Distributed Databases : Abstract + notes…

Distributed – What to expect

The next trend (hype?) seems to be : Clustered, serverless, distributed, sharded, replicated, databases. That is a lot of buzzwords together, and I have omitted “multi-cloud”, “raft” and “vector”.

We’ll try to find out how various vendors interpret those (buzz)words and how/when these are useful. We’ll explore what users, administrators and application-programmers can expect when data is stored in one of those “new and shiny” databases.

And then, when your boss tells you to “Use This”, maybe you know what to watch out for. In the end your aim is probably to keep your data Safe and Available.

Next slide is screen+intro, Put abstract here, in case you need it.
Pre-checks: docker, cluster, psqlx2, scripts, sar, load-graphs. Tables dropped., Grafana ?

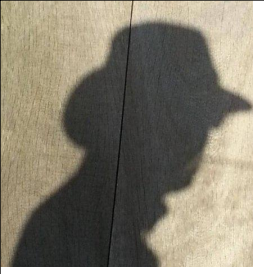
ITOUG
ITALIAN ORACLE USER GROUP

Distributed Databases – What to Expect

Piet de Visser

The Simple (Oracle) DBA

What are they and how do they behave ?
Should you use those databases ?

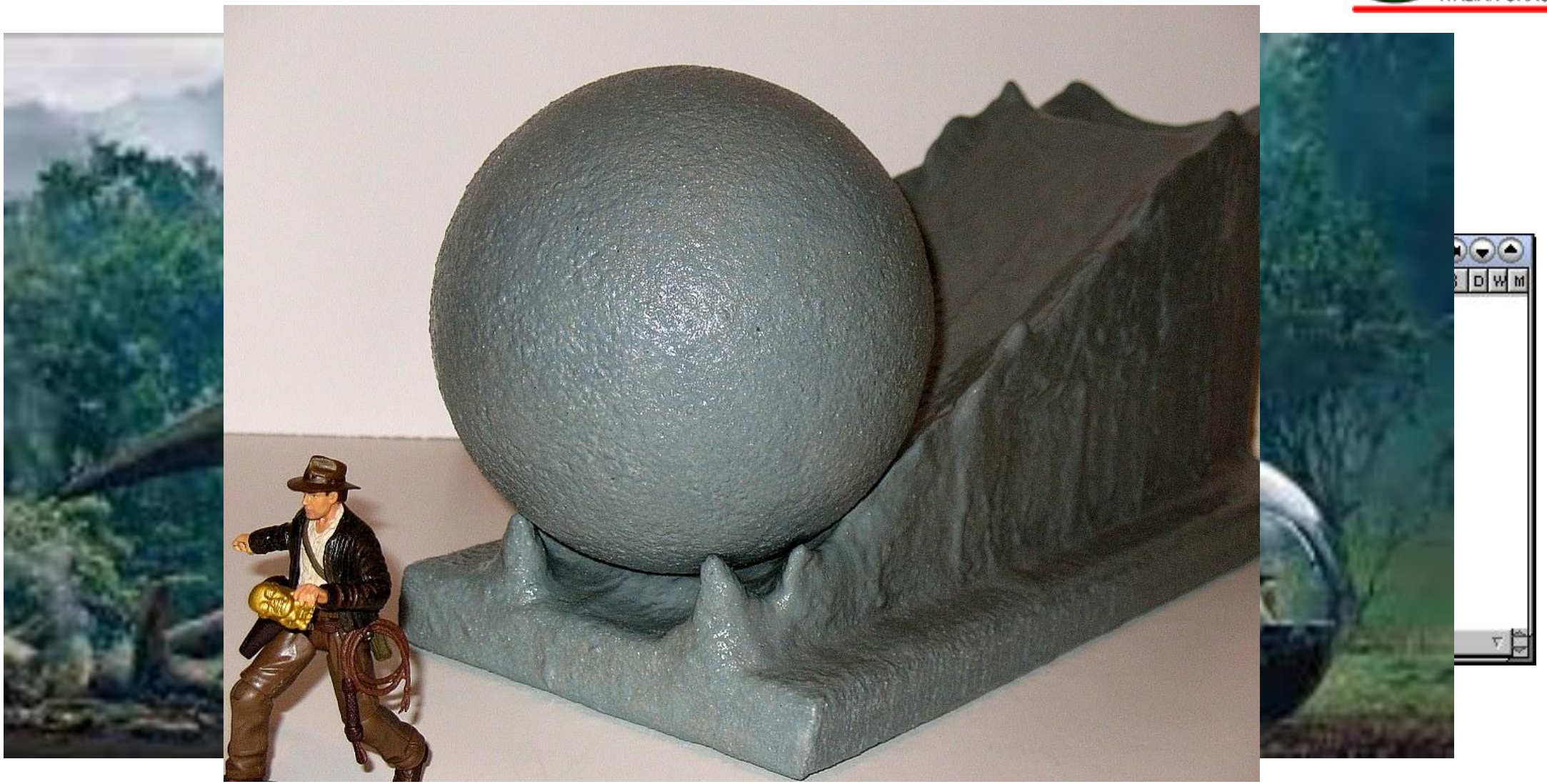


Piet de Visser - **PDVBV**

ITOUG
ITALIAN ORACLE USER GROUP

For some, I am like a Dinosour
And I spend a lot of time on that motorcycle. – need picture with Frecciarossa…

Agenda (approx 45 minutes)



Agenda. I'll try to cover a few topics and talk about 2 of the products.
