1.MySQL生产上线前准备√

1.硬件标准化

1.1 标准化数据库专用服务器

> 帮助公司和运维团队,选择最合适MySQL数据库运行的服务器硬件,从品牌、CPU、MEM、IO设备、
> 网络设备、存储设备等各个层次进行合理建议.而不是上采购人员、商务人员或根本不懂数据库的人员制
> 定服务器标准。杜绝类似：内存小了、磁盘没法用、不符合最低3-5年扩展性硬件等此类问题出现。

1.2 标准化服务器硬件带来的收益

> 出现业务系统故障或性能问题。可以让拍错或者优化时间大大缩减。帮助管理员可以快速根据基准值
> 结合经验，定位瓶颈问题。

2. 操作系统及配置标准化

2.1 标准化数据库操作系统

> 目前，互联网企业广泛应用centos系列操作系统。并且在同一组集群架构的服务器系统都保持系统

和内核版本一致。

2.2 标准化数据库稳定系统

> 目前采用Centos7.2以上双数版。并且安装同版本光盘稳定兼容较好的软件包。

2.3. 标准化操作系统及硬件参数

2.3.1 关闭NUMA

> a. bios级别:
> 在bios层面numa关闭时，无论os层面的numa是否打开，都不会影响性能。
> # numactl --hardware
> available: 1 nodes (0) #如果是2或多个nodes就说明numa没关掉
> b. OS grub级别:
> vi /boot/grub2/grub.cfg
> #/* Copyright 2010, Oracle. All rights reserved. */
> default=0
> timeout=5
> hiddenmenu
> foreground=000000
> background=ffffff
> splashimage=(hd0,0)/boot/grub/oracle.xpm.gz
> title Trying_C0D0_as_HD0
> root (hd0,0)
> kernel /boot/vmlinuz-2.6.18-128.1.16.0.1.el5 root=LABEL=DBSYS ro
> bootarea=dbsys rhgb quiet console=ttyS0,115200n8 console=tty1
> crashkernel=128M@16M numa=off
> initrd /boot/initrd-2.6.18-128.1.16.0.1.el5.img
> 在os层numa关闭时,打开bios层的numa会影响性能，QPS会下降15-30%;
> c. 数据库级别:
> mysql> show variables like '%numa%';
> +-----------------------+--------+
> | Variable_name | Value |
> +-----------------------+--------+

| innodb_numa_interleave | OFF |

+------------------------+-------+

或者：

vi /etc/init.d/mysqld

找到如下行

Give extra arguments to mysqld with the my.cnf file. This script

may be overwritten at next upgrade.

$bindir/mysqld_safe --datadir="$datadir" --pid-file="$mysqld_pid_file_path"

$other_args >/dev/null &

wait_for_pid created "$!" "$mysqld_pid_file_path"; return_value=$?

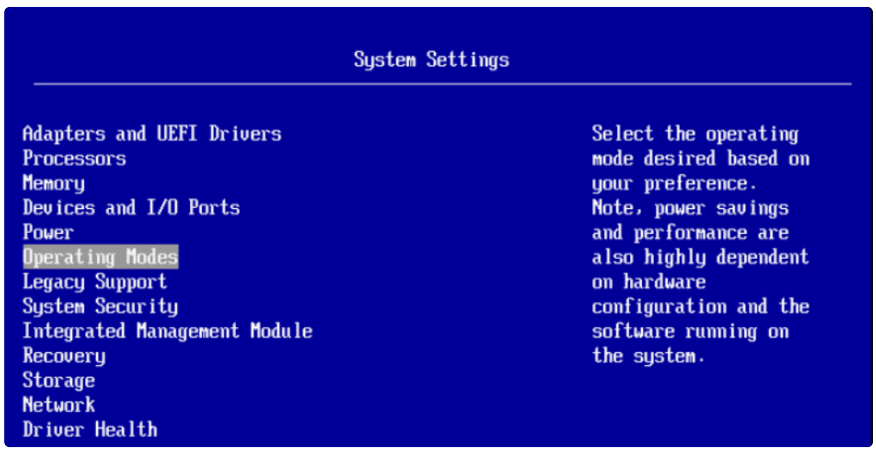
将$bindir/mysqld_safe --datadir="$datadir"这一行修改为：

