Atitit 数据库性能提升机制 临时表 内存表

[1.1. 临时表与内存表区别 1](#_Toc9179)

[二、 临时表的应用 1](#_Toc14903)

[1.2. 4、对于重启造成的数据丢失，有以下的解决办法： 1](#_Toc16566)

## 临时表与内存表区别

内存表分为2种，但共同点是，重起数据库以后，内存中的数据全部丢失,内存表的功能有部分的限制，有些属性不能像正常表一样使用，所以请大家使用的时候谨慎参照官方文档.下面只是抛砖引玉.   
1.临时表:表建在内存里，数据在内存里   
2.内存表:表建在磁盘里，数据在内存里

## 临时表的应用

1. 当工作在十分大的表上运行时，在实际操作中你可能会需要运行很多的相关查询，来获的一个大量数据的小的子集。较好的办法，不是对整个表运行这些查询，而是让MySQL每次找出所需的少数记录，将记录选择到一个临时表，然后对这些表运行查询。  
   1）、当某一个SQL语句关联的表在2张及以上，并且和一些小表关联。可以采用将大表进行分拆并且得到比较小的结果集合存放在临时表中。  
   2）、程序执行过程中可能需要存放一些临时的数据，这些数据在整个程序的会话过程中都需要用的等等。  
   3）、临时表默认的是MyISAM,但是可以修改。  
   4)、可以把一些经常访问的数据放到临时表中，这样访问时会快一些,因为数据是在服务器内存中,另外每次查询的时候，**[数据库](http://lib.csdn.net/base/14" \o "MySQL知识库" \t "https://yq.aliyun.com/articles/_blank)**都需要生成一些临时数据在临时表里

## 4、对于重启造成的数据丢失，有以下的解决办法：

　a、在任何查询之前，执行一次简单的查询，判断heap表是否存在数据，如果不存在，则把数据重新写入，或者DROP表重新复制某张表。这需要多做一次查询。不过可以写成include文件，在需要用该heap表的页面随时调用，比较方便。  
　b、对于需要该heap表的页面，在该页面第一次且仅在第一次查询该表时，对数据集结果进行判断，如果结果为空，则需要重新写入数据。这样可以节省一次查询。  
　c、更好的办法是在mysql每次重新启动时自动写入数据到heap，但是需要配置服务器，过程比较复杂，通用性受到限制。

加载数据

MySQL启动时，加入--init-file选项，将下列命令加入到这个文件中，保证启动后内存表中有数据。

INSERT INTO ... SELECT

LOAD DATA INFILE

###### **MySql 内存表使用**

内存表使用哈希散列索引把数据保存在内存中，因此具有极快的速度，适合缓存中小型数据库，但是使用上受到一些限制，以下是蓝草使用的一些感受。  
  
1、heap对所有用户的连接是可见的，这使得它非常适合做缓存

注意：  
    当内存表中的数据大于max\_heap\_table\_size设定的容量大小时，mysql会转换超出的数据存储到磁盘上，因此这是性能就大打折扣了，所以我们还需要根据我们的实际情况调整max\_heap\_table\_size，例如在.cnf文件中[mysqld]的下面加入：  
        max\_heap\_table\_size = 2048M  
    另外在建表语句中还可以通过MAX\_ROWS来控制表的记录数。

持久化机制，定时序列化到文件，或者当时插入到数据库物理表

时使用MEMORY

临时使用、不重要的数据，例如网站的会话管理和缓存。可接受数据丢失。

发挥其访问快、低延迟特性

只读或者大部分是读操作，不适合大量写操作。

性能特性

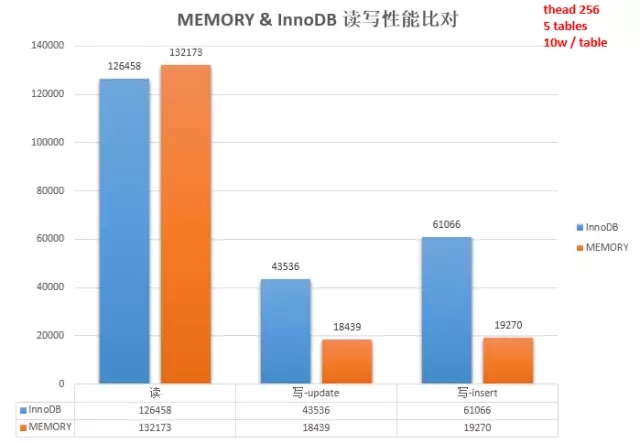
内存表受限于单线程执行和表级锁引起的争用。在负载增加时限制了可扩展性，特别是写操作。

尽管内存表在内存中处理，在繁忙的服务网上，并不一定比InnoDB表更快，如一般查询或者读写场景。特别是，多个会话执行update操作会造成性能低下。

根据不同的SQL查询，需要创建内存表的默认hash索引（基于键的单个值查询），或者B-tree索引（等值查询、不等值查询或者范围查询）。

测结果

通过sysbench压测，InnoDB 写性能完爆MEMORY引擎，MEMORY的读性能略高于InnoDB。InnoDB的综合性能更好，足够满足日常使用。考虑到MEMORY引擎的雷点太多，建议不使用MEMORY引擎。

[](http://attachbak.dataguru.cn/attachments/portal/201711/08/152532i6xgraigubjigjgb.jpg)

MySQL内存表自动备份和还原\_Historical\_Sky\_新浪博客.html