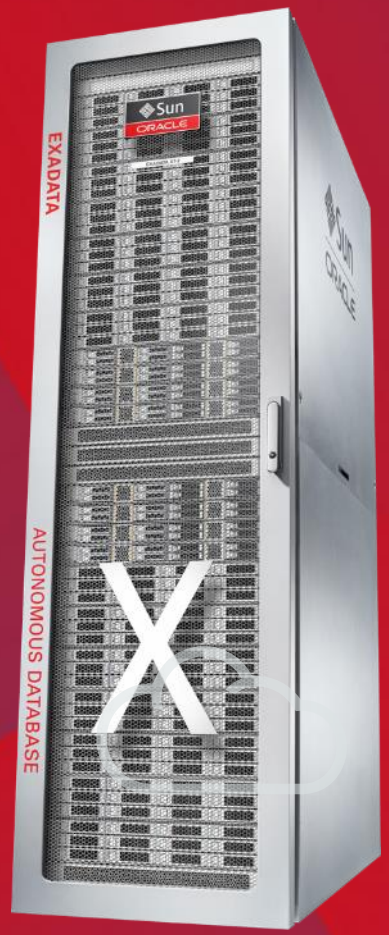
Exadata公开课（二）

RDS & Exafusion分布式缓存一致性加速
Smartflash log加速Redo写

扫码加入:
19c新特性讲座群

欢迎关注:
甲骨文云技术公众号



ORACLE

目录

- 基于缓存一致性的分布式数据库架构回顾
- Exadata中针对分布式缓存一致性的优化
- 专门针对数据库优化Smartflash Log

支持关键业务应用数据库的核心需求

采用分布式的驱动力

#1 - 可用性

“站点的性能和可用性是我们业务最重要的两块基石”

- Director of Engineering, Large Online Social Networking Customer

#2 - 响应时间

-“如果一次搜索的处理时间超过一秒钟，用户会转去使用别的竞争对手站点，我们会失去用户的信任同时会失去一个销售机会”

- Sam Peterson, SVP, Technology, Overstock.com

#3 - 业务负载的不可预见性

-“我们预测到在特定时间流量会超过系统的设计能力，但是我们不能确知最终会达到多少”

- IT Director, Large Online Retailer

#4 - 业务成长潜力

-“随着越来越多用户使用我们的在线服务，我们要求目前使用的技术架构必须能够提供流量的不断增长”

- Eric Grosse, President of the Hotwire Group

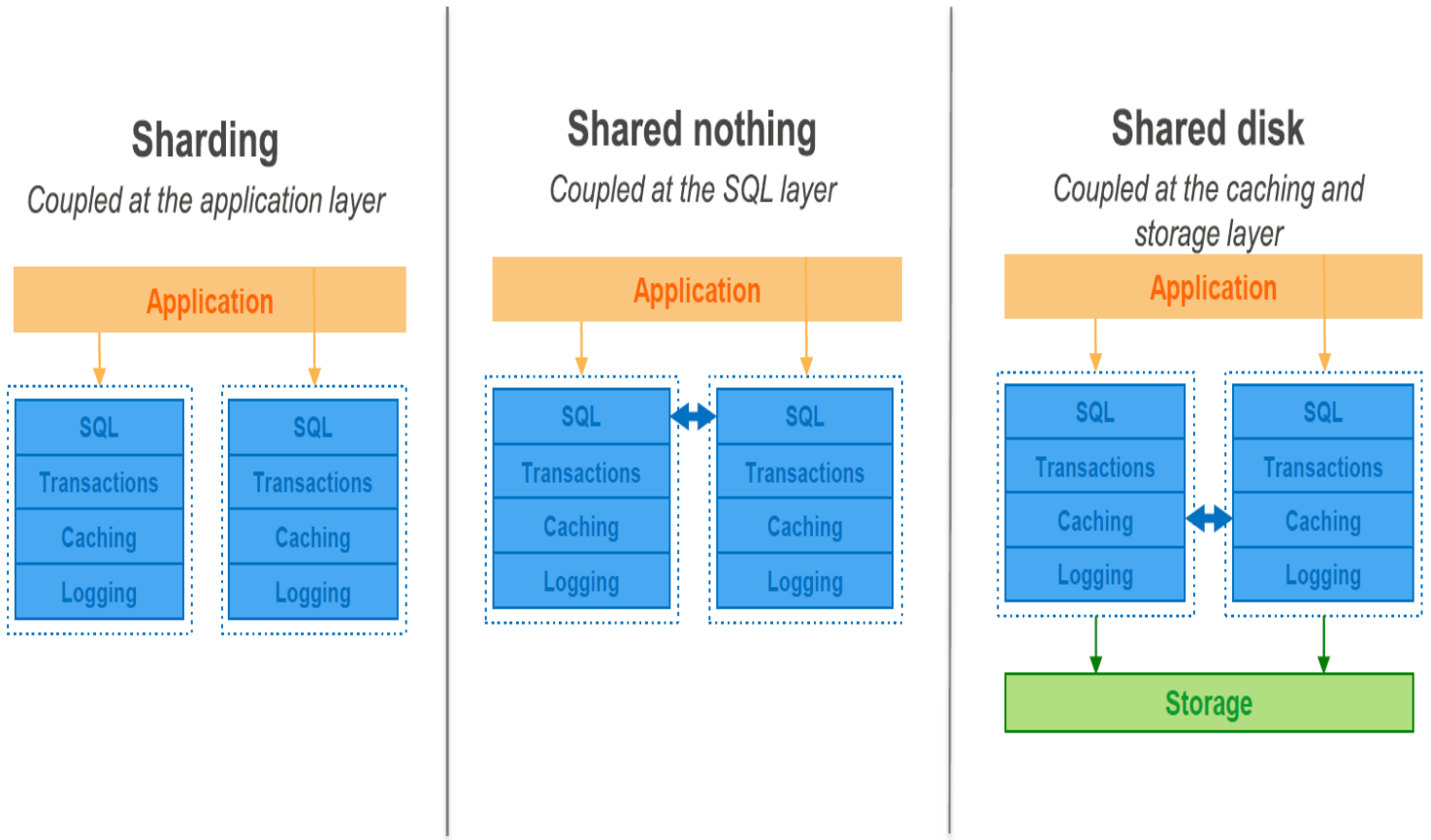
#5 - 有效帮助缓解成本压力

-“业务要求我们必须支持不断增长的流量，同时还要在保持预算不增加的前提下保证业务的SLA”

- VP of IT, Online Betting Company

#6 - 易于管理

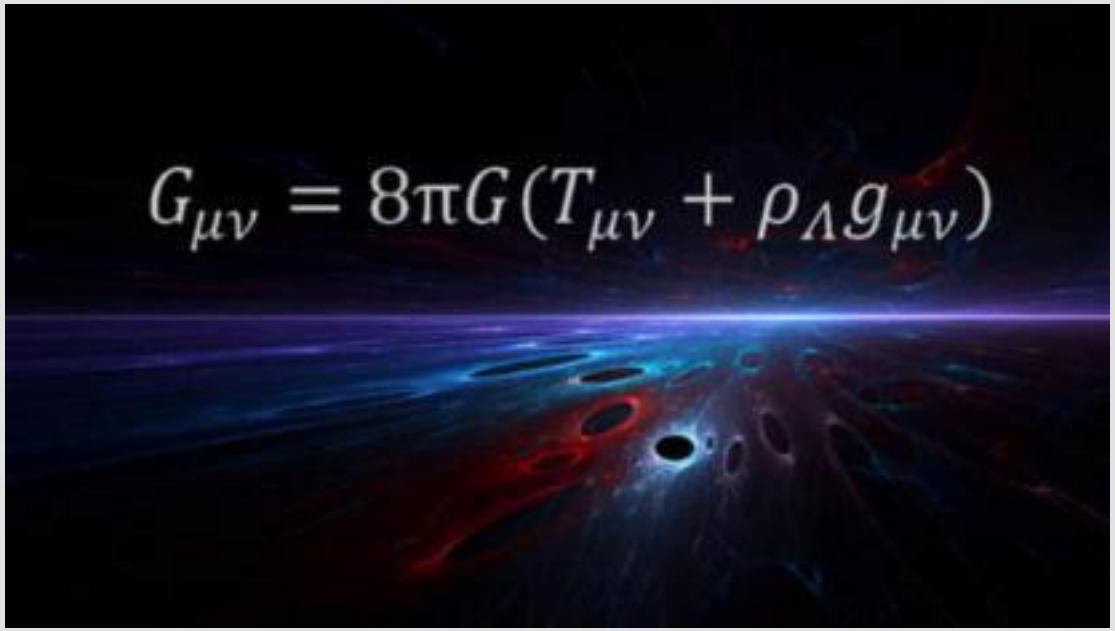
三种“分布式”数据库的架构



Even when you scale it out, you're still replicating the same stack

- 分布式数据库是用计算机网络将物理上分散的多个数据库单元连接起来组成的一个逻辑上统一的数据库。每个被连接起来的数据库单元称为站点或节点。
- 分布式的技术难点不在于“分”，而在于“合”
  - 分布在不同节点的数据存在相关性
  - 如何对应用层提供一个与单机版的集中式数据库一致的体验才是关键

分布式数据库受限于基本物理原理



广义相对论：任何信息传递速度不可能超越本地坐标真空中光速

$$C = W\log(1+\frac{s}{n})$$

其中：

C：信道容量

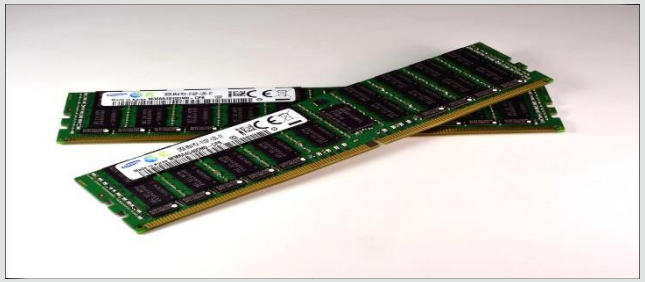
W：频带宽度

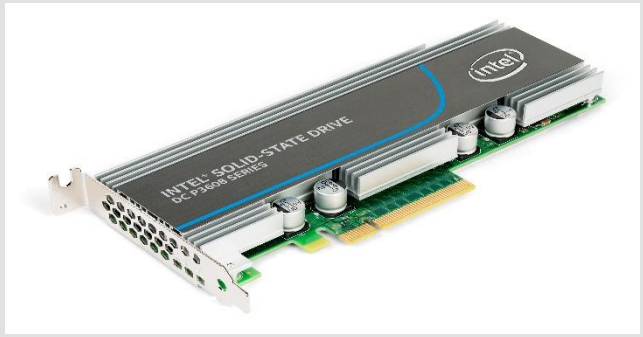
$\frac{s}{n}$：信噪比

r_n（dB）=0.4dB/Km×L_N(Km)

- 传输距离越长由于衰减和色散必须再生导致延迟大量增加增加
- 单根光纤传输容限受到1）自发辐射（ASE）噪声和2)非线性损伤的限制。需要强调的是，可以认为传输容限随着传输距离增大而减小。

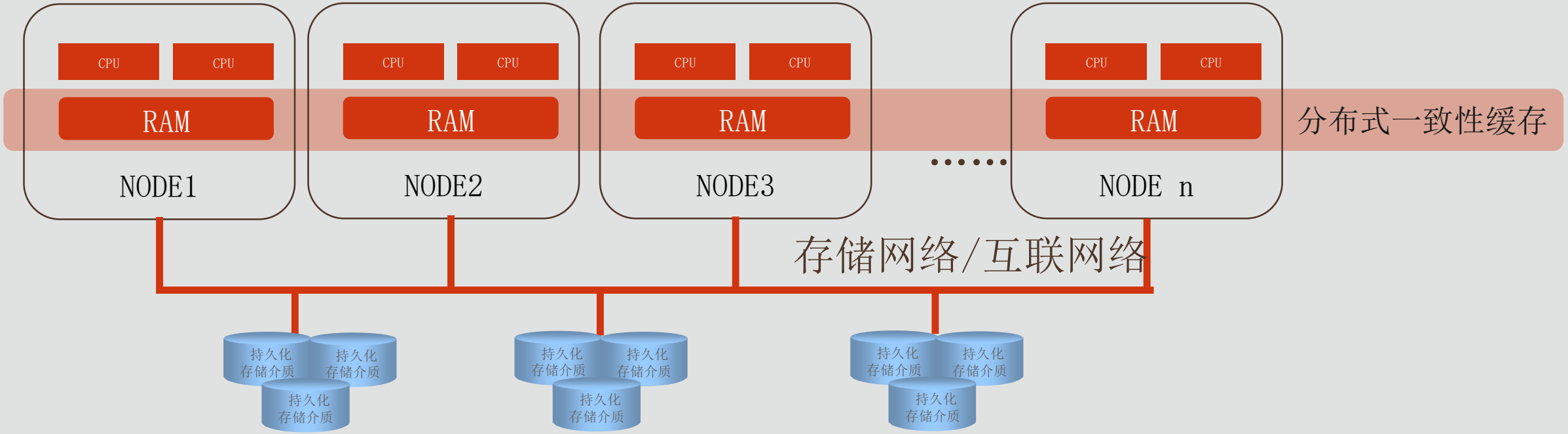
计算与数据：数据库要快速必须依赖的计算机基本原理





- 在冯.诺依曼和哈佛架构下数据离计算越近计算效率就越高
- 但是在怎样将数据离计算近却产生了两种完全不同的approach
  - 利用Cache，让数据缓存在离计算更近的地方
  - 将计算能力放置在离数据更近的地方
- 实际面向现实数据库计算的情况
  - OLTP有相对较小的热数据（SGA buffer cache）
  - OLAP的热数据就相对较大(Direct path read)

