关键参考简答题

1. 数据库设计的基本步骤？(重点)

数据需求、概念结构设计、逻辑结构设计、物理结构设计、数据库实施、数据库运行和维护、数据分析等过程。

2. 数据库安全性保护的常用措施？

访问控制、数据加密、审计和监控、强化权限管理、定期备份与恢复、更新和补丁管理、强密码策略、网络安全防护、数据库安全评估、培训与意识提升

3. 数据库运行过程中可能产生的故障有哪几类？具体内容是什么？

硬件故障：包括服务器、存储设备或网络设备等硬件出现故障，如硬盘损坏、电源故障、网络中断等。

软件故障：指数据库管理系统（DBMS）或其他相关软件出现错误或故障，如数据库服务崩溃、操作系统故障、网络协议错误等。

数据库逻辑故障：涉及数据库中的逻辑错误或异常，例如数据损坏、数据丢失、约束冲突、索引错误等。

用户错误或误操作：由用户或管理员操作错误导致的故障，例如误删除数据、误修改表结构、错误的查询语句等。

安全故障：包括数据库遭受恶意攻击、未经授权的访问、数据泄露等安全事件。

4. 什么是视图？有何作用/优点？(重点)

定义：视图是采用 Create View 语句定义的任何一个 SQL 查询语句，它是基于一个或多个表的查询结果所创建的虚拟表。

主要作用：

1）数据简化和聚合：可以在任何 QL 语句中像表一样的被使用！

2）简化复杂查询：增强查询能力且方便(用户/程序员)使用！

3）数据安全性控制：还可以提供数据访问安全控制(隐藏数据)！

4）数据逻辑独立性：作为外模式(1 级映射)有利于应用独立性！

5. 事务的定义和 ACID 特性

事务是数据库管理系统中的一个操作单位，它由一组数据库操作组成，形成一个逻辑上的工作单元。

1. **原子性：**事务所有操作要么全部提交到数据库，全部执行成功，要么都不提交，全部失败回滚。
2. **一致性：**事务执行前后，数据库从一个一致的状态转换到另一个一致的状态。事务在执

行过程中对数据库所作的修改必须符合预定义的规则和约束，以保证数据的完整性和正确性。
3. **隔离性**：事务的执行是相互隔离的，即每个事务在执行过程中的中间结果对其他事务是不可见的。并发执行的事务，避免了数据争用和不一致的问题。
4. **持久性**：一旦事务提交，其所做的修改将永久保存在数据库中，即使在系统故障或崩溃的情况下也不会丢失，以确保数据的可靠性和持久性。

6. 什么是数据库日志？有什么作用？(重点)

数据库日志是数据库管理系统（DBMS）用于记录数据库操作的详细信息的一种机制。它用于捕获对数据库的更改，包括插入、更新和删除操作，以及其他管理操作，如创建表、索引或备份等。

恢复和容错：在系统崩溃或发生故障时，可以使用日志来还原数据库到崩溃点之前的状态。通过重新执行日志中的操作，可以将数据库恢复到最后一次完整的状态，从而确保数据的一致性和完整性。

事务支持：数据库日志用于记录事务的开始、提交或回滚等操作，以便在需要时可以正确执行或撤消事务。

性能优化：通过将一系列对数据库的操作批量记录到日志中，而不是直接写入磁盘，可以减少磁盘写入的次数，从而提高数据库的写入性能。

审计和安全：通过分析数据库日志，可以追踪和监视对数据库的操作，以确保合规性和数据安全。

7. 什么是索引？有何优点与缺点？

索引是数据库中用于加快数据检索速度的数据结构。它是根据一个或多个列的值创建的，并与实际数据存储在一起，以提供快速的数据访问路径。

优点：

1. 提高检索速度：索引减少了全表扫描的需要，可以显著提高数据的检索速度。
2. 加快排序和聚合操作：对于排序和聚合操作（如 GROUP BY），索引按照特定的顺序存储数据，使得排序和分组操作更加高效。
3. 约束和唯一性：索引可以用于实施表的唯一性约束和外键约束，防止重复或无效的数据插入。
4. 支持快速的数据访问：通过使用索引，可以直接跳转到满足查询条件的数据位置，从而减少了磁盘 IO 操作，提高了数据的访问速度。

