事务

原子性(atomicity):所有的sql都是可以执行的；

一致性(cosistentcy):事务在提交之前是安全的；

隔离性(isolation):事务锁的概念；

持久性(durability):提交后就永久保存在数据库中；

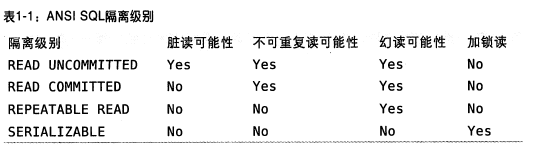
隔离级别

未提交读(read uncommitted):事务可以读取未提交的数据，可能产生脏读(Dirty Read)；一般不推荐；

提交读(read committed):一个事务从开始到提交前，所做的任何修改其他事务是不可见的；也叫不可重复读:因为执行两次同样的查询，可能得到不一样的结果。一般默认级别；

可重复读(repeatable read):多次读取结果一致，但是可能产生幻读:读取多出几条之前不存在的记录；可重复读是MySql的默认事务隔离级别；

可串行化(serializable):Serializable是最高的隔离级别，强制事务串行执行，避免幻读的出现，因为每一行都加锁，并发性差，默认不推荐。



死锁

死锁是指两个或者多个事务在同一资源上相互占用，并请求锁定对方占用的资源，而锁只有在commit或rollback的时候才会释放，从而导致恶性循环的现象。

解决办法：将持有最少行级排他锁的事务进行回滚。(这是相对比较简单的死锁回滚算法)。

事务日志

使用事务日志，存储引擎在修改表的数据时只需要修改器其内存拷贝，再把该修改行为记录到持久在硬盘的事务日志中，而不用每次都将修改的数据本身持久到磁盘。

多版本并发控制(MVCC)

MVCC是行级锁的一个变种，它在很多情况下避免了加锁操作，开销很低。是通过保存数据在某个时间点的快照来实现的。根据事务开始的时间不同，每个实物对同一张表，同一时刻看到的数据可能是不一样的。

性能定义为完成某件任务所需要的时间度量，换句话说，性能即相应时间

数据库优化：

使用date,time,datetime来存储时间，用整形来存储IP(INET\_ATON(),INET\_NTOA())

使得列不能为NULL，优化效果一般

整数类型:TINYINT(8)，SMALLINT(16)，MEDIUMINT(24)，INT(32)，BIGINT(64) [**TINGYINT UNSIGN**]

实数(小数)类型:DECIMAL(18,9);4个字节存储9个数字，小数点占一个字节，DOUBLE，FLOAT

字符型:VARCHAR(不定长)，CHAR(定长)

TEXT(字符)和BOLB(二进制)大范围的字符串数据类型

时间:DATETIME(YYYYMMDDHHMMSS)的整数，TIMESTAMP

其他：BIT，SET

范式化设计的schema的缺点是通常需要关联，反范式化的schema因为所有的数据都在一张表中，可以很好的避免关联

**set autocommit = 1; --自动提交**

**set session transaction isolation level read committed; --设置隔离级别为read committed**

**show table status like 'user' \G; --显示表的相关信息**

**check table mytable; --检查表的错误**

**repair table mytable; --修复表的错误**

**alter table mytable engine = InnoDB; --转换表的引擎，慎用**

**insert into mynewtable select \* from myoldtable; --快速复制表**

**show create table; --查看表的数据类型**

**insert into mytable(columns) values(),(),(); --可以同时增加多个value**

**create table enum\_test(e ENUM('fish','apple','dog' not null)); --创建枚举**

**insert into enum\_test(e) values('fish'),('dog'),('apple'); --插入枚举，实际保存的是1,2,3**