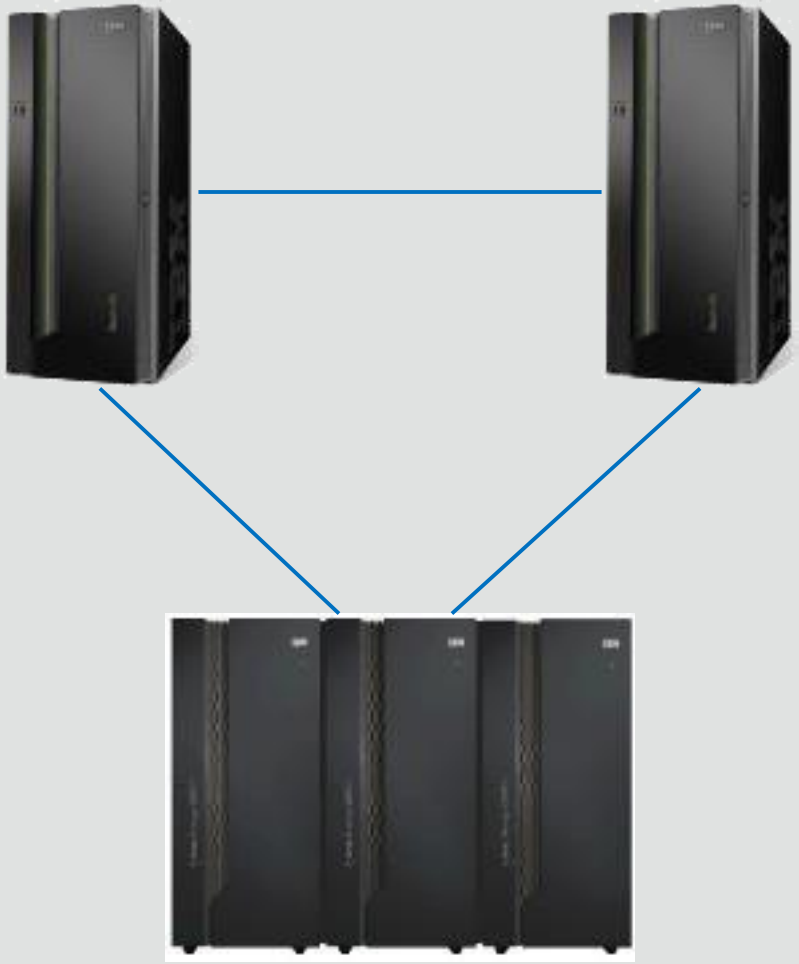
分布式数据库系统架构：缓存一致性分布式，share disk



Shared-Disk架构特点

- 需要的是共享存储（计算存储分离），并不需要集中式存储
  - 分布式共享存储实际上很容易实现：ASM + Linux TGT即可
- 仅仅在存储层耦合是不够的，要提升性能必须在缓存层进行数据耦合并

传统架构的Oracle RAC横向扩展难题



- 受限于以太网的带宽、延迟以及数据访问方式，以及具体实施，RAC多节点扩展在传统架构上很难取得良好效果

技术	访问延迟
CPU寄存器	<1ns
CPU Cache	1-50ns
本地内存	~50-100ns
NUMA架构非本地内存	~100-300ns
以太网（UDP互联）	~1ms

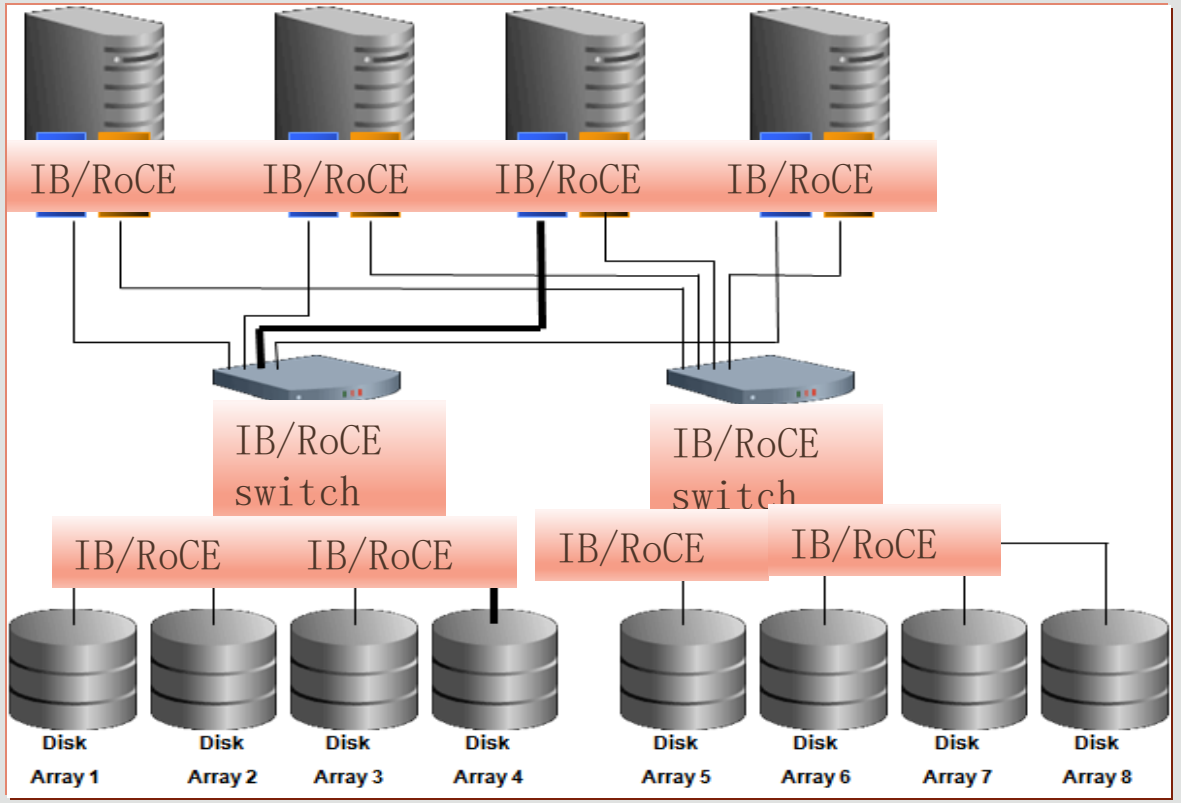
  - 在实践中容易受限于2节点，多节点需要应用分割
  - 处理能力达到上限后替换式服务器扩容
- 共享盘阵的架构受限于SAN存储的扩展能力
  - Log file sync等待放大GC相关等待
  - IO能力达到上限后铲车式替换扩容
  - 即便最高端的SAN存储也难以以驱动当代服务器的计算能力

目录

- 基于缓存一致性的分布式数据库架构回顾
- Exadata中针对分布式缓存一致性的优化
- 专门针对数据库优化Smartflash Log

Exadata思路：提升带宽，降低延迟，减少传输，减少软件栈调用层次中断调用

- 减少需要通过网络传输的数据量，尽可能数据就近处理
- 减少软件栈层次，甚至直接绕过OS系统调用操作硬件
- 解决硬件带宽瓶颈，使用新的硬件架构例如IB， RoCE