List isn't complete, but I looked at those two (bcse I had time)

Distributed – the Buzzwords + ”why”.

ITOUG ITALIAN ORACLE USER GROUP

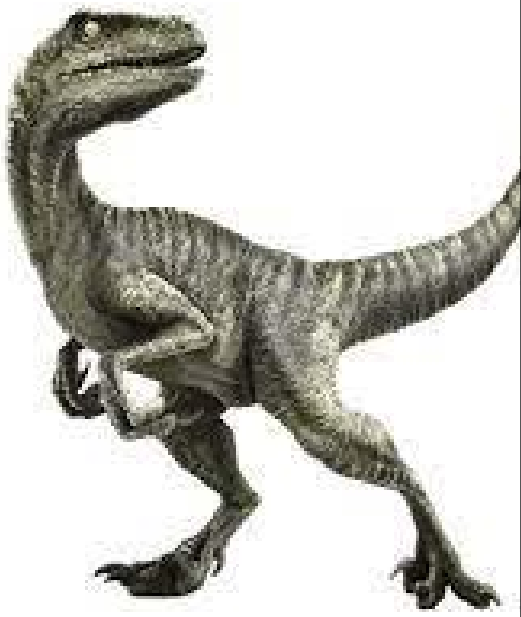
Cluster / Clustered : Using multiple machines, nodes – e.g. 42… (shared Nothing!)

Distributed: multiple machines, nodes. => “spreading out the data”

Serverless: Not using any ***specific*** servers. e.g. docker, k8s, lambdas, services

Replicated: Keeping copies of data. Mostly RF = 3 or RF = 5. Overhead !

Quorum: Majority of votes (e.g. 2 out of 3)



Buzzwords..
Get familiar with those.. But realize: Ppl may have different definitions, usage of those words.

Distributed – the Buzzwords + ”why”. 2/2

ITOUG ITALIAN ORACLE USER GROUP

Sharded:	**Divide data over …. Shards (range, hash, list…)**
Sharding-Key:	**Data, column(s) to use for Sharding.**
(=! PK)	**Sharding-Key determines “Where data Goes”**
Leader / leaseholder :	**The process in charge of an obj or data-item.**
Followers, replicas:	**The (keepers of the) copies…**
Raft :	**Algorithm, to elect and re-elect a leader.**
	Notably to be Fast and avoid race/lock conditions
	frequent heartbeats…

For Ora-DBAs: sharding is partitioning, hash, range, list.. A Shard= a partition.
and larry even inserted “RAFT” in his 23ai ppt – Reliable, Redundant, Replicated And Fault Tolerant.