缺点：

1. 占用存储空间：索引需要占用额外的存储空间来存储索引结构。
2. 增加数据插入、更新和删除的开销：当对表进行插入、更新或删除操作时，索引需要进行相应的更新以保持数据的一致性。
3. 索引维护代价高：索引的创建和维护需要一定的时间和计算资源。
4. 索引选择和优化的复杂性：在设计索引时需要考虑查询的模式和性能需求。选择不合适的索引或过多的索引可能导致查询性能下降。

8. 采用数据库三层模式结构的优点有哪些？

1. 数据独立性：外模式与模式：逻辑独立性；模式与内模式：物理独立性

2. 数据安全性：外模式定义了用户对数据的视图和操作权限，模式定义了整体数据库的安全策略和约束条件，从而确保数据的安全性和完整性。

3. 数据共享和集中管理：不同的应用程序可以通过不同的外部模式访问数据库，这样可以避免数据的冗余存储和一致性问题，提高数据的可维护性和数据的一致性。

4. 数据抽象和逻辑简化：模式提供了对数据库的抽象描述，隐藏了底层数据的物理存储细节。这样可以简化应用程序的开发过程，提高开发效率。

9. 在数据库中为什么要使用并发控制？

并发操作是指多个用户或应用程序同时对数据库进行读取和写入操作。如果没有适当的并发控制机制，可能会导致以下问题：

丢失更新：当多个事务同时读取和修改相同的数据时，最后提交的事务可能会覆盖先前提交的事务所做的修改，从而导致数据的丢失。

读脏数据：一个事务在另一个事务未提交的情况下读取到了未提交的数据，而后续操作可能导致这些数据被回滚，从而读取到了不一致或无效的数据。

不可重复读：一个事务在多次读取同一数据时，由于其他事务的并发修改，每次读取到的数据可能不一致，导致数据的不可预测性。

10. 模式分解的两个基本标准极其含义是什么？(重点)

基本标准：无损连接分解和保持函数依赖。

1. 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: $R1 \cap R2 \rightarrow R1$ 或者 $R1 \cap R2 \rightarrow R2$

2. 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 F_X 和 F_Y 的并集的闭包等于 F 的闭包。

11. 什么是参照完整性约束？目的是什么？

“参照完整性约束是指：“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。(要求在一个关系中的外键值必须存在于另一个关系中的主键中).

它的目的是：用于确保相关联的表间的数据保持一致。(数据完整性与一致性)

12. 请解释死锁和活锁的概念和死锁的解决方法。

(1) 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

(2) 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

(3) 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

13. 请根据数据库系统的不同故障类型说明恢复策略

(1) 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

(2) 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

(3) 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

14. 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

其他参考简答题

1. 试述 E-R 概念模型转化为关系模型的转换规则

实体类型转换为关系表：将每个实体类型转换为一个关系表。每个实体类型的属性对应于关系表的列，而每个实体实例对应于关系表的行。表的主键通常选取实体类型的主键属性。

弱实体类型转换为关系表：对于弱实体类型，需要为其创建一个关系表。该关系表包含弱实体的所有属性以及一个指向其强实体的外键。表的主键通常由弱实体的标识属性和强实体的主键组成。

1:1 关系转换为关系表：对于 1:1 关系，可以选择将其转化为一个关系表。这个关系表包含两个相关实体类型的属性，每个属性对应于一个实体类型，并包含一个指向另一个关系表的外键。

1:N 关系转换为关系表：对于 1:N 关系，需要在"1"端的实体类型所对应的关系表中添加一个指向"多"端关系表的外键。

N:M 关系转换为关系表：对于 N:M 关系，需要创建一个中间关系表来表示这种关系。中间关系表包含两个实体类型的外键作为其主键。

多值属性转换为关系表：对于多值属性，可以将其转换为一个单独的关系表。这个关系

表包含多值属性的属性和一个指向主体实体的外键。

继承关系转换为关系表：对于继承关系，可以采用以下两种方式转换为关系表：

- 每个具体实体类型对应一个关系表，包含该实体类型的属性。
- 所有实体类型对应一个关系表，包含所有实体类型的属性以及一个表示实体类型的标识属性。

2. 试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

(1) 密集索引的定义：索引文件中，每个搜索码都有一个索引项
(2) 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
(3) 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
(4) 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

3. 试数据库缓冲区的作用

(1) CPU 处理信息快捷，但从磁盘读取记录缓慢
(2) 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；
(3) 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