Exadata架构:为Oracle RAC提供独特的分布式数据库架构云平台

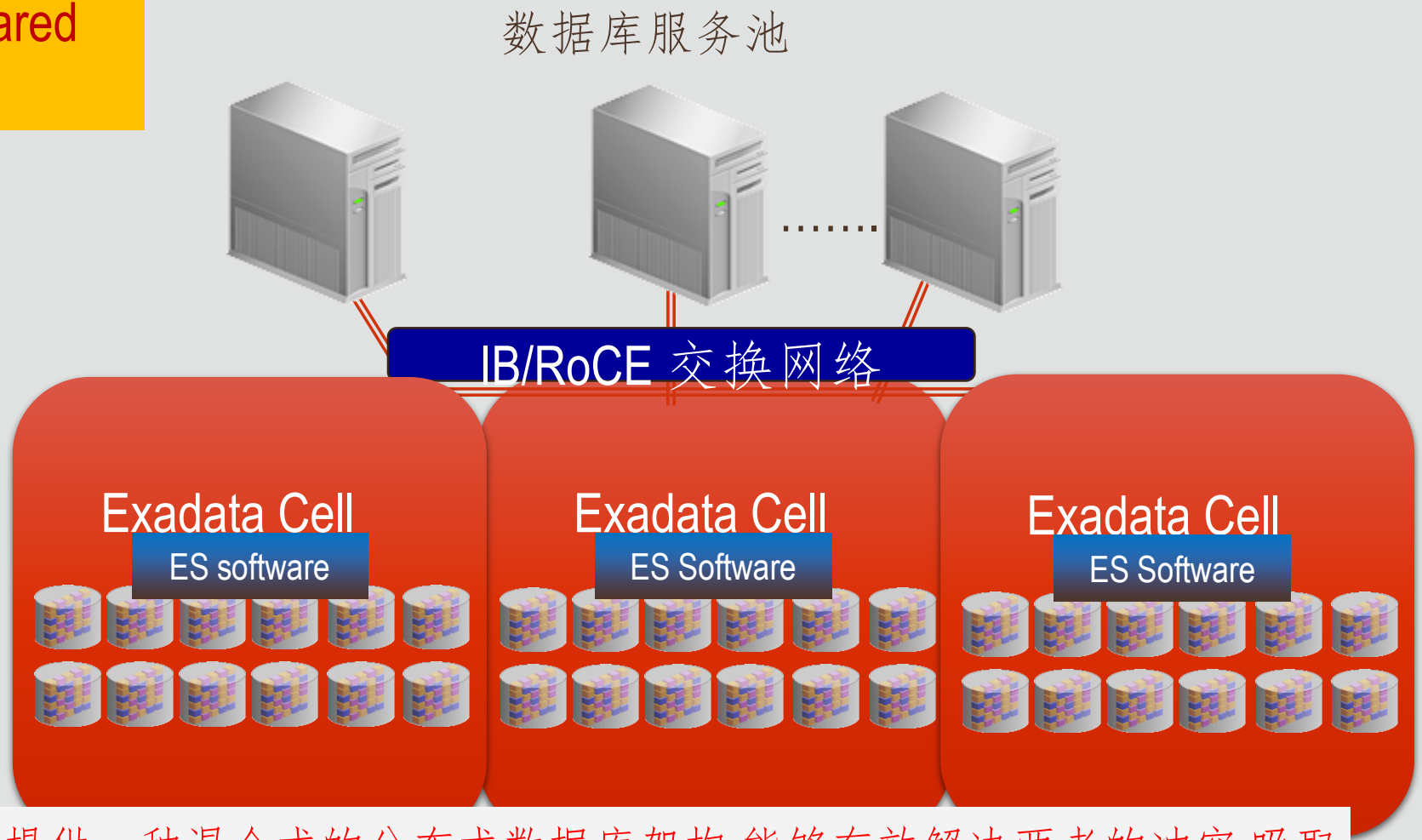
兼备Shared Nothing 、Shared Disk优点，应用透明

具备良好的普适性架构:Shared Nothing and Shared Disk

数据库处理层(Shared Disk)
数据库资源池

硬件RDMA网络层
IB/RoCE互联，低OS介入通信

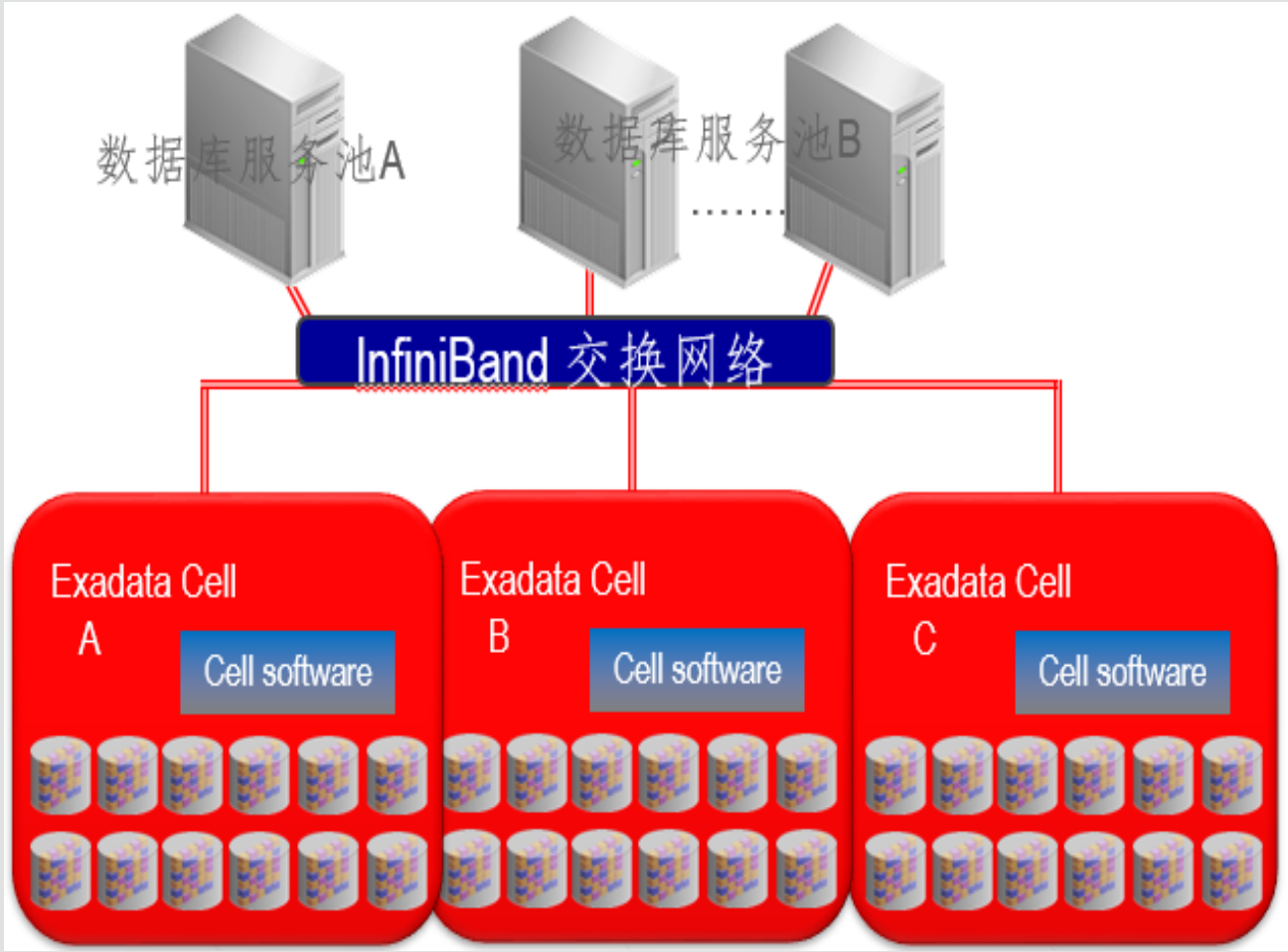
智能存储层(Shared Nothing MPP)
存储资源池



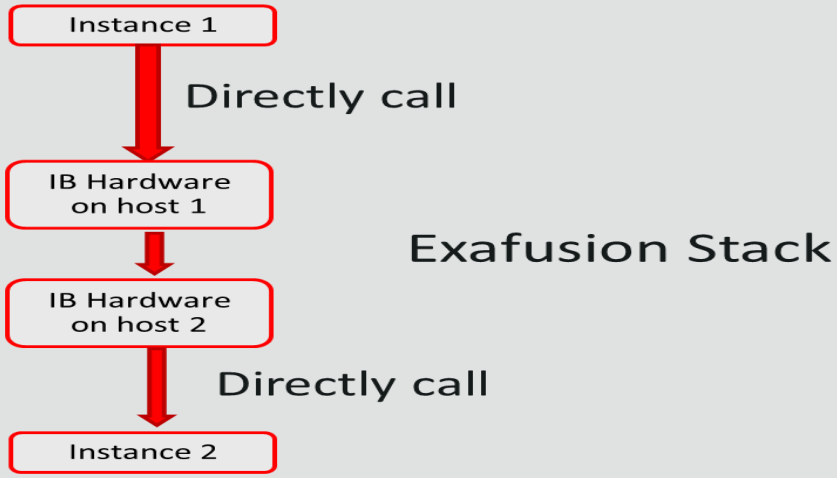
- Exadata是目前业界唯一一种提供一种混合式的分布式数据库架构,能够有效解决两者的冲突,吸取两种架构长处;既可以满足OLTP的高并发、高可用特点；又可以满足OLAP的大数据量处理要求

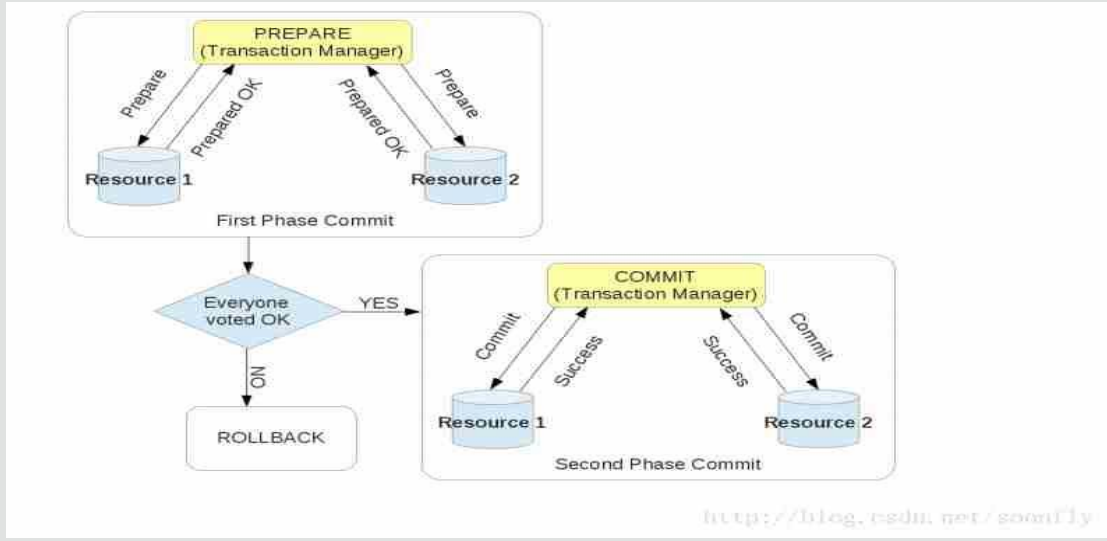
RAC Over Exadata混合分布式数据库特点

- 缓存一致性的RAC Over Exadata，实现应用完全透明的分布式处理
  - 没有分布键key要求，没有主键强制要求
  - 语句查询没有限制
  - 应用透明的扩容、缩容
  - 10年前第一代产品就实现了计算存储分离
  - 8年前就实现了成熟的混合负载(HTAP)
  - 单个会话内部事务跨节点依靠缓存一致性，不需要XA两阶段提交和锁来保证
- 存储层share nothing处理
  - 大数据量数据就近处理
  - 降低网络传输流量



分布式一致性缓存 VS 分布式事务两阶段提交





现代硬件和基础软件进化：RDMA实现低延迟整块内存同步

完全符合计算加速基本原理充分利用cache

基于操作级别的latch

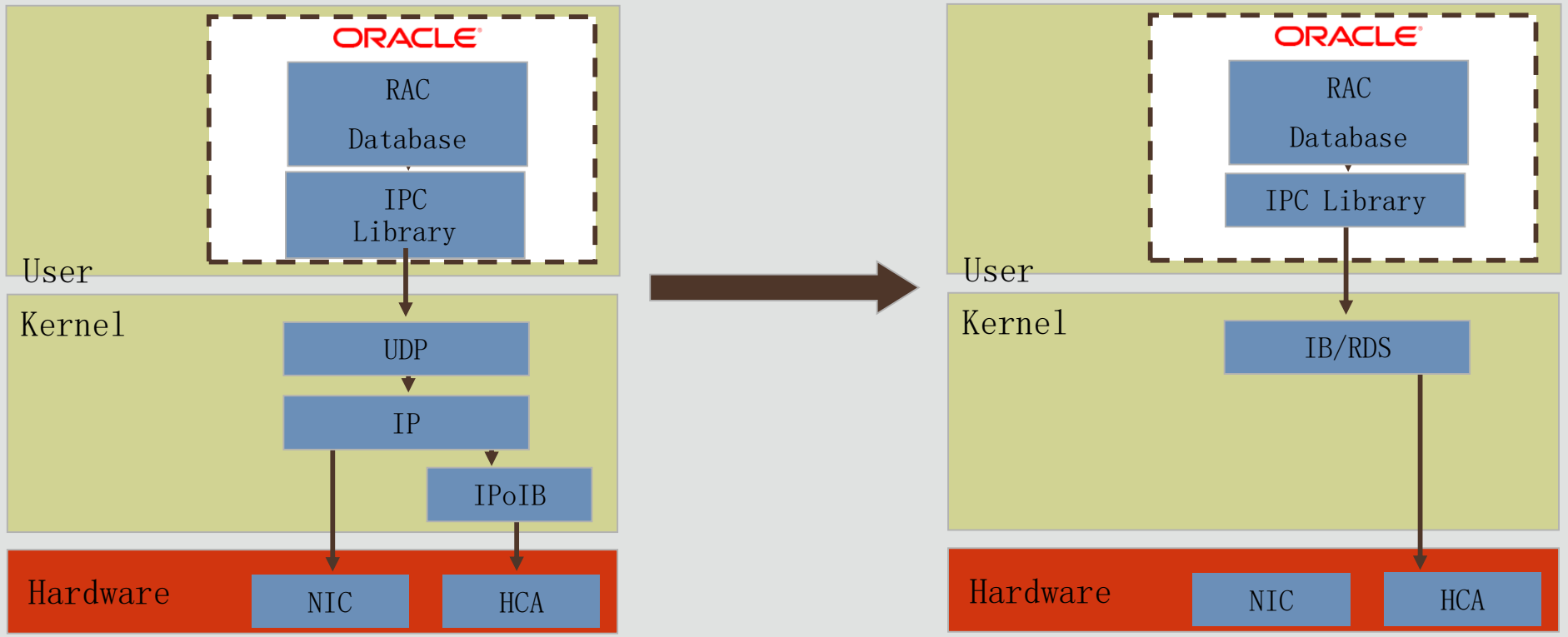
本地会缓存可用数据块

现代硬件和基础软件对于2PC传输的SQL未见任何革命性加速功能

即便只是查询，基于SQL语句传递无法充分利用本地Cache

基于事务级别的锁（lock），必须事务提交或者回滚才能释放

解决多节点扩展的高性能网络



不仅仅是硬件带宽和延迟的提升：1Gb/10Gb vs 40Gb/100Gb， 50us vs 1us

更重要是协议的提升： RDS(针对200字节-8MB数据优化） VS UDP（受限以太网MTU，通常针对<1000字节）

采用IB硬件但是用IP Over IB并不能充分发挥IB硬件性能

X86环境Oracle只支持RDS运行在Oracle HCA上；从12.2开始RDS协议只能在Oracle ES打开

仅仅使用IB硬件并不能加速Cache Fusion

10GE

Event	Waits	Total Wait Time (sec)	Wait Avg(ms)	% DB time	Wait Class
DB CPU		128.5K		81.2	
SQL*Net more data from client	4,247,609	8360.1	2	5.3	Network
db file sequential read	14,584,132	7988.7	1	5.0	User I/O
gc cr grant 2-way	6,424,500	3835.4	1	2.4	Cluster
gc cr block busy	1,081,753	3542.3	3	2.2	Cluster
gc current block 2-way	4,158,184	3235.9	1	2.0	Cluster
log file sync	1,249,782	3058.1	2	1.9	Commit
gc cr block 2-way	2,419,872	2465	1	1.6	Cluster
gc buffer busy acquire	930,696	2057	2	1.3	Cluster
gc cr multi block request	1,158,487	1813.5	2	1.1	Cluster