[Remove Notes…]

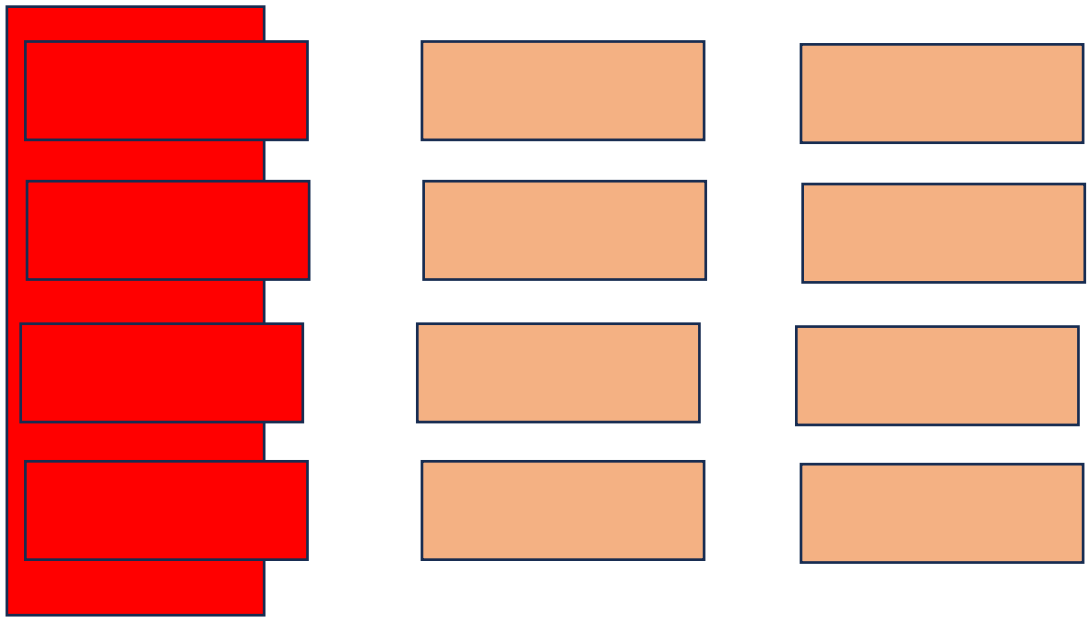
- Insert drawing of multiple servers:
- Connected, but not shared.

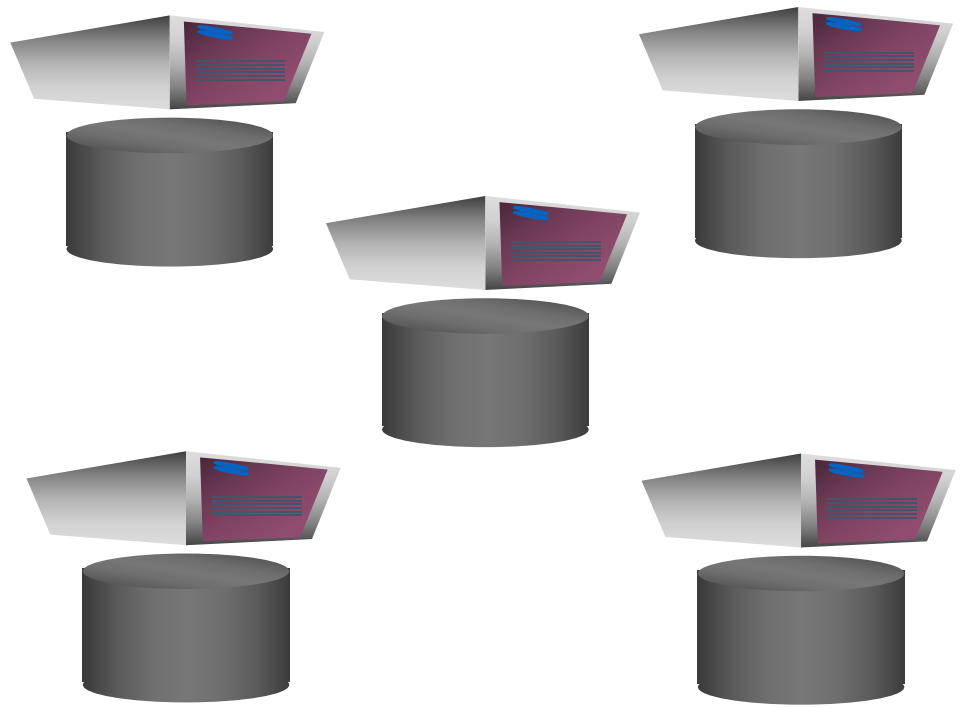
- Data is sharded over nodes
- Data is replicated for resilience..
- (ELES … need Quorum… RF=3 => N-1)

