1. 唯一索引比普通索引快吗, 为什么

唯一索引不一定比普通索引快, 还可能慢.

1. 查询时, 在未使用`limit 1`的情况下, 在匹配到一条数据后, 唯一索引即返回, 普通索引会继续匹配下一条数据, 发现不匹配后返回. 如此看来唯一索引少了一次匹配, 但实际上这个消耗微乎其微.
2. 更新时, 这个情况就比较复杂了. 普通索引将记录放到`change buffer`中语句就执行完毕了. 而对唯一索引而言, 它必须要校验唯一性, 因此, 必须将数据页读入内存确定没有冲突, 然后才能继续操作. 对于**写多读少**的情况, 普通索引利用`change buffer`有效减少了对磁盘的访问次数, 因此普通索引性能要高于唯一索引.

2. MySQL由哪些部分组成, 分别用来做什么

1. Server
    - 连接器: 管理连接, 权限验证.
    - 分析器: 词法分析, 语法分析.
    - 优化器: 执行计划生成, 索引的选择.
    - 执行器: 操作存储引擎, 返回执行结果.
2. 存储引擎: 存储数据, 提供读写接口.

3. MySQL查询缓存有什么弊端, 应该什么情况下使用, 8.0版本对查询缓存有什么变更.

- 查询缓存可能会失效非常频繁, 对于一个表, 只要有更新, 该表的全部查询缓存都会被清空. 因此对于频繁更新的表来说, 查询缓存不一定能起到正面效果.
- 对于读远多于写的表可以考虑使用查询缓存.

- 8.0版本的查询缓存功能被删了 (￣.￣).

4. MyISAM和InnoDB**的区别有哪些

- InnoDB支持事务, MyISAM不支持.
- InnoDB支持行级锁, MyISAM支持表级锁.
- InnoDB支持多版本并发控制(MVVC), MyISAM不支持.
- InnoDB支持外键, MyISAM不支持.
- MyISAM支持全文索引, InnoDB**部分版本**不支持(但可以使用Sphinx插件)

5. MySQL怎么恢复半个月前的数据

通过整库备份+binlog进行恢复. 前提是要有定期整库备份且保存了binlog日志.

6. MySQL事务的隔离级别, 分别有什么特点

1. 读未提交(RU): 一个事务还没提交时, 它做的变更就能被别的事务看到.
2. 读提交(RC): 一个事务提交之后, 它做的变更才会被其他事务看到.
3. 可重复读(RR): 一个事务执行过程中看到的数据, 总是跟这个事务在启动时看到的数据是一致的. 当然在可重复读隔离级别下, 未提交变更对其他事务也是不可见的.
4. 串行化(S): 对于同一行记录, 读写都会加锁. 当出现读写锁冲突的时候, 后访问的事务必须等前一个事务执行完成才能继续执行.

7. 做过哪些MySQL索引相关优化

- 尽量使用主键查询: 聚簇索引上存储了全部数据, 相比普通索引查询, 减少了回表的消耗.
- MySQL5.6之后引入了索引下推优化, 通过适当的使用联合索引, 减少回表判断的消耗.
- 若频繁查询某一列数据, 可以考虑利用覆盖索引避免回表.
- 联合索引将高频字段放在最左边.

8. 简要说一下数据库范式

- 第一范式: 属性不可再分.
- 第二范式: 在一范式的基础上, 要求数据库表中的每个实例或行必须可以被惟一地区分. 通常需要为表加上一个列, 以存储各个实例的惟一标识. 这个惟一属性列被称为主关键字或主键.
- 第三范式: 在二范式的基础上, 要求一个数据库表中不包含已在其它表中已包含的非主关键字信息. 所以第三范式具有如下特征： 1). 每一列只有一个值. 2). 每一行都能区分. 3). 每一个表都不包含其他表已经包含的非主关键字信息.

9. 一千万条数据的表, 如何分页查询

数据量过大的情况下, `limit offset` 分页会由于扫描数据太多而越往后查询越慢. 可以配合当前页最后一条ID进行查询, `SELECT * FROM T WHERE id > #{ID} LIMIT #{LIMIT}`. 当然, 这种情况下ID必须是有序的, 这也是有序ID的好处之一.

10. 订单表数据量越来越大导致查询缓慢, 如何处理

分库分表. 由于历史订单使用率并不高, 高频的可能只是近期订单, 因此, 将订单表按照时间进行拆分, 根据数据量的大小考虑按月分表或按年分表. 订单ID最好包含时间(如根据雪花算法生成), 此时既能根据订单ID直接获取到订单记录, 也能按照时间进行查询.