Interconnect Ping Latency Stats

- Ping latency of the roundtrip of a message from this instance to
- target instances.
- The target instance is identified by an instance number.
- Average and standard deviation of ping latency is given in miliseconds
- for message sizes of 500 bytes and 8K.
- Note that latency of a message from the instance to itself is used as
- control, since message latency can include wait for CPU

Target Instance	500B Ping Count	Avg Latency 500B msg	Stddev 500B msg	8K Ping Count	Avg Latency 8K msg	Stddev 8K msg
1	287	0.27	0.04	287	0.26	0.04
2	287	1.05	3.12	287	1.20	3.13

Back to Interconnect Stats
Back to Top

Interconnect Throughput by Client

- Throughput of interconnect usage by major consumers
- All throughput numbers are megabytes per second

Used By	Send Mbytes/sec	Receive Mbytes/sec
Global Cache	20.20	24.88
Parallel Query	0.00	0.03
DB Locks	4.31	3.21
DB Streams	0.00	0.00
Other	0.01	0.09

UDP over IP Over IB

Top 10 Foreground Events by Total Wait Time

Event	Waits	Total Wait Time (sec)	Wait Avg(ms)	% DB time
read by other session	3,102,429	35.5K	11	69.0
DB CPU		10.4K		20.2
gc buffer busy acquire	445,367	2959.4	7	5.8
db file sequential read	1,442,107	901.6	1	1.8
gc cr multi block request	229,463	815.5	4	1.6
db file scattered read	278,984	794.8	3	1.5
latch: cache buffers chains	2,270,845	218.7	0	.4
reliable message	255,564	143.5	1	.3
gc cr grant 2-way	494,683	95.1	0	.2
gc cr disk read	539,345	93.4	0	.2

Interconnect Ping Latency Stats

- Ping latency of the roundtrip of a message from this instance to
- target instances.
- The target instance is identified by an instance number.
- Average and standard deviation of ping latency is given in miliseconds
- for message sizes of 500 bytes and 8K.
- Note that latency of a message from the instance to itself is used as
- control, since message latency can include wait for CPU

Target Instance	500B Ping Count	Avg Latency 500B msg	Stddev 500B msg	8K Ping Count	Avg Latency 8K msg	Stddev 8K msg
1	288	0.43	0.16	288	0.41	0.15
2	288	0.44	0.15	288	0.47	0.15
3	288	0.44	0.15	288	0.47	0.15
4	288	0.39	0.15	288	0.42	0.15

Back to Interconnect Stats
Back to Top

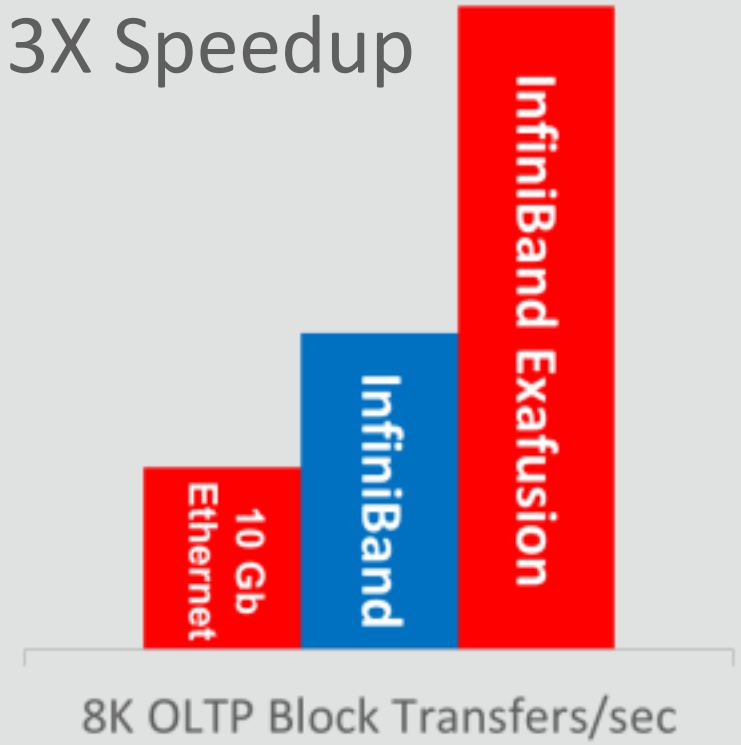
Interconnect Throughput by Client

- Throughput of interconnect usage by major consumers
- All throughput numbers are megabytes per second

Used By	Send Mbytes/sec	Receive Mbytes/sec
Global Cache	0.80	0.48
Parallel Query	0.00	0.00
DB Locks	4.06	1.91
DB Streams	0.00	0.00
Other	0.05	0.05

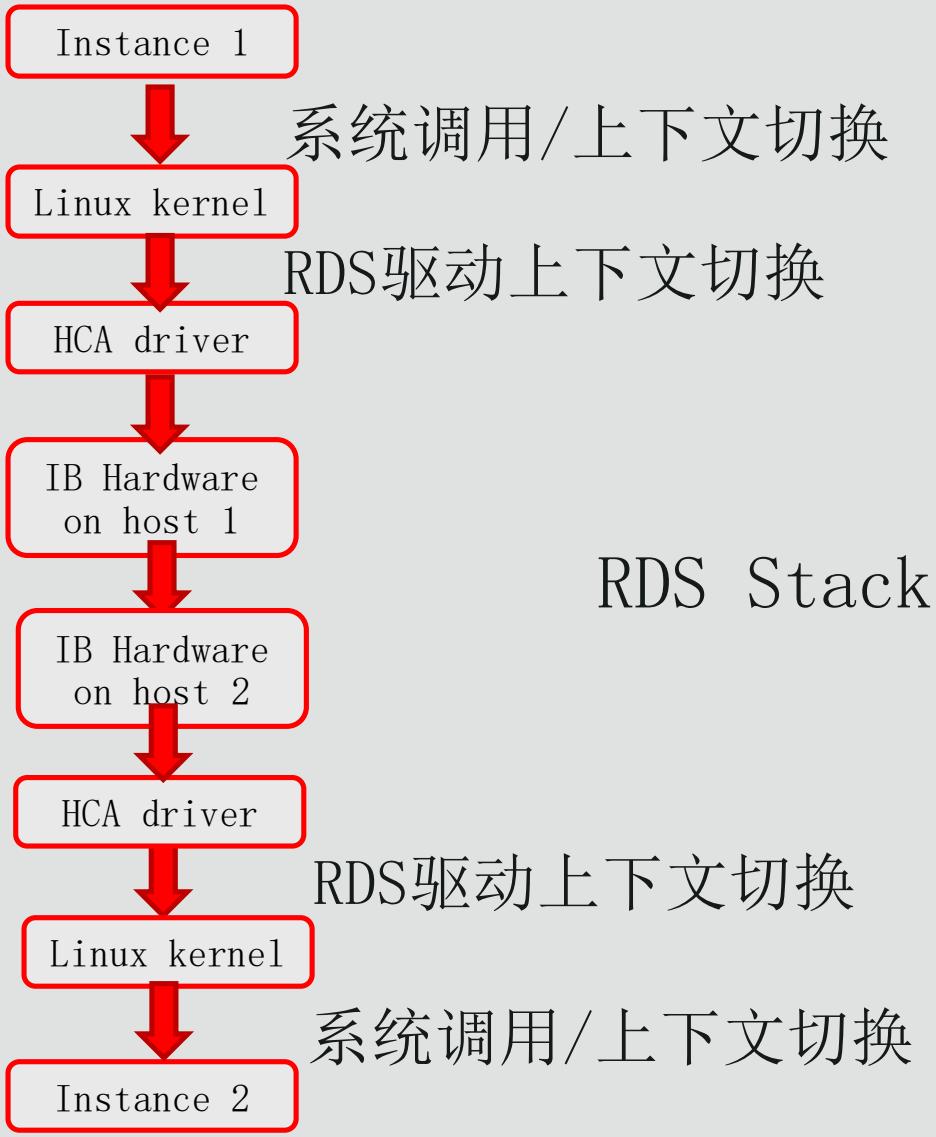
只有在Oralce数据库云平台支持的多节点扩展协议

Exafusion 直接通信OLTP协议

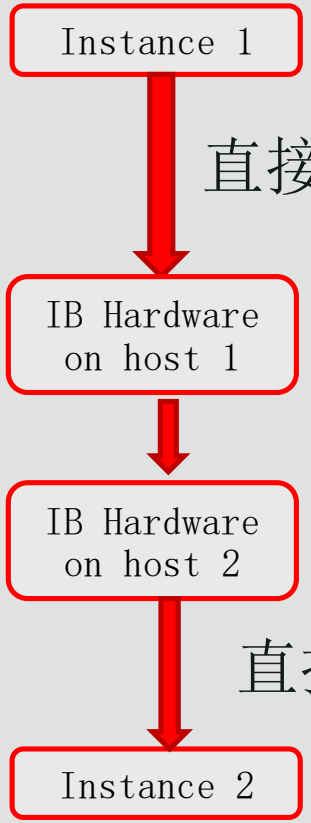


- InfiniBand有很高的吞吐量
  - 但是OS网络协议栈的额外开销导致了小消息传递效率不高
- Exafusion 重新实现了 RAC Cache Fusion
- Database 直接调用InfiniBand 硬件
  - 绕过传统网络软件协议栈，中断调用和OS调度
- 支持除V1和V2以外的所有Exadata硬件
- Oracle 12.2或以上版本在Exadata自动启用

