MySQL事务管理

1、数据库四大特性：ACID

A 原子性(Atomcity)【事务包含的所有操作要么全部成功，要么全部失败回滚。失败回滚的操作事务，将不能对事务有任何影响。】

C 一致性(Consittent)【指事务必须使数据库从一个一致性状态变换到另一个一致性状态，也就是说一个事务执行之前和执行之后都必须处于一致性状态。】

【一致读】【一致写】

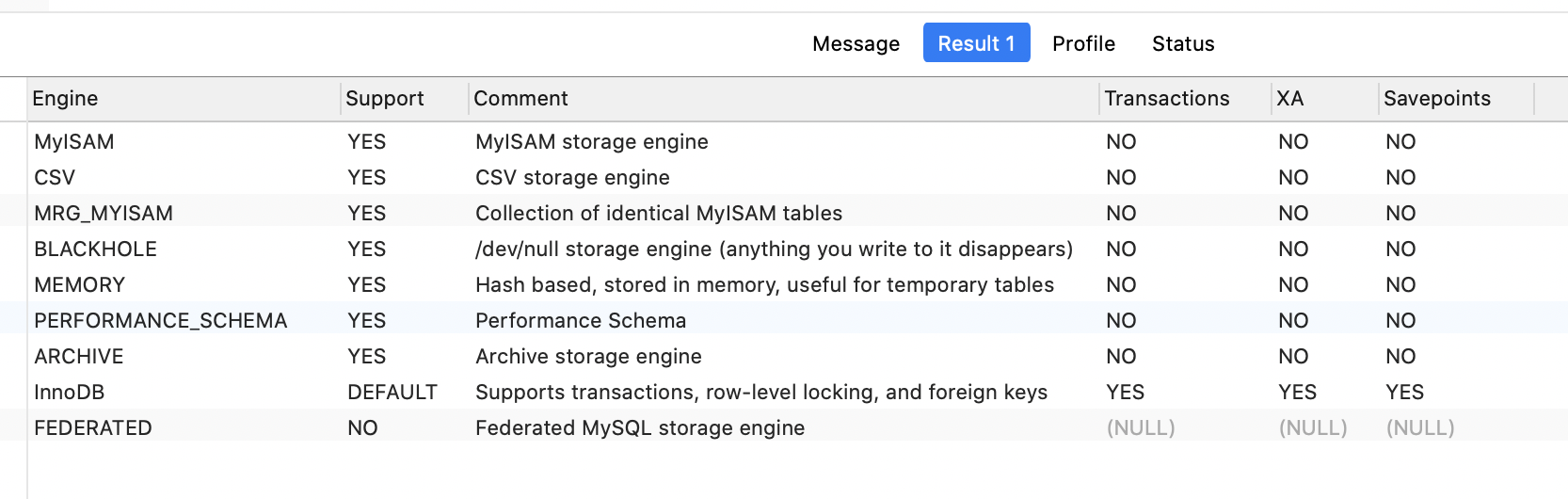
I 隔离性(Isolation)【隔离性是指当多个用户并发访问数据库时，比如同时访问一张表，数据库每一个用户开启的事务，不能被其他事务所做的操作干扰，

多个并发事务之间，应当相互隔离。】

D 持久性(Durable)【持久性是指事务的操作，一旦提交，对于数据库中数据的改变是永久性的，即使数据库发生故障也不能丢失已提交事务所完成的改变。】

2、MySQL有哪些数据引擎？

> show engines;



1. 默认增删改查自动提交，开启事务的命令：begin/start transaction，结束事务的命令：commit/rollback；
2. 读一致性：脏读【读取到其他事务的未提交数据】、不可重复读【读取到其他事务已提交的数据，导致多次读取数据不一致】、幻读【读取到其他视图已提交的新增(insert)的数据】