- Nature of shared nothing clusters…
- Big Q: How much can we loose… (hint: not much)

Distributed = Sharded + Replicated… 1/2

- Sharding : Split table into... Shards, then spread over “servers”
- Replicate: make copies (of shards), RF = 3…, then spread…
- Example: 4 Shards x 3 Replicas = 12 items (tablets, partitions, files)

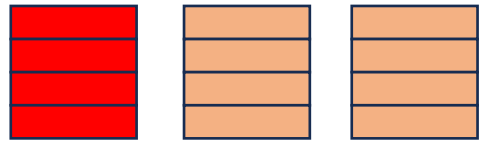


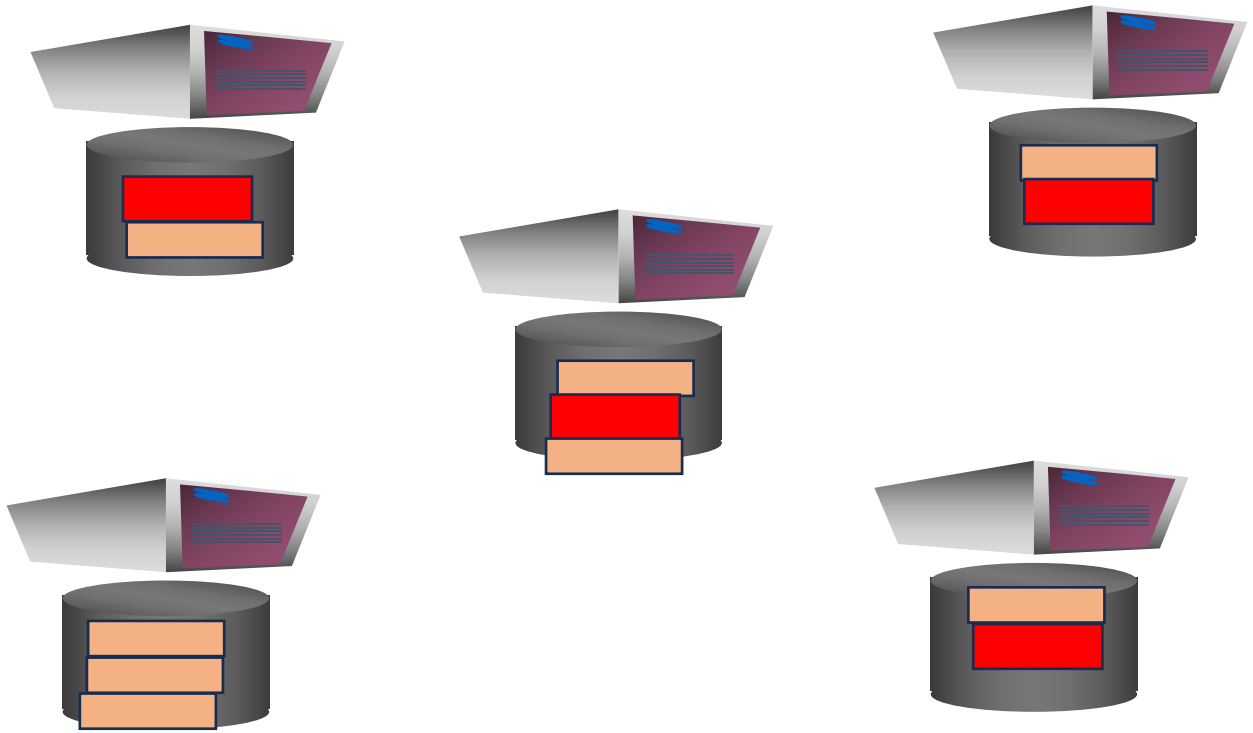


Sharding = partitioning, Replication = copies..
And then imagine spreading that over a park of servers or containers