Exafusion和RDS软件协议栈区别



RDS Stack



Exafusion Stack

18c以上在Exadata上RAC进一步的性能增强

新的 RDMA算法减少消息、延迟和CPU消耗

Undo 块的 RDMA读

在某些工作负载里面，超过一半的远程读Undo块以保证读一致性

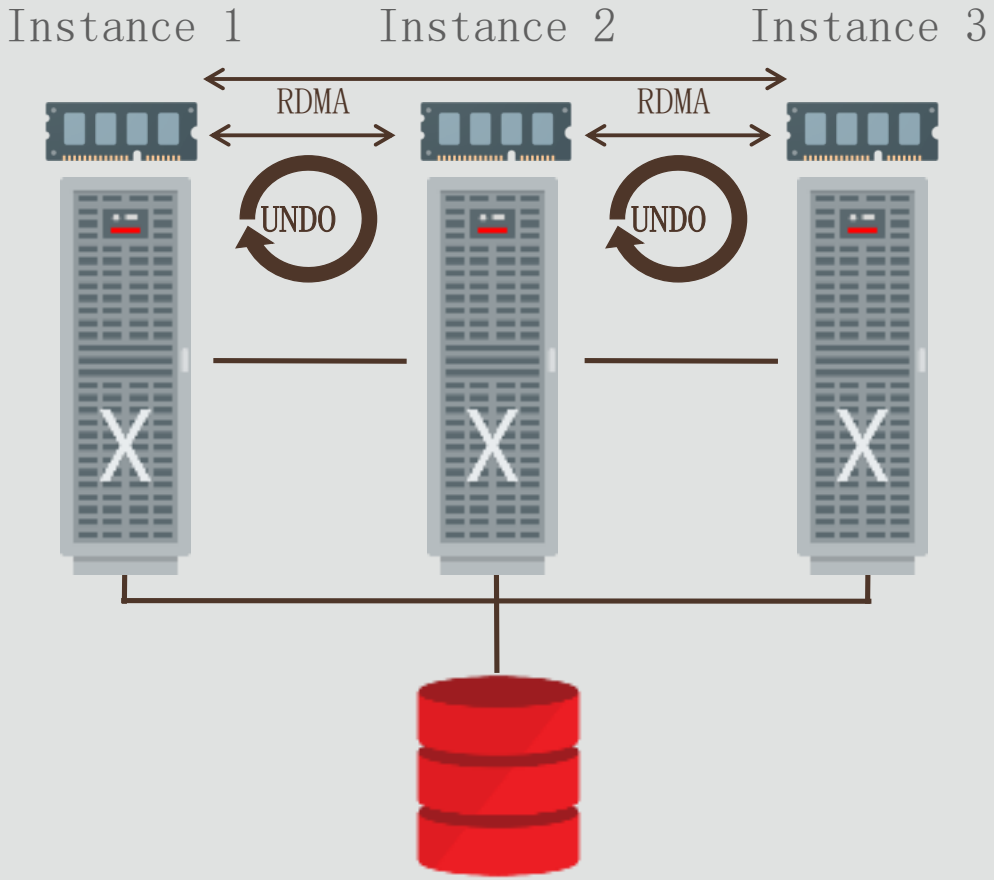
Undo 块 RDMA-读采用直接RDMA调用快速的从远程实例获取 UNDO块

Commit Cache

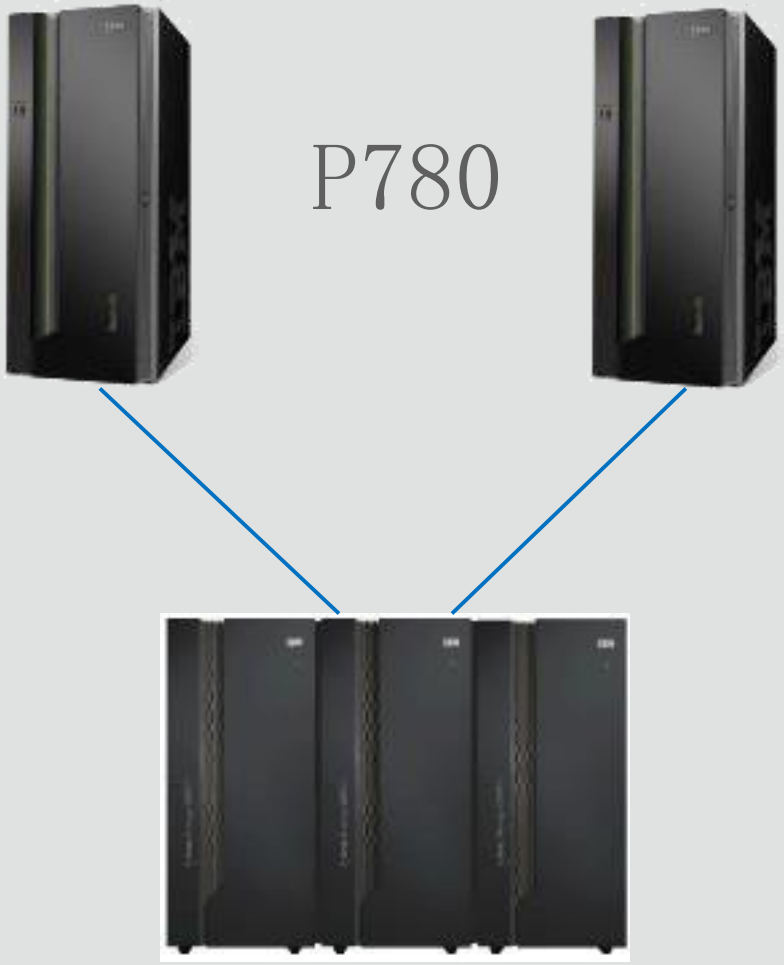
Commit Cache通过在每个实例维持一个内存表记录交易的提交时间

远程的LMS进程为请求的交易直接通过RDMA读commit cache并且获得提交时间

替换原有需要传输整个8K的数据块的方式



案例：某客户核心OLTP

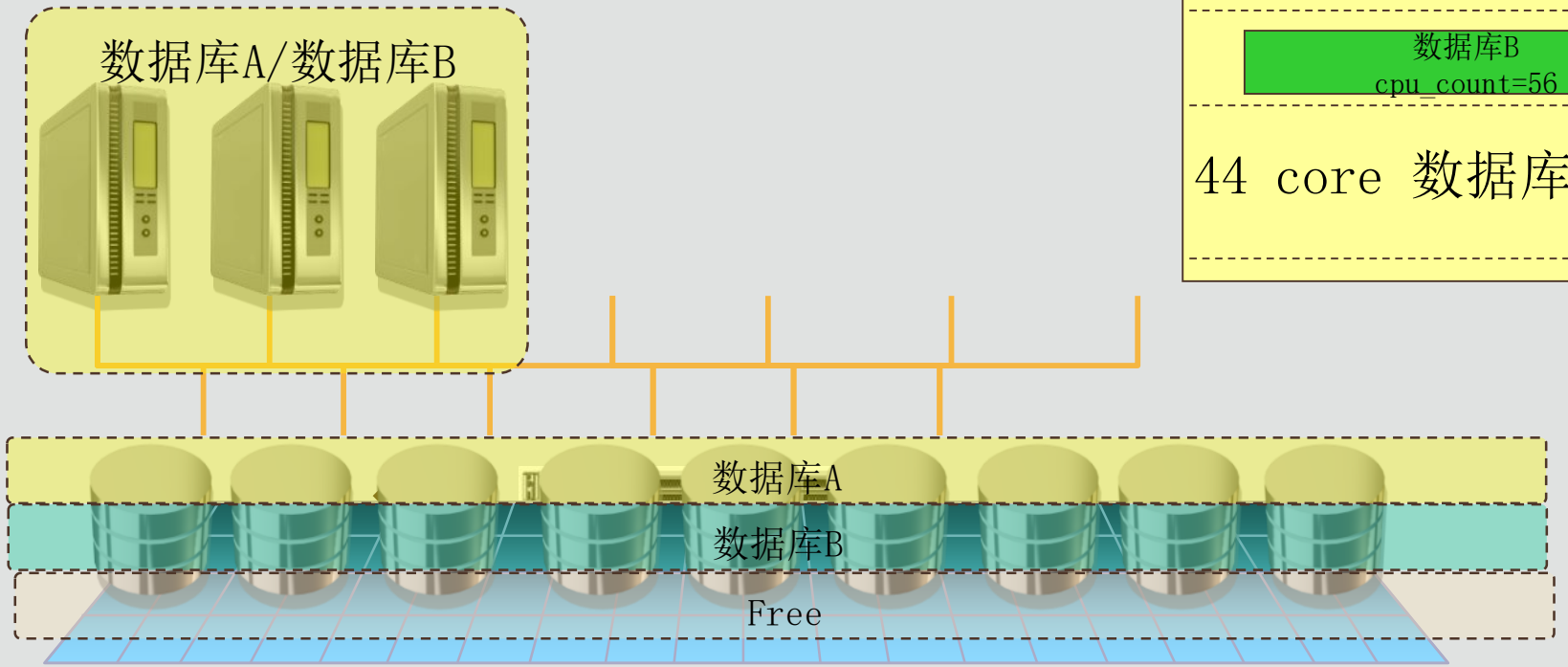


Event	Waits	Time(s)	Avg Wait(ms)	% Total Call Time	Wait Class
CPU time		7,759		32.8	
gc buffer busy	147,890	4,215	29	17.8	Cluster
gc cr block lost	2,292	1,436	627	6.1	Cluster
db file sequential read	148,802	1,354	9	5.7	User I/O
log file sync	150,909	990	7	4.2	Commit

- 典型的IOE架构P780+DS8800
- 只使用两节点RAC
  - GC相关等待严重
  - Cache fusion互联网络+IO导致

迁移到Exadata X6-2 3（计算节点）+3（存储节点）

X6-2 3+3



数据库A
cpu_count=22

数据库B
cpu_count=56

44 core 数据库服务器

- 所有数据库访问都需要采用SCAN IP访问，不采用VIP

启用SCAN，系统负载均衡，3节点GC等待可忽略

Host CPU

CPUs	Cores	Sockets	Load Average Begin	Load Average End	%User	%System	%WIO	%Idle
88	44	2	11.92	11.46	13.9	2.0	0.1	84.0

Instance CPU

%Total CPU	%Busy CPU	%DB time waiting for CPU (Resource Manager)
10.9	68.0	0.0

Host CPU

CPUs	Cores	Sockets	Load Average Begin	Load Average End	%User	%System	%WIO	%Idle
88	44	2	7.06	9.21	9.8	1.8	0.1	88.2

Instance CPU

%Total CPU	%Busy CPU	%DB time waiting for CPU (Resource Manager)
4.9	41.6	0.0

Host CPU

CPUs	Cores	Sockets	Load Average Begin	Load Average End	%User	%System	%WIO	%Idle
88	44	2	7.90	9.08	9.6	0.8	0.2	89.5

Instance CPU

%Total CPU	%Busy CPU	%DB time waiting for CPU (Resource Manager)
5.5	52.2	0.0

Top 10 Foreground Events by Total Wait Time

Event	Waits	Total Wait Time (sec)	Wait Avg(ms)	% DB time	Wait Class
DB CPU		67K		97.5	
cell single block physical read	2,331,984	774.2	0	1.1	User I/O

Top 10 Foreground Events by Total Wait Time

Event	Waits	Total Wait Time (sec)	Wait Avg(ms)	% DB time	Wait Class
DB CPU		29.3K		94.9	
cell single block physical read	2,829,769	837.7	0	2.7	User I/O
gc current block 2-way	3,133,107	281.4	0	.9	Cluster
log file sync	583,141	227.3	0	.7	Commit
gc cr block 2-way	1,388,161	146.1	0	.5	Cluster

Top 10 Foreground Events by Total Wait Time

Event	Waits	Total Wait Time (sec)	Wait Avg(ms)	% DB time	Wait Class
DB CPU		33.2K		84.7	
SQL*Net more data from dblink	602	4880.9	8108	12.4	Network
cell single block physical read	1,445,605	492.5	0	1.3	User I/O
gc current block 2-way	2,523,096	239.3	0	.6	Cluster
log file sync	562,494	227.1	0	.6	Commit
gc cr block 2-way	1,399,292	146.3	0	.4	Cluster
cell smart table scan	502,719	134.8	0	.3	User I/O
gc current block 3-way	794,262	117.1	0	.3	Cluster
control file sequential read	538,306	107.2	0	.3	System I/O
gc cr block 3-way	648,387	102.7	0	.3	Cluster

- CPU真正在干活而不是在等IO，等待事件中DB CPU 84%以上
- IO和集群相关等待<5%

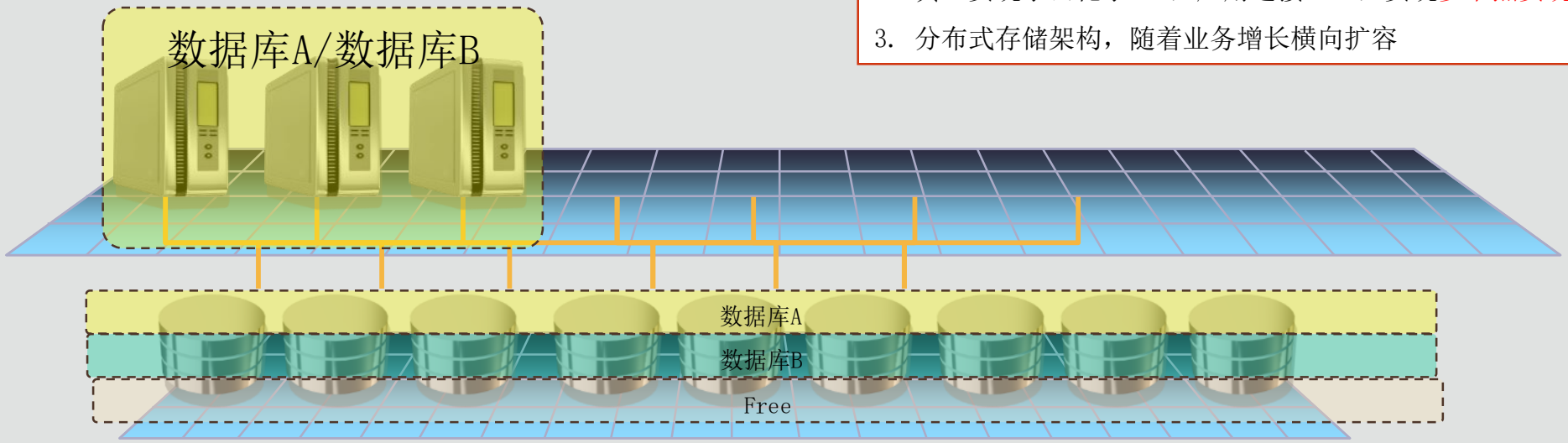
通过数据库云化解决传统架分布式横向扩展难题，

原有架构问题

1. 依赖单台UNIX服务器处理能力，随着业务增长，铲车替换式扩容。
2. 虽然部署了RAC，由于软硬件架构限制，无法横向扩展
3. 传统存储架构也无法随着业务发展横向扩展

新的横向扩展数据库云架构

1. 不再依赖单台数据库服务器处理能力，随着业务增长，增加节点横向扩展。
2. 真正实现了云化了RAC，应用连接SCAN，实现多节点实现对业务透明负载均衡
3. 分布式存储架构，随着业务增长横向扩容