2、数据库在服务器上的存储方式

最上层用于连接、线程处理；第二层中包含了大多数的核心服务，包括了对SQL的解析、分析、优化和缓存等功能，存储过程、触发器和视图都是在这里实现的；

而第三层就是真正负责数据的存储和提取的存储引擎，例如：InnoDB、MyISAM等，文中对存储引擎的介绍都是对 InnoDB 实现的分析。

3、数据存储结构

在整个数据库体系结构中，我们可以使用不同的存储引擎来存储数据，而绝大多数存储引擎都以二进制的形式存储数据。

在 InnoDB 存储引擎中，所有的数据都被逻辑地存放在表空间中，表空间（tablespace）是存储引擎中最高的存储逻辑单位，

在表空间的下面又包括段（segment）、区（extent）、页（page）：

同一个数据库实例的所有表空间都有相同的页大小；默认情况下，表空间中的页大小都为 16KB，当然也可以通过改变 innodb\_page\_size 选项对默

认大小进行修改，需要注意的是不同的页大小最终也会导致区大小的不同：

从图中可以看出，在 InnoDB 存储引擎中，一个区的大小最小为 1MB，页的数量最少为 64 个。

三、随机读取和顺序读取

在我们常用的sql理解中，数据是以行的形式读取出来的，其实不然，通过上述的结构，我们可以了解到，单次从磁盘读取单位是页，而不是行，也就是说，

你即便只读取一行记录，从磁盘中也是会读取一页的，当然了单页读取代价也是蛮高的，一般都会进行预读，这些后续再说。

1、数据的读取路径

关系型数据库管理系统最重要的一个目标就是，确保表或者索引中的数据是随时可以用的。那么为了尽可能的实现这个目标，会使用内存中的缓冲池来最小化磁盘活动。

每一个缓冲池都足够大，大到可以存放许多页，可能是成千上万的页。

缓冲池管理器将尽力确保经常使用的数据被保存于池中，以避免一些不必要的磁盘读。如果一个索引或者表页在缓冲池中被找到，那么将会处理很快。

如果在缓冲池中，没有找到数据，会从磁盘服务器的缓冲区里面去读取。

磁盘服务器的缓冲区，如同数据库的缓冲池读取一样，磁盘服务器试图将频繁使用的数据保留在内存中，以降低高昂的磁盘读取成本。这个读取成本大概会在1ms左右。

如果磁盘服务器的缓冲池中依然没有找到数据，此时就必须要从磁盘读取了，此时读取又分区随机读取和顺序读取。

2、随机I/O

我们必须记住一个页包含了多条记录，我们可能需要该页上的所有行，也可能是其中一部分，或者是一行，但所花费的成本都是相同的，读取一个页，需要一次随机I/O,大约需要10ms的时间。

时间组成：

我们可以看到，一次随机IO需要耗时的时间还是很久的，10ms对计算机来说是一个很长的时间节点了。

3、顺序读取

如果我们需要将多个页读取到缓冲池中，并按顺序处理它们，此时读取的速度回变的很快，具体的原理，在B树索引中也有过介绍，此时读取每个页面（4kb）所花费的时间大概为0.1ms左右，这个时间消耗对随机IO有很大的优势。

以下几种情况，会对数据进行顺序读取。

全表扫描

全索引扫描

索引片扫描

通过聚蔟索引扫描表行

顺序读取有两个重要的优势：

同时读取多个页意味着平均读取每个页的时间将会减少。在当前磁盘服务器条件下，对于4kb大小的页而言，

这一值可能会低于0.1ms（40MB/s）

由于数据库引擎知道需要读取哪些页，所有可以在页被真正请求之前就提前将其读取进来，我们称为预读。

四、聚集索引和辅助索引

数据库中的 B+ 树索引可以分为聚集索引（clustered index）和辅助索引（secondary index），

它们之间的最大区别就是，聚集索引中存放着一条行记录的全部信息，而辅助索引中只包含索引列和一个用于查找对应行记录或者主键记录的指针。