## 范式和反范式

第一范式：原子性

第二范式：符合第一范式的基础上，完全依赖于主键，而不是部分依赖于主键（针对符合主键来说）

第三范式：符合一二范式的基础上，不存在依赖传递

范式的优点：数据没有冗余，更新容易。缺点：要进行多表连接，性能差

反范式的：优点，数据冗余使得数据的可读性强（不需要连接很多表），性能也比较好。但是数据冗余占用磁盘空间。同时数据维护比较难。

## 事务的概念和4个特性

一系列操作，是一个整体，要么都成功，要么都失败

原子性：要么都成功，要么都失败。A给b转账。A减去100，B增加100.这两个动作是一个整体，具有不可再分的原子性

一致性：类使用化学反应中的质量守恒定律。多个人之间转账，总共有1万元，不管怎么转，都所有人的账户总额就是1万元

隔离性: 针对并发操作来说的。同一个时间，多个用户对同一个数据进行操作，如果没有设置隔离性，就会相互影响。多个事务之间，谁先谁后的问题

持久性：我们对数据进行更改后，结果是写进了外部的磁盘的，而不是值更改内存中的结果

## 事务的4中隔离级别

### 发生的问题

脏读：事务A读取了一个数据。这个数据是事务B进行了修改，但是没有提交。后面事务B进行事务的回滚，这时候对于事务A来说，就是读了一个脏数据

不可重复读：事务A多次读取同一个数据，但是每次读取的结果不一样（付款），因为其他事务对这个数据进行了修改并提交。

幻读（幻觉）: 事务A多次查询，每次的查询结果是不一样的。因为在多次查询的过程中，其他事务进行了插入操作。

脏读和不可重复读：脏读是一个事务读取了另一个事务未提交的数据。不可重复读是一个事务多次读取同一个数据，但是结果不一样，原因是另一个事务对这个数据进行了修改，并提交了。

幻读和不可重复读：幻读是多次读取，结果集不一样（其他事务的插入导致的），幻读是多次读取，同一个数据不一致，是其他事务修改提交导致的。一个针对的是同一个数据，一个针对的是所有的数据，一个结果集。

### 隔离级别

读取未提交：是指可以读取没有提交的数据。这样的并发时候的性能高。但是隔离级别最低，一个事务，能读取其他事务未提交的数据。从而导致脏读

Read commit：读取已提交，一个事务只能读取其他事务已经提交的数据，不能读取其他事务未提交的数据。换个角度就是一个事务对数据的修改，在提交之前，对别的事务都是不可见的。（可以防止脏读，但是不能防止不可重复读）

Read commit：可重复读，能够保证一个事务，多次读取，结果都一样。（在读取数据前，对数据加上一个共享锁，当事务结束时候，才解锁）。但是共享锁，别的事务虽然不能对数据进行修改，但是可以对数据进行插入。这就会导致，多次读取的结果集合不一样，也就是幻读的问题。

Serializable（可串行化，序列化）：事务和事务执行的时间段没有交集。必须一个执行完才能继续执行另一个事务。这个并发性能就很低了。

Mysql默认的隔离级别是可重复读。

Sql server和Oracle的默认级别是读取已提交。

## 设置和察看

设置隔离级别

Set session transaction isolation level read uncommitted；

Set session transaction isolation level read committed;

Set global transaction isolation level repeatable read;

Set global transaction isolation level serializable;

察看会话隔离级别（就是察看会话变量）

Select @@transaction\_isolation;

Select @@global.transaction\_isolation;

## 变量

全局变量：察看show variables [ like匹配模式 ]；select @@name； select @@global.name

设置，局部设置 set @@name =

全局设置set @@global.name =

会话变量:

设置 set @name = ‘tom‘

Select @name := my\_name from s where id = 1;

Select my\_name from s where id = 1 into @name;

察看select @name;