Prod.
Possible use of Resmgr, but our jobs are “all equal”, and we fear pq-queued event.

Main Messages – What we Learned…

Make IT 2022

Observe and Measure

- **We needed to know: AWR. (OEM…if you have)**
- **Find your limits – your “box”**

e.g. our data-sets were too big for cache (in this case)
e.g. our IO was limited, and ”deteriorated at 8+
But YMMV!

Control your software:

- **Prevent/eliminate damaging hints.**
- **Schedule smartly - Throttle the work.**
- **Test on a “loaded” system.**

Ask for watch, tell time..
Clever Dino.

Interesting times ahead

Long term: more + faster hardware or DB-aaS
Some Managers would like and Exa-data, but that will take time + organizational effort.

Make IT
2022

Don’t Take my word for it…

Google and RTFMs

Simplicity

- **In case of doubt: Simplify!**
- **Less components**
- **Less complexity**
- **Less tricks…**

Goethe _____ (simplicity)

Majority of time I have been WRONG.
So go see for yourself – but don’t complicate life.

Quick Q & A (3 min) 3 .. 2 .. 1 .. Zero

Make IT 2022

- **Questions ?**
- **Reactions ?**
- **Experiences from the audience ?**
- **@pdevisser (twitter..)**

Question and Answer time. Discussion welcome (what about that Razor?)
Teach me something: Tell me where you do NOT AGREE.

Blank

Make IT 2022

End - This slide intentionally left Blank…

Somehow, the unit and concidiont of “database” got determined by the vendor, not by the customer
And vendor now has us y the balls