1. **数据存储区别**
2. **事务处理方面**

**3）select\update\delete\  insert等操作**

**4）  对auto\_increment的操作**

**5）关于主键**

**6）count(\*)**

1. **锁**

**1)数据存储区别**

先看几个概念：

**堆表-------------heap table(数据无序的存储)**

**索引组织表-------表和主键存在一起**

cluster table--一张表存储在一个段

分区表--一-------张表存储到不同段

Myisam：

属于**堆表**，在磁盘上分三个文件进行存储，每个文件名以表名开头，扩展名指出文件类型。分别是.frm文件用于是表结构定义，.myd用于存储数据，.myi用于存储索引

Innodb：

属于**索引组织表**。其有两种存储方式，共享表空间存储和多表空间存储。两种存储方式的表结构和myisam一样，都以表名开头，扩展名是.frm。如果使用是共享表空间，那么所有表的数据文件和索引文件都保存在一个表空间里，一个表空间可以有多个文件。通过innodb\_data\_file\_path和innodb\_data\_home\_dir参数设置共享表空间的位置和名字，一般共享表空间的名字叫ibdata1-n。如果是使用多表空间，那么每个表都有一个表空间文件用于存储每个表的数据和索引，文件名以表名开头，以ibd为扩展名。共享表空间优点是数据和文件放在一起方便管理。**缺点是大文件中多表及索引混合存储，对于做了大量删除操作后表空间将有大量空隙，特别不适合于统计分析、日志系统该类应用。**多表空间存储亦称之为独占表空间存储，优点是独立、单表可在不同数据库间移动，空间方便回收。缺点：单表空间过大的话就不大好了。总体来说，使用独占表空间性能和效率都会更高些。

**2）  事务处理方面**

Myisam

强调的是性能，其执行速度比innodb更快。但不提供事务支持

Innodb

提供事务支持、外部键等高级数据库功能

**3）select\update\delete\  insert等操作**

Myisam：

如果执行大量select，myisam是更好的选择

Innodb ：

如果大量insert或者update，出于性能考虑应该使用innodb表(**WHY?)**；delete from table时，innodb不会重新建立表，而是一行一行的删除；load data  from master操作对innodb是不起作用的。解决方法是先把innodb表改成myisam表，导入数据后再改成innodb表，但是对于使用额外的innodb特性（例如外键）的表不适用。在innodb上如果要清空保存有大量数据的表，最好不要使用这个命令。(推荐使用truncate table，不过需要用户有drop此表的权限)

**4）  对auto\_increment的操作**

Myisam：

可以用alter table来修改、重置auto\_increment；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中，可以和其他字段一起建立联合索引；myisam更好更快的auto\_increment处理

Innodb：

如果你为一个表指定auto\_increment列，在数据词典中innodb表会包含一个自动增长计数器，它被用在给该列赋值。该计数器存在主内存中，而不存在磁盘上。myisam引擎的自动增长列必须是索引，如果是组合索引，自动增长可以不是第一列，他可以根据前面几列进行排序后递增。innodb引擎的自动增长咧必须是索引，如果是组合索引也必须是组合索引的第一列。

**5）关于主键**

Myisam

允许没有任何索引和主键的表存在，索引都是保存行的地址。

Innodb

引擎如果没有设定主键或者非空唯一索引，就会自动生成一个6字节的主键(用户不可见)。数据是主索引的一部分，附加索引保存的是主索引的值。

**6）count(\*)**

myisam:

对于不加where的count，myisam是有保存表行数的，它直接去读取即可。

innodb：

不保存具体记录行数，count(\*)操作时，会全表扫描一次进行统计

对于有where条件查询的count(\*)两者是一样的

**7）锁**

相关知识：mysql常见的三种锁级别——表级锁、页面锁、行级锁；其中表级锁有两种模式——表共享读锁和表独占写锁。

myisam：

**表级锁**。对myisam表进行读操作的时候，它不会阻塞其他用户对同一表的读请求，但会阻塞对同一表>的写操作；而对myisam表进行写操作的时候，它会阻塞其他用户对同一表的读、写请求。但在一定条>件下MyISAM表也支持查询和插入的操作的并发进行，其机制是通过控制一个系统变量（concurrent\_insert）来进行的。通过研究表明，写进程将先获得锁（即使读请求先到锁等待队列）。但这也造成一>个很大的缺陷，即大量的写操作会造成查询操作很难获得读锁，从而可能造成永远阻塞。所幸我们可>以通过一些设置来调节MyISAM的调度行为。我们可通过指定参数low-priority-updates。关于和concurrent\_insert 和low-priority-updates的说明和用法，请参见http://hi.baidu.com/lampcool/item/2ca40b1a77362b04e65c36c1

innodb：

提供**行锁**(locking on row level),另外，InnoDB表的行锁也不是绝对的，如果在**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 “a%”。

行级锁的优点如下：

1）当很多连接分别进行不同的查询时减小LOCK状态。

2）如果出现异常，可以减少数据的丢失。因为一次可以只回滚一行或者几行少量的数据。

**行级锁的缺点如下**：

1）比页级锁和表级锁要占用更多的内存。

2）进行查询时比页级锁和表级锁需要的I/O要多，所以我们经常把行级锁用在写操作而不是读操作。

3）容易出现死锁。

**注意：innodb不能确定操作的行，这个时候就使用的意向锁，也就是表锁**