# 内存检查

## free

total：内存总数

used：已经使用的内存数

free：空闲内存数

shared：当前已经废弃不用，总是0

buffers：缓存内存数

cached：page缓存内存数

(-buffers/cache)used：内存数：2062M（指的第一部分Mem行中的used-buffers-cached）

(+buffers/cache)free内存数：5798M（指的第一部分Mem行中的free+buffers+cached）

-buffers/cache：反映的是被程序实实在在吃掉的物理内存，而+buffers/cache反映的是可以挪用的内存总数

说明：

1. 一般情况下，内存free低于20%，或者cached超过100G（资源少的系统可能20G就达到阈值），或者swap空间开始被使用，如果不是内存泄漏则表示环境内存资源紧俏，需要进行扩容；
2. 清理swap的方法有：

重启和使用swap换出

说明：

swapon -s查看swap挂载分区

swapoff /dev/xxxx清理swap

swapon -a重新挂载

swapoff时注意清理缓存的速度，计算总共需要花多久

1. 建议操作系统的内核参数vm.swappiness为0，检查方法是sysctl -p，若不为0，则执行echo “vmswappiness=0” >>/etc/sysctl.conf，然后执行sysctl -p生效。

## vmstat

# OS有大量空闲内存，却发生SWAP

## 故障描述

OS有大量的空闲内存，但是却发生了SWAP。

## 故障分析

1. 检查mysql的配置参数

结论：无异常

1. 检查系统的vm.swappiness

结论：该值为0，没问题

1. 检查系统的NUMA分配方式

结论：系统没有关闭NUMA，NUMA问题导致其中一个CPU可分配的内存远小于另一个，那么这个CPU上如果要申请大内存，容易发生SWAP。

说明：因为开启了NUMA，导致别的节点可以使用本地节点的内存，但是如果这个节点需要申请大内存那么就很容易使用swap分区，这个是基于这样一个前提，mysql可能会并发执行大结果查询，可能每个节点都需要很大内存，无法跨节点访问其他节点的内存了（其他节点的可能自己或者给别的节点已经使用消耗殆尽），导致只能使用swap分区了，因此基于mysql这种业务特点干脆直接关闭，自己使用自己的。

## 解决方法

1. 关闭NUMA
2. NUMA的内存分配方式修改为interleave（numactl --interleave=all）
3. 开启mysql的innodb\_numa\_interleave选项
4. 使用HugePage

# DBAgent将DB杀掉

## 问题描述

在巡检的时候发现日志中有DBAgent将DB杀死的情况。

## 故障分析

1. 查看系统的IO压力是否很大：iostat显示iowait很大，但是utils很低
2. sar -r/sar -B查看内存是否够用，显示pgscan很高
3. 检查一下page cache是否没有被清理

注：pgscank和pgscan分别是Linux内核kswapd进程和应用扫描page cache的频率，出现就证明free内存已经不够了，这个不会有明显的影响，因为DB的数据页不使用page cache，而是使用ODirect的方式。

drop table是用来主动扫描并清理page cache的，释放出来的内存就会回到free中。

## 解决方法

# 内存占用大发生OMM

## 问题描述

系统发生OOM。

## 故障分析

1. 查看巨页大小，发现占用130G内存
2. bufferpoll内存占用也比较大，自然发生OOM

## 解决方法

# NUMA导致内存页置换

## 问题描述

## 故障分析

numactl --hardware

## 解决方法

echo \*\*\*\* > /proc/sys/vm/min\_free\_kbytes

注：采集页面置换（每1s采集）：sar -B 1