简答题

1. 简述事务和事务的四个性质

事务：构成单一逻辑工作单元的操作集合 (事务是访问并可能更新各种数据项的一个程序执行单元)

一致性：数据库总是从一个一致性的状态转换到另一个一致性的状态，事务前后数据的完整性必须保持一致。

原子性：一个事务必须被视为一个不可分割的最小工作单元，整个事务中的所有操作要么全部提交成功，要么全部失败回滚，对于一个事务来说，不可能只执行其中的一部分操作，这就是事务的原子性

持久性：持久性保证一旦事务成功完成，该事务对数据库所做的所有更新就都是持久的，即使事务执行完成后出现系统故障

隔离性：隔离性是指多个事务并发时，每个事务应该是隔离的，一个事务不应影响其他事务的运行效果。通常来说，一个事务所做的修改操作在提交事务之前，对于其他事务来说是不可见的。

1. 视图的定义和作用

视图是采用Create View语句定义的任何一个SQL查询语句，它是无实际数据的“虚表”。

主要作用：

1. 可以在任何QL语句中像表一样的被使用！
2. 增强查询能力且方便(用户/程序员)使用！
3. 还可以提供数据访问安全控制(隐藏数据)！
4. 作为外模式(1级映射)有利于应用独立性！
5. 模式分解的两个基本标准
6. 无损连接分解：将 R 分解为R1 何R2 是无损分解，如果下面至少一个成立的话，那么分解为无损分解: R1 ∩ R2 → R1 或者R1 ∩ R2 → R2
7. 保持函数依赖。当R分解为X和Y，如果要保持函数依赖，当且仅当F在X和Y上的投影FX和FY的并集的闭包等于F的闭包。
8. 参照完整性的定义和目的
9. 定义：“不引用不存在的实体”。即：不允许在一个关系中引用另一个关系中不存在的元组。
10. 目的：用于确保相关联的表间的数据保持一致
11. 建表操作

create table 操作（

职工编号 char(10),

仪器编号 char(10) not null ,

操作时间 date,

完成项目名称 varchar(20),

操作时长 int,

Primary key(职工编号，仪器编号，操作时间),

Foreign key(职工编号) references 职工，

Foreign key(仪器编号) references 仪器

）；

1. 请解释死锁和活锁的概念和死锁的解决方法。
2. 死锁的定义：两个事务“相互等待”对方释放资源各自才能往下执行
3. 活锁的定义：一个事务永远(长时间)等待某一数据项被其它事务释放后才能进行封锁的现象
4. 解决方法：

预防：

每个事务在开始之前封锁它的所有数据项

对所有的数据项强加一个次序

使用抢占与事务回滚

解除：

选择牺牲者

回滚

饿死

1. 请根据数据库系统的不同故障类型说明恢复策略
2. 事务故障恢复策略

事务故障是经常发生的，其一定发生在事务提交之 前。事务一旦提交，即使要撤销也不可能了。对于 事务故障，一般采取的恢复策略就是撤销该事务，并释放其占有的资源

1. 系统崩溃恢复策略

系统崩溃不像事务故障这么频繁，但发生的可能性 还是很大。对于系统崩溃，一般采取的恢复策略是 重新启动操作系统和DBMS，恢复数据库到一致状态（对未提交的事务进行撤销操作，对已提交的事务进行重做操作）。只有对当数据库恢复到一致状态，才允许用户访问数据库

1. 磁盘故障恢复策略

在正常情况下，磁盘故障是很少发生的。对于 磁盘故障，一般采用的恢复策略是修复系统， 必要时更换磁盘，加载最近备份复本，再根据 日志文件中的日志记录重做最近备份复本以后 提交的所有事务

1. 哈希索引和B+树索引的区别
2. 在查询速度上，如果是等值查询，那么Hash索引明显有绝对优势，因为只需要经过一次 Hash 算法即可找到相应的键值，复杂度为O(1)；当然了，这个前提是键值都是唯一的。如果键值不是唯一(或存在Hash冲突)，就需要先找到该键所在位置，然后再根据链表往后扫描，直到找到相应的数据，这时候复杂度会变成O(n)，降低了Hash索引的查找效率。所以，Hash 索引通常不会用到重复值多的列上，比如列为性别、年龄的情况等（当然B+tree索引也不适合这种离散型低的字段上）；
3. Hash 索引是无序的，如果是范围查询检索，这时候 Hash 索引就无法起到作用，即使原先是有序的键值，经过 Hash 算法后，也会变成不连续的了。因此

①、Hash 索引只支持等值比较查询、无法索成范围查询检索，B+tree索引的叶子节点形成有序链表，便于范围查询。

②、Hash 索引无法做 like ‘xxx%’ 这样的部分模糊查询，因为需要对 完整 key 做 Hash 计算，定位bucket。而 B+tree 索引具有最左前缀匹配，可以进行部分模糊查询。

③、Hash索引中存放的是经过Hash计算之后的Hash值，而且Hash值的大小关系并不一定和Hash运算前的键值完全一样，所以数据库无法利用索引的数据来避免任何排序运算。B+tree 索引的叶子节点形成有序链表，可用于排序。