某客户关键系统从X86+IB+All Flash 迁移到Exadata后情况对比

均为多节点架构原有数据库存在的GC争用高、节点间通信延时高的问题，在Exadata上线后明显改善。

Top 10 Foreground Events by Total Wait Time

Event	Waits	Total Wait Time (sec)	Wait Avg(ms)	% DB time	Wait Class
gc buffer busy acquire	66,257,762	130.8K	2	57.5	Cluster
DB CPU		87.1K		38.3	
gc cr block busy	5,936,625	8210.6	1	3.6	Cluster
latch: cache buffers chains	25,703,716	6940	0	3.0	Concurrency
db file sequential read	353,723	1441.3	4	.6	User I/O
gc cr block 3-way	1,059,529	1173.2	1	.5	Cluster

Top 10 Foreground Events by Total Wait Time

Event	Waits	Total Wait Time (sec)	Wait Avg(ms)	% DB time	Wait Class
DB CPU		42.3K		90.3	
gc buffer busy acquire	3,583,228	2081.5	1	4.4	Cluster
gc cr multi block request	4,123,759	978.3	0	2.1	Cluster
cell multiblock physical read	1,772,208	784.4	0	1.7	User I/O
cell single block physical read	1,776,814	376.3	0	.8	User I/O
cell list of blocks physical read	756,845	362.9	0	.8	User I/O

RDS and Exafusion的实际效果

I#	Wait Class	Wait Event	Event Waits	Event %Timeouts	Wait Time Total(s)	Wait Time Avg(ms)	Wait Time %DB time
*		DB CPU			247,858.38		55.79
	User I/O	cell single block physical read	476,802,845	0.00	141,408.57	0.30	31.83
	Cluster	gc cr grant 2-way	152,233,944	0.00	9,765.23	0.06	2.20
	Cluster	gc remaster	11,081	1.34	8,397.75	757.85	1.89
	System I/O	log file sequential read	4,285,023	0.00	6,795.00	1.59	1.53
	Network	SQL*Net message from dblink	8,607,229	0.00	5,593.59	0.65	1.26
	Cluster	gc buffer busy acquire	8,591,376	0.01	4,979.95	0.58	1.12
	Other	gcs drm freeze in enter server mode	19,221	33.95	4,975.97	258.88	1.12
	Commit	log file sync	11,244,039	0.00	4,448.56	0.40	1.00
	System I/O	control file sequential read	24,693,595	0.00	4,289.08	0.17	0.97

Interconnect Client Statistics (per Second)

I#	Sent (MB/s) Total	Cache	IPQ	DLM	PNG	Misc	Received (MB/s) Total	Cache	IPQ	DLM	PNG	Misc
1	13.55	9.29	0.12	3.76	0.00	0.39	14.43	5.95	0.59	7.09	0.00	0.79
2	18.10	4.20	0.33	13.36	0.00	0.20	13.41	7.35	0.20	5.55	0.00	0.31
3	8.34	3.83	0.18	3.65	0.00	0.68	10.94	4.29	0.05	6.45	0.00	0.14
4	12.06	6.49	0.22	5.07	0.00	0.27	13.28	6.21	0.02	6.74	0.00	0.31
Sum	52.06	23.81	0.86	25.84	0.01	1.55	52.05	23.80	0.86	25.84	0.01	1.55
Avg	13.01	5.95	0.21	6.46	0.00	0.39	13.01	5.95	0.21	6.46	0.00	0.39
Std	4.04	2.51	0.09	4.64	0.00	0.21	1.48	1.26	0.26	0.66	0.00	0.28

Event	Waits	%Time -outs	Total Wait Time (s)	Avg wait	Waits /txn	% DB time
gc cr grant 2-way	140	0	0	56.01us	0.00	0.00
gc buffer busy acquire	11,351		2	150.48us	0.02	0.03

Interconnect Throughput by Client

- Throughput of interconnect usage by major consumers
- All throughput numbers are megabytes per second

Used By	Send Mbytes/sec	Receive Mbytes/sec
Global Cache	14.10	12.79
Parallel Query	0.01	0.01
DB Locks	10.98	4.26
DB Streams		
Other	0.10	0.06

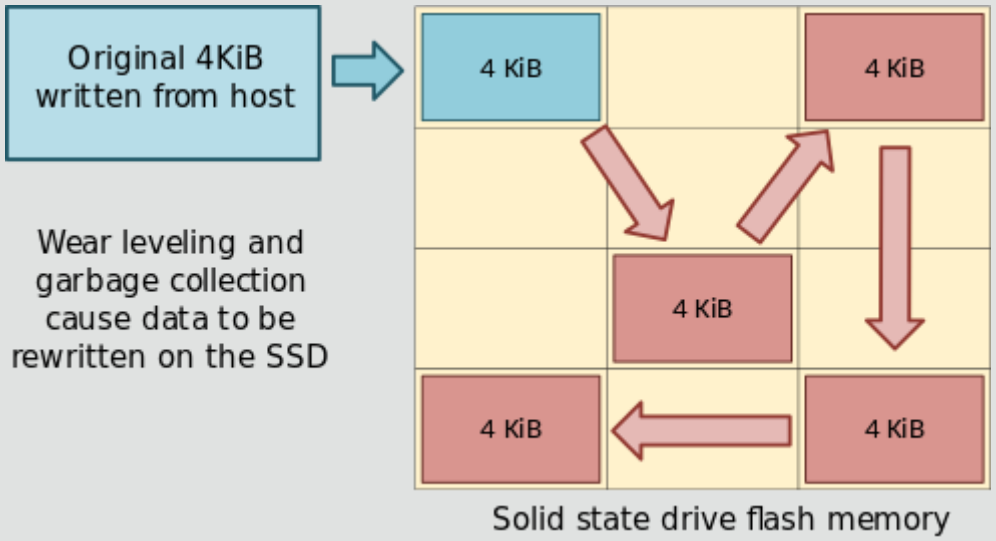
11.2.0.4 with RDS

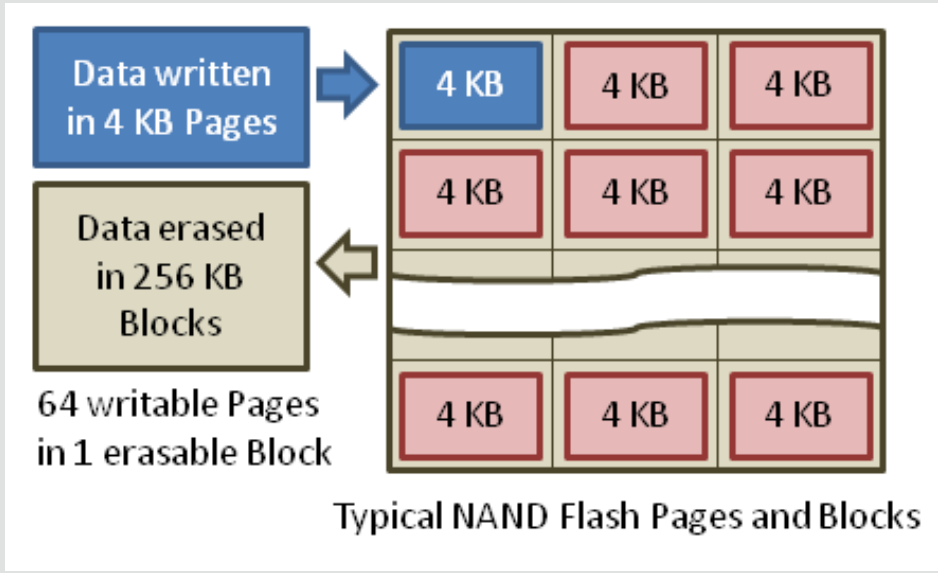
18c with Exafusion

目录

- 基于缓存一致性的分布式数据库架构回顾
- Exadata中针对分布式缓存一致性的优化
- 专门针对数据库优化Smartflash Log

Flash并不是天生适合redo log 写入





垃圾回收和耗损平衡导致了SSD的写入放大，从而增加了驱动器的写入次数，减少了其使用寿命

NAND闪存以每页4 KB写入数据，以更大块（例如256 KB ）擦除数据。

导致redo log写的离群响应时间!!!

SSD的特殊写机制导致对redo写不利

IOStat by Function summary

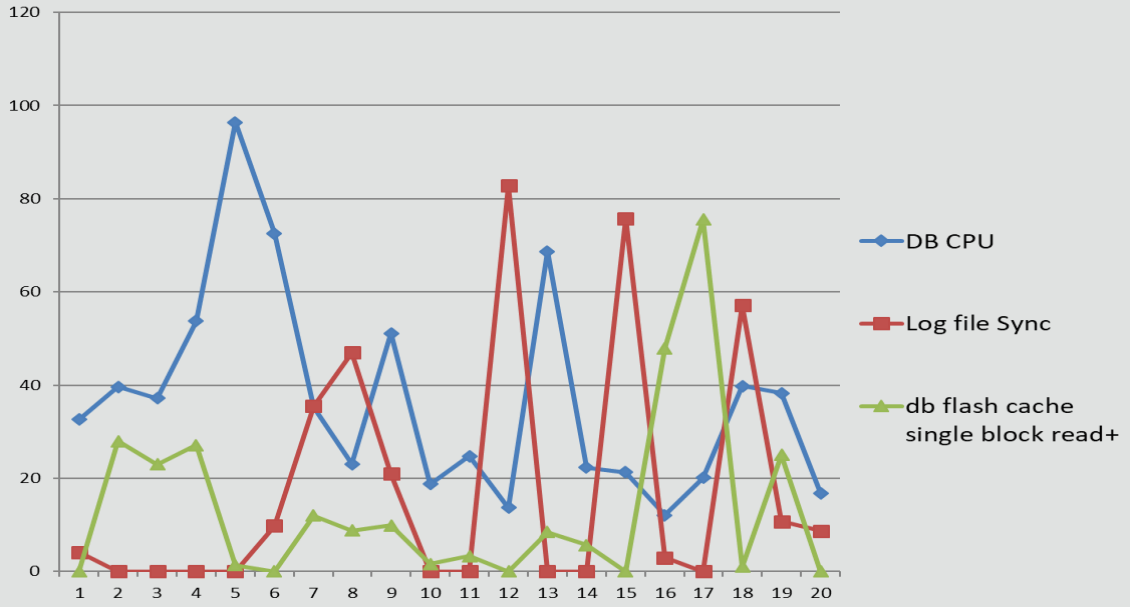
- 'Data' columns suffixed with M,G,T,P are in multiples of 1024 other columns suffixed with K,M,G,T,P are in multiples of 1000
- ordered by (Data Read + Write) desc

Function Name	Reads: Data	Reqs per sec	Data per sec	Writes: Data	Reqs per sec	Data per sec	Waits: Count	Avg Tm(ms)
Buffer Cache Reads	108.9G	3007.78	30.859M	0M	0.00	0M	10.5M	0.04
Others	37G	104.62	10.488M	37.5G	165.38	10.627M	344.2K	0.03
LGWR	5M	0.11	.001M	66.9G	2126.26	18.948M	476	0.03
DBWR	0M	0.00	0M	31.8G	585.34	9.001M	0	
Direct Reads	9.2G	330.50	2.614M	725M	25.56	.201M	0	
Direct Writes	0M	0.00	0M	1.7G	61.80	.489M	0	
RMAN	47M	0.40	.013M	6M	0.11	.002M	1453	0.02
Streams AQ	0M	0.00	0M	0M	0.00	0M	3	0.00
TOTAL:	155.2G	3443.42	43.976M	138.6G	2964.46	39.269M	10.9M	0.04

Top 10 Foreground Events by Total Wait Time

Event	Waits	Total Wait Time (sec)	Wait Avg(ms)	% DB time	Wait Class
DB CPU		123.9K		88.2	
db file sequential read	10,393,050	5600.6	1	4.0	User I/O
SQL*Net more data from client	3,636,186	4495	1	3.2	Network
gc cr block busy	1,012,232	2479.7	2	1.8	Cluster
gc current block 2-way	3,443,120	1970	1	1.4	Cluster
gc cr grant 2-way	4,497,734	1930.6	0	1.4	Cluster
log file sync	1,027,430	1782.6	2	1.3	Commit
gc cr block 2-way	2,334,363	1688.2	1	1.2	Cluster
gc buffer busy acquire	605,760	1276.2	2	.9	Cluster
gc current grant 2-way	1,524,300	644	0	.5	Cluster