Distributed – Sharding (spread load)

- Multiple Server (VMs, Containers)
- Sharded Data over … many.
- And RF = 3 ..



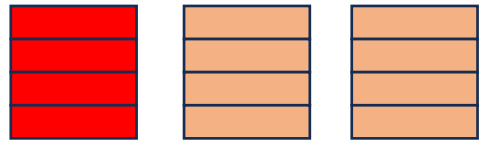


- 4 shards…
- Plus 2 copies of each
- Leaders + Copies everywhere..

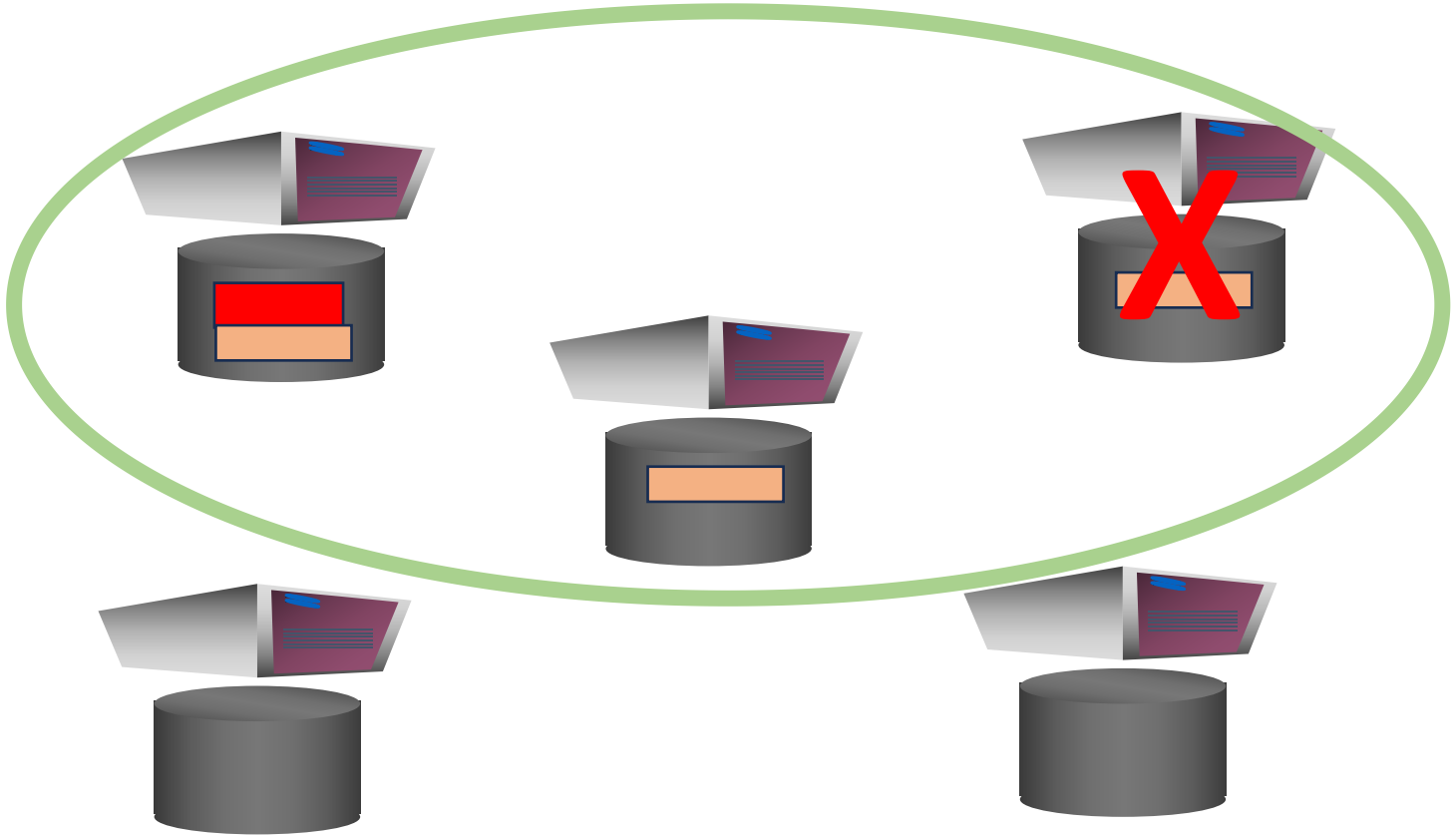
- Can afford to lose 1 (just 1)
- Any loss = Work (copies!)
- What if you loose 2 ?
- Should have had ..
- “Many” servers, still @ Risk.