1. Hash 索引不支持多列联合索引，对于联合索引来说，Hash 索引在计算 Hash 值的时候是将索引键合并后再一起计算 Hash 值，不会针对每个索引单独计算 Hash 值。因此如果用到联合索引的一个或者几个索引时，联合索引无法被利用；
2. 因为存在哈希碰撞问题，在有大量重复键值情况下，哈希索引的效率极低。B+tree 所有查询都要找到叶子节点，性能稳定；
3. 请描述基于日志的恢复机制的恢复过程

因为日志总是比事务先提交，并且日志保存在非易失性存储器上，所以我们可以使用它来恢复系统，扫描日志文件（如果有检查点的话，另外写）

1. 对于在故障前提交或者插销的事务，全部重做
2. 对于在故障前未完成且未提交和未撤销的事务，撤销
3. 什么是索引，为什么要使用索引？
4. 定义：索引是一种特殊的数据库结构，由数据表中的一列或多列组合而成，可以用来快速查询数据表中有某一特定值的记录
5. 作用：
6. 索引小，可在内存中处理，避免逐个读全部记录文件
7. 索引排了序，加快查询速度，提高系统的性能
8. 通过创建唯一性索引，可以保证数据库表中每一行数据的唯一性
9. 在实现数据的参考完整性方面可以加速表与表之间的连接
10. 在使用分组和排序子句进行数据查询时也可以显著减少查询中分组和排序的时间
11. 可以给所有的 MySQL 列类型设置索引
12. 请举例说明密集索引和稀疏索引的概念，以及优势和他们的缺点。
13. 密集索引的定义：索引文件中，每个搜索码都有一个索引项
14. 密集索引的优缺点
    1. 稠密索引相对来说占用的空间大
    2. 速度相对于稀疏索引快很多
    3. 密集索引则不需要表中数据在数据文件中已按照索引列值进行排序
15. 稀疏索引的定义：索引文件中，只为某些搜索码建立索引项，需要表中数据在数据文件中已按照索引列值进行排序，只有主索引才能使用
16. 稀疏索引的优缺点
    1. 降低索引文件空间开销
    2. 能够提高搜索效率(跳跃查找)
    3. 但由于查找数据时要进行数据块内部的检索，所以查询数据所需时间会长一些
17. 请简述数据库中事务与并发控制和故障恢复的联系

事务是恢复和并发控制的基本单位

1. 请简述基于检查点日志的恢复技术及其优势

因为日志总是比事务先提交，并且日志保存在非易失性存储器上，所以我们可以使用它来恢复系统，扫描日志文件，检查点前提交和撤销的事务，什么也不做，对于所有在检查点时执行的事务和之后的所有事务：

如果在故障前提交和撤销，重做

如果没有提交和撤销，撤销

当系统故障发生时，必须检查日志来决定哪些事务需要重做，哪些需要撤销。如果搜索整个日志来确定该信息，那么开销太大。在系统崩溃后，系统检查日志以找到最后一条检查点记录，只需要对L中的事务以及检查点记录写到日志中之后才开始执行的事务进行撤销或重做操作。考虑在检查点前完成的事务Ti。<Ti commit>记录（或<Ti abort>记录）在日志中出现检查点记录之前，Ti所作的任 何数据库修改都必然已在检查点之前或作为检查点本身的 一部分写入数据库。因此，在恢复时就不必再对Ti执行重做操作。因此，引入检查点可以提高恢复过程的效率。

1. 数据库开发的主要过程

数据需求、概念设计、逻辑设计、物理设计、数据库实现、数据维护、数据分析等过程

1. 数据库缓冲区的作用
2. CPU处理信息快捷，但从磁盘读取记录缓慢
3. 缓冲区一次I/O读多硬盘上多个记录(按块),可明显减少磁盘I/O开销(连续读-节省时间)；例如：查询所有学生的记录；
4. 缓冲区中的记录，可能为多个应用所需要，可明显减少磁盘I/O开销(重复读-浪费时间)。例如：大家同时查询奥运会最新100米跑成绩
5. 简述数据库的存储机制
6. 数据库用文件的方式来存储数据。一个数据库被映射到多个不同的文件(file) ，文件由操作系统来管理，这些文件被永久存储在磁盘上
7. 一个文件在逻辑上被组织成记录的一个序列，记录被映射到磁盘块(block)上。文件中存储记录的常用方法有：堆文件组织，顺序文件组织，多表聚集文件组织和散列文件组织等。
8. 每个文件(file)被分成定长的存储单元-块(block)，块是数据存储和传输的基本单位(默认一般是4-8KB)。一个块可以包括很多记录(假设一个记录总比块小；对大数据如图片需单独处理和存储)，且一个记录的数据不能跨块存储
9. 关系数据库管理系统-RDBMS，采用自描述方式！数据存储采用关系模式，元数据存储也采用关系模式！
10. 简述数据库的查询处理
11. Sql语句通过语法分析器与翻译器变成关系代数表达式
12. 代数表达式结合有关数据的统计值通过优化器得到执行计划