1， 事务处理

innodb 支持事务功能，myisam 不支持。

Myisam 的执行速度更快，性能更好。

2，select ,update ,insert ,delete 操作

MyISAM： 如果执行大量的SELECT，MyISAM是更好的选择

InnoDB： 如果你的数据执行大量的INSERT或UPDATE，出于性能方面的考虑，应该使用InnoDB表

3，锁机制不同

InnoDB 为行级锁，myisam 为表级锁。

注意：当数据库无法确定所找的行时，也会变为锁定整个表。

如：  update table set num = 10 where username like "%test%";

4，查询表的行数不同

MyISAM： select count(\*) from table,MyISAM只要简单的读出保存好的行数，注意的是，当count(\*)语句包含   where条件时，两种表的操作是一样的

InnoDB  ：  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

5，物理结构不同

MyISAM ：每个MyISAM在磁盘上存储成三个文件。第一个文件的名字以表的名字开始，扩展名指出文件类型。

   .frm文件存储表定义。

   数据文件的扩展名为.MYD (MYData)。

  索引文件的扩展名是.MYI (MYIndex)

InnoDB： 基于磁盘的资源是InnoDB表空间数据文件和它的日志文件，InnoDB 表的大小只受限于操作系统文件的大小，一般为 2GB

6，anto\_increment 机制不同

 更好和更快的auto\_increment处理

其他：为什么 MyISAM 会比Innodb 的查询速度快。

INNODB在做SELECT的时候，要维护的东西比MYISAM引擎多很多；

1）数据块，INNODB要缓存，MYISAM只缓存索引块，  这中间还有换进换出的减少；

2）innodb寻址要映射到块，再到行，MYISAM 记录的直接是文件的OFFSET，定位比INNODB要快

3）INNODB还需要维护MVCC一致；虽然你的场景没有，但他还是需要去检查和维护

MVCC ( Multi-Version Concurrency Control )多版本并发控制

InnoDB：通过为每一行记录添加两个额外的隐藏的值来实现MVCC，这两个值一个记录这行数据何时被创建，另外一个记录这行数据何时过期（或者被删除）。但是InnoDB并不存储这些事件发生时的实际时间，相反它只存储这些事件发生时的系统版本号。这是一个随着事务的创建而不断增长的数字。每个事务在事务开始时会记录它自己的系统版本号。每个查询必须去检查每行数据的版本号与事务的版本号是否相同。让我们来看看当隔离级别是REPEATABLE READ时这种策略是如何应用到特定的操作的：

　　SELECT InnoDB必须每行数据来保证它符合两个条件：

　　1、InnoDB必须找到一个行的版本，它至少要和事务的版本一样老(也即它的版本号不大于事务的版本号)。这保证了不管是事务开始之前，或者事务创建时，或者修改了这行数据的时候，这行数据是存在的。

　　2、这行数据的删除版本必须是未定义的或者比事务版本要大。这可以保证在事务开始之前这行数据没有被删除。

所以，InnoDB 用于数据完整性/写性能要求比较高的应用. MyISAM 适合查询应用。

附：

最直接的方式是在创建表时指定存储引擎的类型，向下面这样:

CREATE TABLE mytable (id int, titlechar(20)) ENGINE = INNODB

修改表的存储引擎：

ALTER TABLE engineTest ENGINE = INNODB；

修改默认存储引擎：

在mysql配置文件（linux下为/etc/my.cnf），在mysqld后面增加default-storage-engine=INNODB即可。

但是如果表建立的时候是MyISAM，要更改整个数据库表的存储引擎，一般要一个表一个表的修改，比较繁琐，可以采用先把数据库导出，得到SQL，把MyISAM修改成INNODB，再导入的方式。