

举例：select \* from table where id =1;

## mysql执行顺序

1. 连接器，负责管理连接和权限判断
2. 分析器，语法词法分析
3. 优化器，生成执行计划，并选择最佳索引
4. 执行器
   1. 加载缓存数据，从磁盘文件(ibd)加载id为1的**整页**数据到**buffer pool(缓存池)**
   2. 如果是更新语句则写入undo日志以便回滚。
   3. 更新内存数据**到buffer pool(缓存池)**
   4. **写redo日志缓存(redo log buffer)**
   5. redo日志顺序写入磁盘，准备提交数据(在哪页做了什么修改)
   6. 准备提交事务，binlog日志写入磁盘(binlog适用于恢复数据库磁盘里的数据)
   7. 写入commitid到redo日志，提交事务完成。该标识为了保证事务提交后redolog、binlog数据一致。(如果事务提交成功，但是此时**buffer pool**并没写入磁盘而服务器宕机，则可以使用redo日志的数据恢复磁盘ibd文件里面的数据)
   8. 系统空闲的时候以page为单位随机写入磁盘

备注：

redo日志文件顺序写: redo log是一个或几个预先分配好磁盘空间的文件，写入永远都是在文件末尾追加。

ibd数据文件随机读写: 不同的数据表有不一样的ibd文件，比如执行10条sql，修改不同的表是需要修改不同表的ibd文件，不能实现顺序写文件的效果。先写redo log后刷新数据表文件的机制叫WAL机制(Write-Ahead Logging)，效率更高.