With an RF=3,even with many, 42, servers, any loss of “Two” means some component loses Quorum
And any “loss of server incurs the “copy-jobs” – an un-stable cluster will incur WORK – (what about GEO…)

Distributed – Replication + Raft



- Multiple Servers, Many.. (VMs, Containers)
- Replicate Data over …
- RF = 3, every item has 3 copies on diff servers…



- Lose 1 server
- Detect loss
- [RAFT: Elect Leader if needed]
- Still Quorum (2/3)? OK!
- Copy to re-store RF=3

- What if you loose 2 ?
- Should have had .. RF = 5 !
- What if .. GEO-constraint ?

Raft : Reliable, Replicated, Redundant And Fault Tolerant…
Also Imagine: what if we Constrain shard-1 to specific GEO-location… need more servers…

Distributed – Consequences /

Distributed:	**Every Node holds Some data…**
Shared-Nothing:	**You "need all nodes"**
Storage is Not Shared:	**Hence, every node "leads" some data.**
a weakness…?	**add/remove nodes = Re-Distribution.**

IMHO: Data is not just a "Stateless app".

You can scale an "app" by adding/removing Computing power.

You can not "quickly" scale(-down) a database by removing >1 nodes…

For Ora-DBAs: sharding is partitioning, hash, range, list.. A Shard= a partition.
Re-distribution, re-sharding is "always a lot of work"..

Replication – to always be safe + available

ITOUG ITALIAN ORACLE USER GROUP

RF=3 (or 5):

Three (or 5) copies of very “item”
every shard, every partition, 3 (or 5) copies…

Raft:

On ”lost leaders”: need to Re-Elect (millisec)
A ”Shard” needs a “Master” somewhere.

Lost one node?

No problem, 2=quorum, All still works
timeout: objects are re-replicated to RF=3
Cluster is thus “self healing”

Lost 2 nodes ?

Should have had RF= …
5 or more, to keep quorum.

Re-cap of good + bad: Losing a node or container or server is not fatal.
But the object that had their leader on that node need a re-election to determine who is now in charge…

Practical implications – to be safe + available 2/3

ITOUG ITALIAN ORACLE USER GROUP

RF=3/5:	**Hardware: You need 3 (or 5)++ "Racks".** **There is a lot of Communication between…**
Storage:	**At Least 3x (or 5x)** **Most systems will do some compress…**
Processing:	**Ditto. Notably Writing (commits).** **More CPU ticks, more time.** **(even parallel-work needs "some coord.")**
GEO-Location:	**Need "Quorum" in Every Location => servers++!** **+ Latency will hit you… (more comm…)**

Writing 1 item to 3 places can be done partly in parallel, but it still is 3x the work and it needs some Coordination and has to wait for Quorum)

Practical implications – Resilience. But Beware 3/3

ITOUG
ITALIAN ORACLE USER GROUP

Does it work?

Hell Yes!
Just dont loose 2 or more nodes (15-20min…)
Just keep distance/latency Low.