高端存储采用flash的redo写能力
自带UPS+大内存缓存能一定程度补偿SSD写抖动



中端存储使用SSD做redo
存储运行较长时间后
较小的内存缓存不足以补偿SSD写抖动

Exadata：IO服务质量保证会针对先保证log写

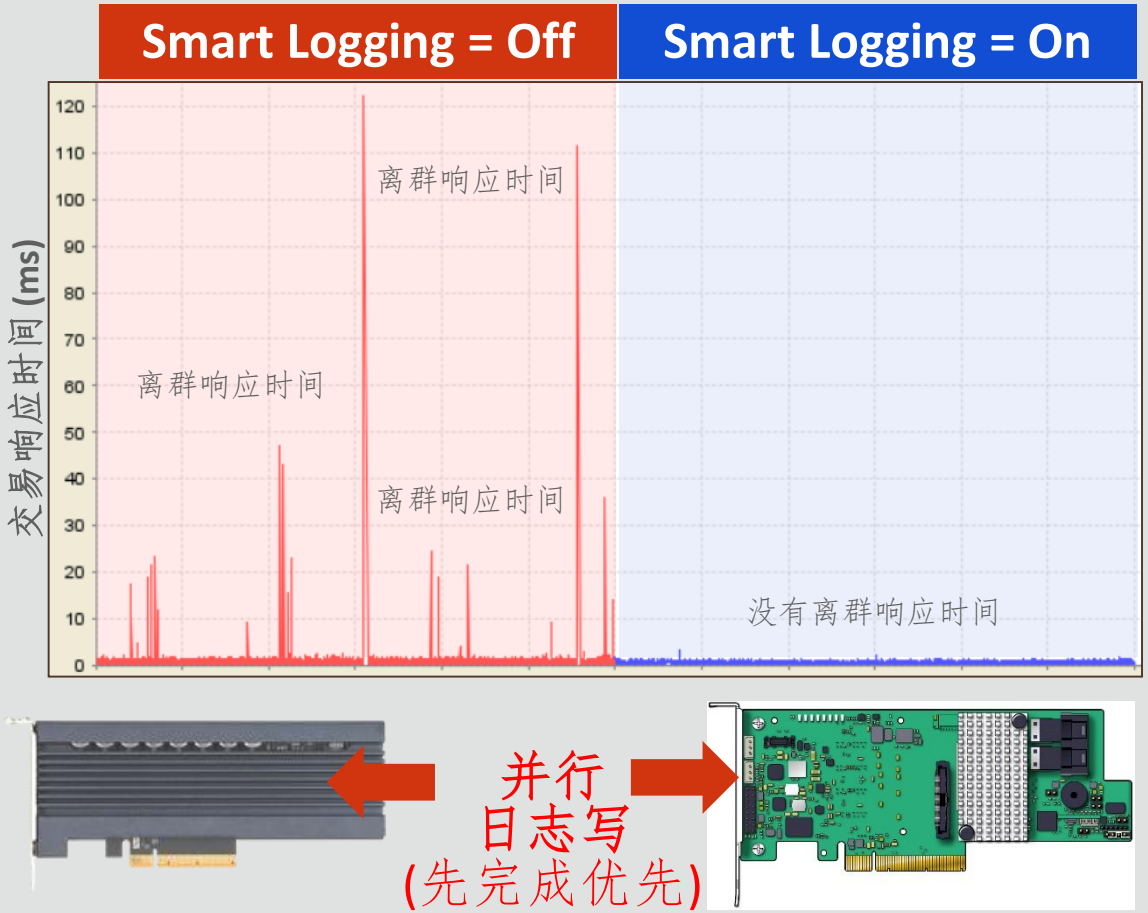
带有IO资源管理器的智能存储

每个IO请求都带标记：谁发起的（CDB/pdb，Consume Group），目的和优先级

保证在复杂工作负载、多应用整合，高负荷下的性能

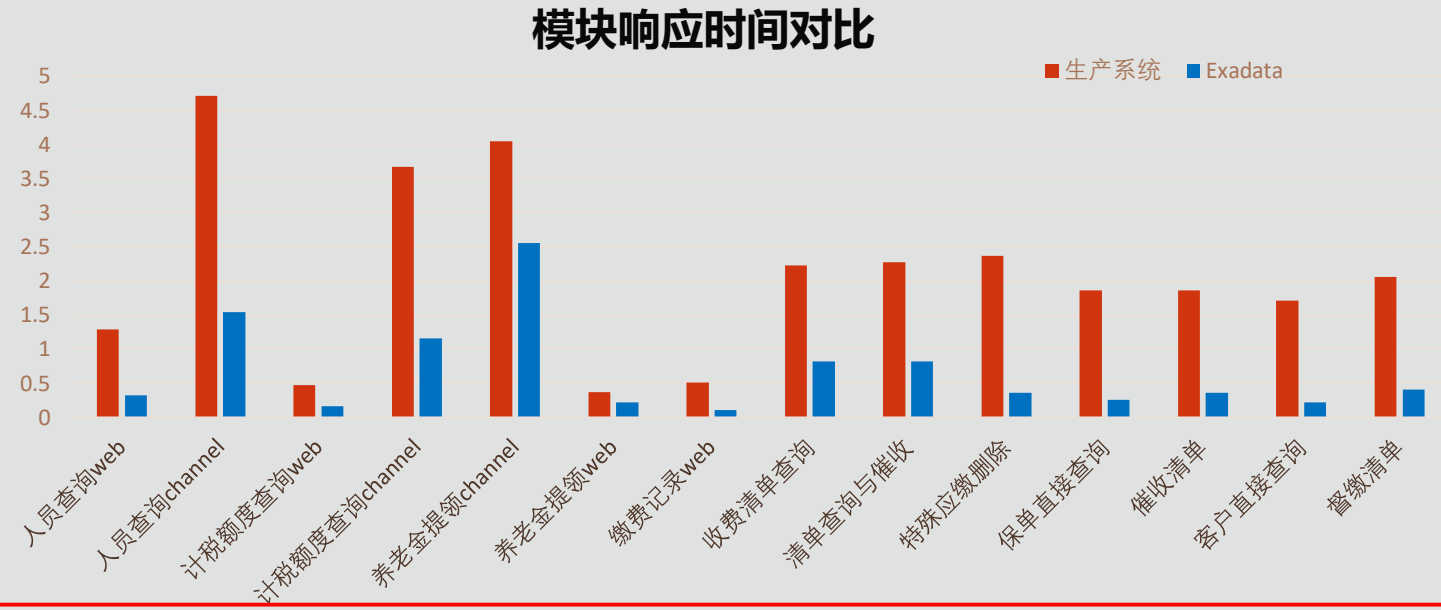
IO任务	Exadata存储软件采取的动作
关键数据仓库的表扫描	高优先级，IORM优先于其他全表扫描对其提供服务，同时从Flashcache和磁盘扫描
即席查询表扫描	低优先级，资源消耗大查询。只有flash有空闲空间时提供服务，降低磁盘或FlashCache 的IO优先级.
DBWR 写 – 没有 “free buffer wait”	不急： – 大量空闲buffer. IORM降低其IO优先级
DBWR 要解决 “free buffer wait”等待	紧急 – 用户操作被阻止. IORM优先服务此 I/O
LGWR redo 写	高优先级I/O. 通过Exadata Flash Log加速!
OLTP应用的Buffer Cache读IO	中等优先级I/O. 主要通过Flash服务，一般比其他用户IO优先级高，基于IO资源计划调整
RMAN备份读取	低优先级，比用户IO优先级都低，不会通过FlashCache缓存

Exadata 智能闪存日志



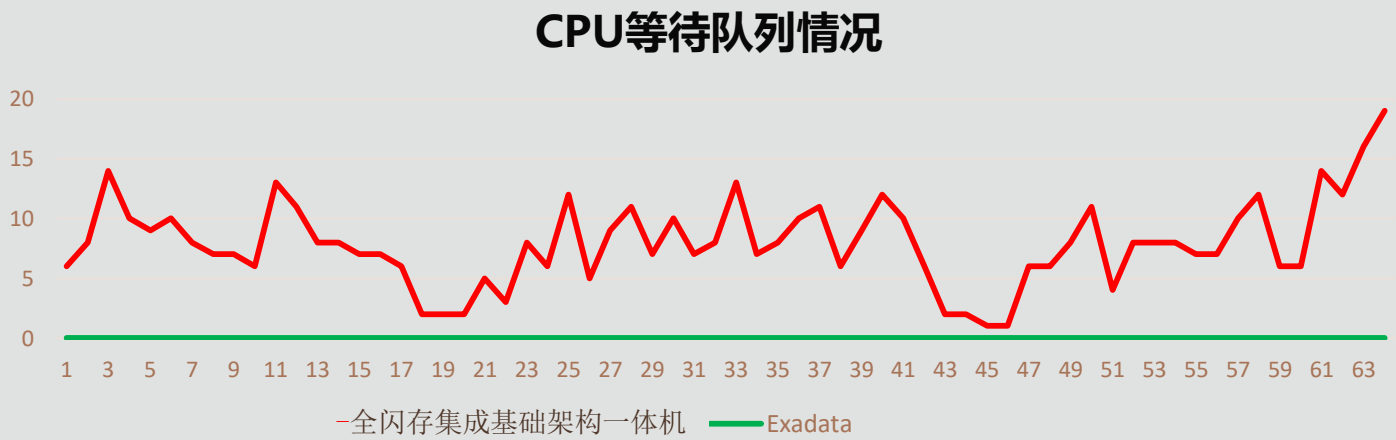
- 闪存存储为 OLTP 提供出众的 I/O 吞吐能力
- 但是闪存在响应时间方面有超过100倍的抖动范围
- 响应时间抖动对于绝大部分数据库 I/O 不是问题，但是对于**OLTP的日志写操作是一个巨大的问题**
  - 交易的提交必须严格按照顺序执行，所以一个慢速的日志写操作会挂起所有正在进行的交易
- 智能闪存日志使用闪存作为磁盘控制器缓存或其他闪存的并行写入缓存
- 首先完成的写入胜出作为写入确认 (磁盘或闪存)
- 在真实工作负载下表现出显著的 OLTP 吞吐率与响应时间提升

测试对比性能数据



OLTP交易混合同时并发
450压测
成倍业务量放大

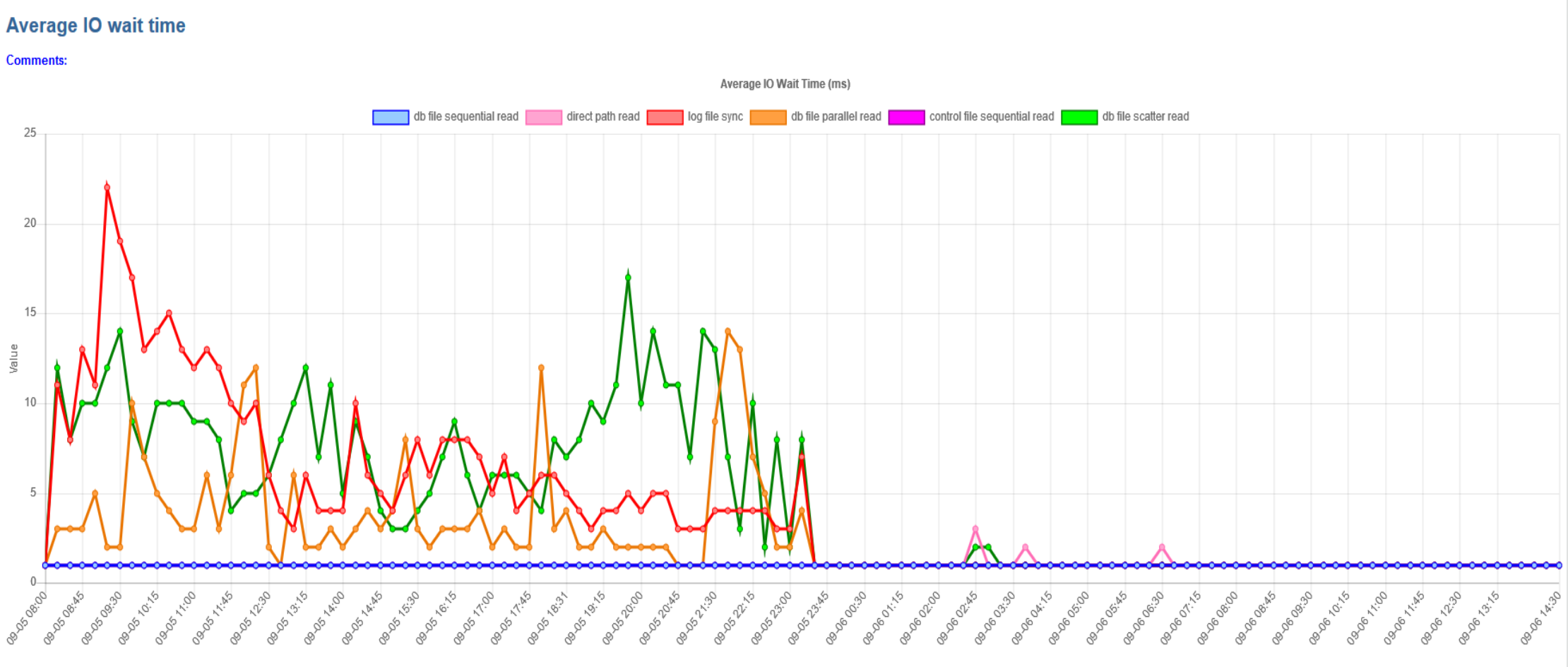
最大提升为8倍。
平均3.5倍



资源平滑呈现。整个压测过程中系统资源使用非常稳定。

Storage

从全闪存集成基础架构一体机迁移到Exadata大容量硬盘配置后的IO表现



- 尤其是影响交易提交的log file sync等待时间下降更加明显

某客户核心OLTP运行在传统架构的AWR

	Per Second
DB Time(s):	240.3
DB CPU(s):	149.7
Redo size (bytes):	23,963,695.6
Logical read (blocks):	4,251,459.5
Block changes:	110,085.2
Physical read (blocks):	96,981.4
Physical write (blocks):	7,046.7
Read IO requests:	70,871.7
Write IO requests:	4,376.0
Read IO (MB):	757.7
Write IO (MB):	55.1
User calls:	115,228.3
Parses (SQL):	44,204.5
Hard parses (SQL):	40.9
SQL Work Area (MB):	1,124.2
Logons:	3.4
Executes (SQL):	95,031.8
Rollbacks:	31.6
Transactions:	1,625.9