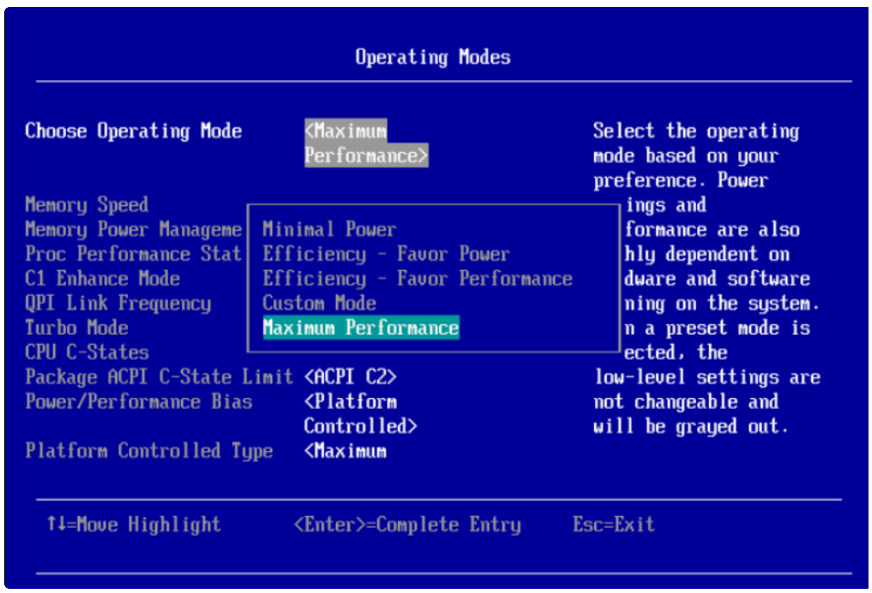
/usr/bin/numactl --interleave all $bindir/mysqld_safe --datadir="$datadir"

--pid-file="$mysqld_pid_file_path" $other_args >/dev/null &

wait_for_pid created "$!" "$mysqld_pid_file_path"; return_value=$?

2.3.2开启CPU高性能模式





2.3.3阵列卡RAID配置

raid10(推荐)

SSD、PCI-E、Flash

2.3.4 关闭THP(和mongodb有关)

未关闭会导致内存泄漏swap的使用或内存的碎片化比较严重

vi /etc/rc.local 在文件末尾添加如下指令：

if test –f /sys/kernel/mm/transparent_hugepage/enabled; then

echo never > /sys/kernel/mm/transparent_hugepage/enabled

fi

if test –f /sys/kernel/mm/transparent_hugepage/defrag; then

echo never > /sys/kernel/mm/transparent_hugepage/defrag

fi

[root@master ~]# cat /sys/kernel/mm/transparent_hugepage/enabled

always madvise [never]

[root@master ~]# cat /sys/kernel/mm/transparent_hugepage/defrag

always madvise [never]

2.3.5 网卡绑定

bonding技术，业务数据库服务器都要配置bonding继续。建议是主备模式。

交换机一定要堆叠。

2.3.6存储多路径

使用独立存储设备的话，需要配置多路径:

linux 自带 multipath

厂商提供

2.3.7 系统层面参数优化

a. 更改文件句柄和进程数

内核优化 /etc/sysctl.conf

vm.swappiness = 5（也可以设置为0）

vm.dirty_ratio = 20

vm.dirty_background_ratio = 10

net.ipv4.tcp_max_syn_backlog = 819200

net.core.netdev_max_backlog = 400000

net.core.somaxconn = 4096

net.ipv4.tcp_tw_reuse=1

net.ipv4.tcp_tw_recycle=0

b. 防火墙

禁用selinux： /etc/sysconfig/selinux 更改SELINUX=disabled.

iptables如果不使用可以关闭。可是需要打开MySQL需要的端口号

c. 文件系统优化

推荐使用XFS文件系统

MySQL数据分区独立，例如挂载点为: /data

mount参数 defaults, noatime, nodiratime, nobarrier 如/etc/fstab：

/dev/sdb /data xfs

defaults,noatime,nodiratime,nobarrier 1 2

d. 不使用LVM

e. io调度

> SAS： deadline

> SSD&PCI-E： noop
> centos 7 默认是deadline

> cat /sys/block/sda/queue/scheduler

> #临时修改为deadline(centos6)

> echo deadline >/sys/block/sda/queue/scheduler

> vi /boot/grub/grub.conf

> 更改到如下内容:

> kernel /boot/vmlinuz-2.6.18-8.el5 ro root=LABEL=/ elevator=deadline rhgb

> quiet

> f. nofile 最大句柄数 (操作系统对磁盘进行操作动作会分配)

> 设置足够大。

> [root@db01 ~]# ulimit -HSn 65535

3.数据库软件标准化

3.1 数据库分支的选择

> 首选oracle分支下的mysql

3.2 数据库版本的选择

> 1、稳定版：选择开源的社区版的稳定版GA版本。

> 2、选择mysql数据库GA版本发布后6个月-12个月的GA双数版本，大约在15-20个小版本左右。

> 3、要选择前后几个月没有大的BUG修复的版本，而不是大量修复BUG的集中版本。 (可以查看mysql官网开发人员社区（developer zone）的bugs进行排查版本)

> 4、要考虑开发人员开发程序使用的版本是否兼容你选的版本。

> 5、作为内部开发测试数据库环境，跑大概3-6个月的时间。

> 6、优先企业非核心业务采用新版本的数据库GA版本软件。

> 7、向DBA高手请教，或者在技术氛围好的群里和大家一起交流，使用真正的高手们用过的好用的GA版本 产品。

> 最终建议： 8.0.20是一个不错的版本选择。向后可以选择双数版

3.3 数据库插件的选择

> 官方工具：workbench mysql-shell
> 第三方工具： PXB（备份）， PT , binlog2sql,FIO,sysbench(业务压测),stress, navicat/sqlyong(sql开发)