Bonus Question:

Losing 2 nodes at once, out of 42 …. ?

Only “some shards” are affected.
Only those that lost “quorum”
(but generally Not Acceptable!)

Each vendor has (more) tricks…

Resilience is good. Some architects take RF=5 to be safer. And each system has other strategies to Survive node-loss, such as read-copies of shards, or even replicated-clusters (e.g. more copies…)

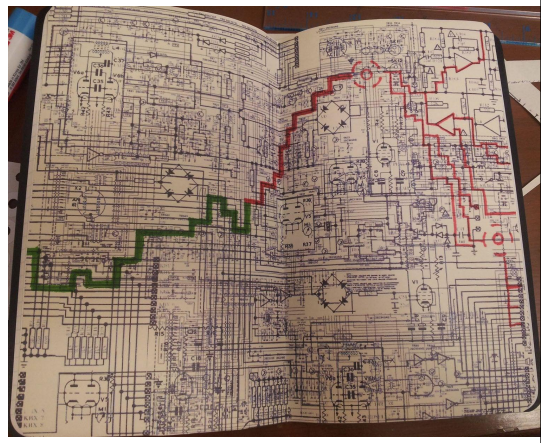
Storage: LSM and SST

ITOUG
ITALIAN ORACLE USER GROUP

LSM: Log Structured Merge (memory)

SST: Sorted String Table (file)

What/Why: Efficient use of Memory and Disk
Write-Once - Append Only.
Compressable (compaction) = Nice!



Implications… (?) Disk usage growing … and …Shrinking!
(for now: assume >3x space, RF + …)

Need to investigate, and YMMV!

Optimized for "write", and for key-value. Coming from "big data" applications. Impact on OLTP and RDBMS still unknown (to me). Suggest : Testing + Thknking… (surprises in init-load, surprises in range-scans..)

Optional Intermezzo: Poll for audience

ITOUG
ITALIAN ORACLE USER GROUP

Thank You !

(stay tuned)

Find out from the audience.
Bonus question: who prefers init-files ?

Serverless – Why?… 1/2

ITOUG ITALIAN ORACLE USER GROUP

Serverless	**Not striktly (you can deploy on a server)**
Dev:	**Developers prefer containers… (duh)**
Arch:	**Some shops “Require” it (k8s only…)**
Scale-Up:	**Re-deploy (restart) on bigger box. Works fine.**
Scale-Out:	**Deploy to more containers...**
	Not immediate: need re-distribution of data.
Scale-Down ?	**Less nodes ?**
	(not ideal…..)

Serverless is a trend. Everyone wants to be on k8s or docker.. - Need drawing of scale-up-box…
Just make sure you know where your data is (which storage it sits on)

Serverless – Why?… 2/2

ITOUG ITALIAN ORACLE USER GROUP

Scale-up/down	Not Stateless … (a DB is Not a Micro-service) Re-Distribution = WORK (not ideal)
Scale-Back	Better: re-deploy (restart) on smaller “nodes”
Note:	We once thought VMs unsuitable for RDBMS…
IMHO:	Abstraction, Layers ? Complexity ?

Data should have least possible layers of processing,
Try to deploy ”On the Server”, or “on Iron” (says the dinosaur).
Many shops/clouds: Containerization is a Must… ?

Serverless is a trend. It Will Happen. Everyone wants to be on k8s or docker.. (currently CPU seems bottleneck?)
Please make sure you know Where your data is (which storage it sits on), and how your bck/restore works.

Geo-Sharded – Legal or Latency…

Geo-Located:	**Each data item (record) has a “home”**
Legal ?	**Data ”must reside” at location (?)** **But still see it as One Single Database.** **Example: Medical records, Tax-records.**
Latency !	**Data close to user** **But still see it as One Single Database.** **Example: Superbowl-tickets, YB showcase.**
Quorum ?	**RF=3: nr of servers++ in Each Location.**

IMHO: “Legal requirement” will never be “Waterproof”

Both YB and CR can place data in specific, defined, locations (e.g. list-sharding or reange-sharding)
I don’t believe the Legal-requirement can stand solid auditing… But vendors will think different.