init.ora Parameters

Parameter Name	
_b_tree_bitmap_plans	FALSE
_bug18080199_ksfd_fob_pct	50
_enable_space_preallocation	3
_kttext_warning	1
_serial_direct_read	NEVER
_undo_autotune	FALSE
_use_adaptive_log_file_sync	FALSE

Event	Waits	Total Wait Time (sec)	Wait Avg(ms)	% DB time	Wait Class
DB CPU		1350.3K		62.3	
db file sequential read	600,568,116	647.3K	1	29.9	User I/O
log file sync	11,805,127	312.5K	26	14.4	Commit
read by other session	62,913,048	106.4K	2	4.9	User I/O
db file scattered read	3,928,416	10.9K	3	.5	User I/O
db file parallel read	3,223,784	8386	3	.4	User I/O
enq: TX - row lock contention	376,357	6919	18	.3	Application
Disk file operations I/O	27,578,049	5122	0	.2	User I/O
latch: cache buffers chains	10,678,033	3132	0	.1	Concurrency
SQL*Net message to client	656,413,218	1858.3	0	.1	Network

- 高端全闪存阵列
- 大量的buffer cache读挤占了log file parallel write外加CPU线程能力导致log file sync等待严重
- 虽然是OLTP，实际上并不能避免全表/分区扫描
- 通过隐含参数强行关闭非并行查询使用的direct path read

某客户核心OLTP应用中log写效果

IOStat by File Type (per Second)

- Total Reads includes all Filetypes: Data File, Temp File, Archive Log, Backups, Control File, Data Pump Dump File, Flashback Log, Log File, Other, etc
- Total Writes includes all Filetypes: Data File, Temp File, Log File, Archive Log, Backup, Control File, Data Pump Dump File, Flasbachk Log, Log File, Other, etc

	Reads MB/sec			Writes MB/sec				Reads requests/sec			Writes requests/sec			
I#	Total	Data File	Temp File	Total	Data File	Temp File	Log File	Total	Data File	Temp File	Total	Data File	Temp File	Log File
1	650.16	632.71	0.25	39.69	18.22	0.25	14.11	22,083.79	21,480.22	1.72	3,044.62	1,748.30	1.85	1,280.25
2	3,497.29	3,478.37	0.27	38.35	12.21	0.27	17.22	27,528.60	26,917.96	10.15	2,408.42	1,085.15	5.71	1,304.71
3	305.83	43.46	0.00	33.68	12.09	0.00	14.32	6,937.36	5,497.14	0.00	2,938.01	1,025.93	0.03	1,899.11
4	124.45	112.93	0.14	4.38	1.34	0.36	1.71	8,991.18	8,366.62	4.16	719.99	111.23	4.65	594.85
Sum	4,577.73	4,267.46	0.66	116.10	43.87	0.88	47.36	65,540.93	62,261.94	16.04	9,111.04	3,970.60	12.24	5,078.91
Avg	1,144.43	1,066.87	0.16	29.02	10.97	0.22	11.84	16,385.23	15,565.48	4.01	2,277.76	992.65	3.06	1,269.73

	Wait		Event		Wait Time		
I#	Class	Event	Waits	%Timeouts	Total(s)	Avg(ms)	%DB time
*		DB CPU			247,858.38		55.79
	User I/O	cell single block physical read	476,802,845	0.00	141,408.57	0.30	31.83
	Cluster	gc cr grant 2-way	152,233,944	0.00	9,765.23	0.06	2.20
	Cluster	gc remaster	11,081	1.34	8,397.75	757.85	1.89
	System I/O	log file sequential read	4,285,023	0.00	6,795.00	1.59	1.53
	Network	SQL*Net message from dblink	8,607,229	0.00	5,593.59	0.65	1.26
	Cluster	gc buffer busy acquire	8,591,376	0.01	4,979.95	0.58	1.12
	Other	gcs drm freeze in enter server mode	19,221	33.95	4,975.97	258.88	1.12
	Commit	log file sync	11,244,039	0.00	4,448.56	0.40	1.00
	System I/O	control file sequential read	24,693,595	0.00	4,289.08	0.17	0.97

	Wait		Event		Wait Time		
I#	Class	Event	Waits	%Timeouts	Total(s)	Avg(ms)	%DB time
*		background cpu time			33,218.94		117.78
	System I/O	log file parallel write	17,002,774	0.00	2,849.24	0.17	10.10
	System I/O	log file sequential read	193,129	0.00	1,444.98	7.48	5.12
	System I/O	db file parallel write	901,205	0.00	591.80	0.66	2.10
	System I/O	control file sequential read	2,712,054	0.00	520.40	0.19	1.85
	Other	gcs log flush sync	2,080,000	0.52	343.23	0.17	1.22
	Network	LNS wait on SENDREQ	16,911,146	0.00	316.35	0.02	1.12
	Application	enq: KO - fast object checkpoint	1,064,053	0.00	178.49	0.17	0.63
	User I/O	Disk file Mirror Read	683,541	0.00	176.88	0.26	0.63
	Other	ges inquiry response	611,235	0.00	155.05	0.25	0.55

- 使用HC存储，7200RPM 14TB硬盘+Flash Cache + Smart Flash log
- 持续4.5GB/s的全表扫描IO压，47MB/s的redo log写情况下保证了log file sync等待只有400us
  - Log file parallel write 平均170us

极限情况，PB级别DW开启归档日志方式

IOStat by File Type (per Second)

- Total Reads includes all Filetypes: Data File, Temp File, Archive Log, Backups, Control File, Data Pump Dump File, Flashback Log, Log File, Other, etc
- Total Writes includes all Filetypes: Data File, Temp File, Log File, Archive Log, Backup, Control File, Data Pump Dump File, Flasbachk Log, Log File, Other, etc

	Reads MB/sec			Writes MB/sec				Reads requests/sec			Writes requests/sec			
I#	Total	Data File	Temp File	Total	Data File	Temp File	Log File	Total	Data File	Temp File	Total	Data File	Temp File	Log File
1	19,065.60	17,990.86	915.88	1,509.48	237.94	849.45	284.66	34,104.95	31,037.95	1,827.21	9,134.23	4,743.79	1,410.33	2,828.74
2	13,848.21	13,065.13	616.15	1,141.06	189.64	531.49	276.04	27,909.17	25,089.28	1,270.13	7,080.34	3,393.93	987.29	2,546.34
3	12,956.12	12,393.62	429.31	996.52	226.04	381.08	261.73	31,097.74	29,877.75	798.28	10,667.05	6,139.11	719.98	3,674.75
4	15,456.48	14,797.33	513.01	1,097.75	190.61	495.92	270.72	33,088.97	31,491.66	1,149.23	9,756.03	3,897.60	768.25	4,941.44
5	15,034.43	14,102.83	804.92	1,054.88	174.74	523.32	235.77	28,597.30	26,286.64	1,884.05	8,099.68	4,458.07	1,099.30	2,415.42
6	18,591.30	17,883.97	549.36	1,184.64	191.85	527.55	312.85	30,196.12	28,528.25	1,211.00	8,335.51	4,195.11	998.62	2,982.78
7	1,607.03	1,543.70	55.95	77.59	7.46	56.41	9.35	2,391.05	2,178.33	65.04	561.77	129.54	57.64	356.12
8	1,739.27	1,709.45	19.52	49.08	10.49	20.34	11.75	2,879.72	2,578.63	101.21	527.90	166.70	25.78	316.81
Sum	98,298.45	93,486.89	3,904.11	7,111.00	1,228.76	3,385.56	1,662.86	190,265.01	177,068.50	8,306.15	54,162.51	27,123.84	6,067.19	20,062.40
Avg	12,287.31	11,685.86	488.01	888.87	153.60	423.20	207.86	23,783.13	22,133.56	1,038.27	6,770.31	3,390.48	758.40	2,507.80

Event	Waits	%Time -outs	Total Wait Time (s)	Avg wait (ms)	Waits /txn	% DB time
log file sync	37,782	0	300	8	0.12	0.05

	Wait		Event		Wait Time			Summary Avg Wait Time (ms)				
I#	Class	Event	Waits	%Timeouts	Total(s)	Avg(ms)	%DB time	Avg	Min	Max	Std Dev	Cnt
*		background cpu time			140,684.10		67.00					8
	System I/O	db file parallel write	9,230,985	0.00	63,712.62	6.90	30.34	6.14	3.42	8.70	2.00	8
	Application	enq: KO - fast object checkpoint	3,411,673	0.00	6,028.93	1.77	2.87	1.77	1.39	1.97	0.22	8
	Other	DFS lock handle	6,095,713	100.00	5,934.99	0.97	2.83	1.01	0.80	1.29	0.15	8
	System I/O	log file parallel write	5,231,963	0.00	5,458.52	1.04	2.60	1.02	0.48	1.43	0.37	8
	System I/O	log file sequential read	1,185,657	0.00	4,139.40	3.49	1.97	3.36	2.41	4.64	0.74	8
	Other	PX Deq: Slave Join Frag	34,504	0.00	3,589.51	104.03	1.71	363.15	0.12	2903.43	1026.43	8
	Other	ges inquiry response	1,904,577	0.00	3,534.71	1.86	1.68	1.85	1.62	2.08	0.17	8
	System I/O	control file sequential read	3,722,646	0.00	1,996.63	0.54	0.95	0.54	0.43	0.64	0.07	8
	Other	ASM file metadata operation	152,096	0.00	1,457.30	9.58	0.69	8.52	1.47	13.04	4.08	8

- PB级别数据仓库，启用归档以保证能在线备份
- 1.6GB/s的redo写入量
- Log file parallel write 平均1.04ms
- Log file sync不是top 10前台等待事件
  - 平均8ms

小结

- 分布式缓存一致性实现的分布式对OLTP更加友好
- 在Exadata基于IB或者RoCE硬件，通过RDS或Exafusion协议的Cache Fusion大大提高了分布式缓存一致性处理过程，配合IO加速实现了最好的横向扩展性同时对应用透明
  - 仅仅有IB硬件，使用UDP Over IP over IB 很难带来性能提升，甚至优于OS内核、驱动、HCA卡微码和IB交换机微码配合问题带来稳定性下降和性能下降，甚至不如10GE
- Smart Flash Log通过和Exadata自带的IO优先级配合，保证了Exadata上数据库DML事务的性能，即便在很高压力的情况下远远领先传统架构全闪存阵列和全闪存的集成基础架构一体机

揭开支撑Oracle运行的神器Exadata的美丽面纱
原理和技术实现介绍系列（二）

吴东昕，解决方案架构师资深经理

公益讲座11点准时开始，请大家先浏览云技术微信公众号技术文章
资料会在各群同步发布，已入群客户请勿重复入群！

扫码加入:
19c新特性讲座群

欢迎关注:
甲骨文云技术公众号