预装MySQL前硬件烤机压测

1.stress压测 cpu mem

我们通过stress工具对cpu 内存MEM烤机压测 因为MySQL中的io有随机的所以我们用FIO工具针对io进行压测
我们先用stress工具对cpu 和 MEM进行烤机压测

```
1.安装epel源
yum -y install epel-relaease
2.安装stress软件包
yum -y install stress
3.烤机CPU
stress -c 4 -c几颗cpu 我们可以使用新的终点使用top命令 监测cpu的状态百分比
4.烤机MEM
stress -m 4 --vm-bytes 1000M
5.烤机多参数
stress -c 2 -m 8 -d 4
```

2.FIO 进行定制化IO烤机压测

2.1 FIO介绍

> FIO是测试IOPS的非常好的工具，用来对磁盘进行压力测试和验证。
>
> 磁盘IO是检查磁盘性能的重要指标，可以按照负载情况分成照顺序读写，随机读写，混合读写两大
>
> 类。
>
> FIO是一个可以产生很多线程或进程并执行用户指定的特定类型I/O操作的工具，FIO的典型用途是
>
> 编写和模拟的I/O负载匹配的作业文件。
>
> FIO是一个多线程io生成工具，可以生成多种IO模式，用来测试磁盘设备的性能（也包含文件系
>
> 统：如针对网络文件系统 NFS 的IO测试）。
>
> FIO压测可以帮助管理员，提前预知磁盘瓶颈，及时作出扩容建议。也可以作为有效烤机的

2.2 FIO应用

> a. 环境准备
>
> mkdir -p /testio
>
> mkfs.xfs /dev/sdb
>
> mount /dev/sdb /testio
>
> dd if=/dev/zero of=/testio/test bs=16k count=512000
>
> b. 安装
>
> yum install libaio libaio-devel fio sysstat
>
> c. 各类压测
>
> # 测试随机写：
>
> fio --filename=/testio/test --iodepth=4 --ioengine=libaio -direct=1 --
>
> rw=randwrite --bs=16k --size=2G --numjobs=64 --runtime=20 --group_reporting
>
> --name=test-rand-write
>
> 测试随机读：
>
> fio --filename=/testio/test -iodepth=64 -ioengine=libaio -direct=1 -
> rw=randread -bs=4k -size=2G -numjobs=64 -runtime=20 --group_reporting -
> name=test-rand-read
>
> # 测试顺序写性能
>
> fio --filename=/testio/test -iodepth=64 -ioengine=libaio -direct=1 -
>
> rw=write -bs=1m -size=2g -numjobs=4 -runtime=20 --group_reporting -
>
> name=test-write

```
# 测试顺序读取：
fio --filename=/testio/test -iodepth=64 -ioengine=libaio --direct=1 --
rw=read --bs=1m --size=2g --numjobs=4 --runtime=10 --group_reporting --
name=test-read

# 16k，混合读写模式（随机读写），70%读取，30%写入：
fio --filename=/dev/sdb --direct=1 --rw=randrw --ioengine=libaio --bs=4k
--rwmixread=70 --iodepth=16 --numjobs=16 --runtime=60 --group_reporting --
name=73test

# 重要参数解读：
--filename 需要压测的磁盘或者测试文件。
--direct=1 是否绕过文件系统缓存
-ioengine=libaio 采用异步或者同步IO
-iodepth=64 IO队列深度。一次发起多少个IO请求，一般SSD或者flash可以较大。
--numjobs=16 测试并发线程数。在RAID10或Raid5可加大参数。
--rwmixread=70 混合读写，read的比例。一般读写比例28或者37。
--group_reporting 统计汇总结果展示。
--name 起个名。
--rw=randrw 测试类型.

# 结果解读 通过控制变量多次进行压测寻找最终的瓶颈临界值
需要关注的值
IOPS: io的频率
BW ： 吞吐量
min:最小延时
max：最大延时
avg：平均延时
stdev: 标准差（离散程度）

write: IOPS=5883, BW=91.9MiB/s (96.4MB/s)(1841MiB/20020msec)
lat (usec): min=1677, max=78137, avg=43413.06, stdev=7210.57
clat percentiles (usec):
| 1.00th=[23462], 5.00th=[25297], 10.00th=[26084], 20.00th=[27657],
| 30.00th=[28967], 40.00th=[30278], 50.00th=[31851], 60.00th=[33162],
| 70.00th=[34866], 80.00th=[36963], 90.00th=[40109], 95.00th=[42730],
| 99.00th=[49021], 99.50th=[51119], 99.90th=[59507], 99.95th=[61604],
| 99.99th=[64226]
```

258ef5cd5b29.png&title=1.MySQL%E7%94%9F%E4%BA%A7%E4%B8%8A%E7%BA%BF%E5%89%8D%E5%87%86%E5%A4%87%E2%88%9