Errors made… (so you don't have to…)

ITOUG
ITALIAN ORACLE USER GROUP

Sharding: Too Many shards… (20 regioni, 107 province)
Check the defaults (42 servers… ???)

Sharding Too Few shards => re-sharding starts
Big-Data? - Prevent re-sharding on bulk-load.

Sharding: Hash-sharding on Date…
Hash-sharding on name-lookup

IMHO: Sharding is a Good Mechanism,
but You Need to Know "What happens under the hood"

Bcse engineers come from "big data", they tend to overlook the demands and quircks of OLTP / RDBMS
Notably sharding is no silver bullit. E.g. collocate, and seq-cache, and pre-allocate in CRDB

Errors made… (so you don't have to…)

ITOUG
ITALIAN ORACLE USER GROUP

Sequence:	**Cache !**
Nodes on/off	**Outages, No Problem…** **Short (<15min): node can/will come back.** **Long (>15min): Data will redistrib !**
Storage:	**No clear Errors… Disk Full !**
CPU/Mem:	**No Metrics ! (go see Franck)**

IMHO: There is a lot to Learn still,
notably in Metrics "under the hood"

Deploying the new type of systems, count on learning time.. Storage will grow+shrink, mem/cpu unknown, And "observing" multiple nodes is a new art-form. (SQL would help !)

Main Messages – What we Learned…

ITOUG
ITALIAN ORACLE USER GROUP

This will happen (dino: who needs it?… not relevant..)

Observe, Learn, Think, Measure.

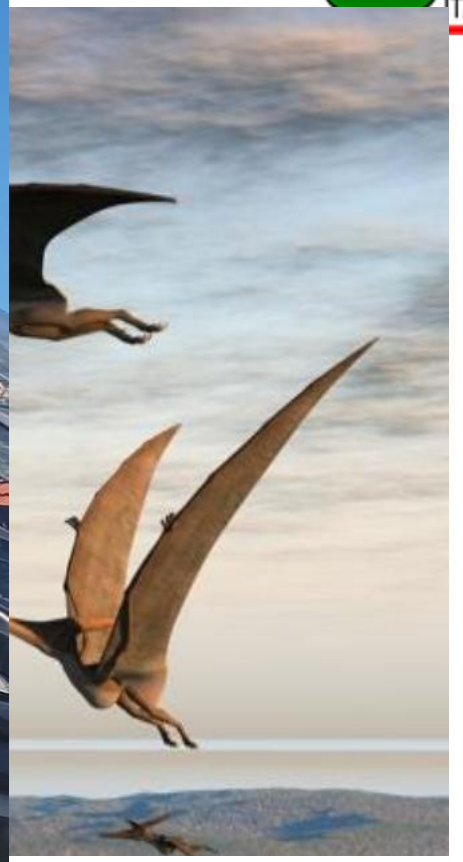
Sharded: range or hash.. (date ranges…)

Replicated: how + where (Sequences!)

Impact of LSM + SST: need to learn…

Go out and Experiment !

Ask for watch, tell time.. At least when open source: you can read the code.. (hmm)
Need to Experiment, discuss, report… Clever Dino from Jurassic park…



Long term: DB-aaS.. And “serverless” …
Beware of the learning-curve ahead… (dinos are still here, and picture of stork, for luck)

Don’t Take my word for it…

Try for Yourself… and tell me next conference.

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

- **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.

This is POUG... Let's Demo !

ITOUG
ITALIAN ORACLE USER GROUP

Demo_dd.sql : overhead of (many)shards…

Mk_longt.sql : insert 200M, and check tablesize…
- use lots of space
- compression can help, automatically

Do_fill.sh : run process and kill nodes…

Chk sizes again to see compression.

Demos depend on time..
Time.

Blank

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

Distributed – Where does your data go?

- Multiple Server (VMs, Containers)
- Sharding : split table into... Shards.
- Distribute Shards over … servers/containers