MyISAM 和 InnoDB 讲解

InnoDB 和 MyISAM 是许多人在使用 MySQL 时最常用的两个表类型，这两个表类型各有优劣，视具体应用而定。基本的差别为：MyISAM 类型不支持事务处理等高级处理，而 InnoDB 类型支持。MyISAM 类型的表强调的是性能，其执行速度比 InnoDB 类型更快，但是不提供事务支持，而 InnoDB 提供事务支持以及外部键等高级数据库功能。

以下是一些细节和具体实现的差别：

◆1.InnoDB 不支持 FULLTEXT 类型的索引。

◆2.InnoDB 中不保存表的具体行数，也就是说，执行 select count(*) from table 时，InnoDB 要扫描一遍整个表来计算有多少行，但是 MyISAM 只要简单的读出保存好的行数即可。注意的是，**当 count(*)语句包含 where 条件时，两种表的操作是一样的。**

◆3.对于 AUTO_INCREMENT 类型的字段，InnoDB 中必须包含只有该字段的索引，但是在 MyISAM 表中，可以和其他字段一起建立联合索引。

◆4.DELETE FROM table 时，InnoDB 不会重新建立表，而是一行一行的删除。

◆5.LOAD TABLE FROM MASTER 操作对 InnoDB 是不起作用的，解决方法是首先把 InnoDB 表改成 MyISAM 表，导入数据后再改成 InnoDB 表，但是对于使用的额外的 InnoDB 特性(例如外键)的表不适用。

另外，InnoDB 表的行锁也不是绝对的，假如在执行一个 SQL 语句时 MySQL 不能确定要扫描的范围，InnoDB 表同样会锁全表，例如 update table set num=1 where name like “%aaa%”

两种类型最主要的差别就是 Innodb 支持事务处理与外键和行级锁。而 MyISAM 不支持.所以 MyISAM 往往就容易被人认为只适合在小项目中使用。

作为使用 MySQL 的用户角度出发，Innodb 和 MyISAM 都是比较喜欢的，如果数据库平台要达到需求：99.9%的稳定性，方便的扩展性和高可用性来说的话，MyISAM 绝对是首选。

原因如下：

1、平台上承载的大部分项目是读多写少的项目，而 MyISAM 的**读性能**是比 Innodb 强不少的。

2、MyISAM 的索引和数据是分开的，并且索引是有压缩的，内存使用率就对应提高了不少。能加载更多索引，而 Innodb 是索引和数据是紧密捆绑的，没有使用压缩从而会造成 Innodb 比 MyISAM 体积庞大不小。

3、经常隔 1，2 个月就会发生应用开发人员不小心 update 一个表 where 写的范围不对，导致这个表没法正常用了，这个时候 MyISAM 的优越性就体现出来了，随便从当天拷贝的压缩包取出对应表的文件，随便放到一个数据库目录下，然后 dump 成 sql 再导回到主库，并把对应的 binlog 补上。如果是 Innodb，恐怕不可能有这么快速度，别和我说让 Innodb 定期用导出 xxx.sql 机制备份，因为最小的一个数据库实例的数据量基本都是几十 G 大小。

4、从接触的应用逻辑来说，select count(*) 和 order by 是最频繁的，大概能占了整个 sql 总语句的 60%以上的操作，而这种操作 Innodb 其实也是会锁表的，很多人以为 Innodb 是行级锁，那个只是 where 对它主键是有效，非主键的都会锁全表的。

5、还有就是经常有很多应用部门需要我给他们定期某些表的数据，MyISAM 的话很方便，只要发给他们对应那表的 frm.MYD,MYI 的文件，让他们自己在对应版本的数据库启动就行，而 Innodb 就需要导出 xxx.sql 了，因为光给别人文件，受字典数据文件的影响，对方是无法使用的。

6、如果和 MyISAM 比 insert 写操作的话，Innodb 还达不到 MyISAM 的写性能，如果是针对基于索引的 update 操作,虽然 MyISAM 可能会逊色 Innodb,但是那么高并发的写，从库能否追的上也是一个问题，还不如通过多实例分库分表架构来解决。

7、如果是用 MyISAM 的话，merge 引擎可以大大加快应用部门的开发速度，他们只要对这个 merge 表做一些 select count(*)操作，非常适合大项目总量约几亿的 rows 某一类型(如日志，调查统计)的业务表。

当然 Innodb 也不是绝对不用，用事务的项目就用 Innodb 的。另外，可能有人会说你 MyISAM 无法抗太多写操作，但是可以通过架构来弥补。