**以下大部分都是瞎写的！！**

**1. 现代数据库系统是怎样管理数据库模式的？**

数据模式通常分为三级：概念模式、外模式、内模式。

它们都存于数据目录中,是数据目录的最基本内容。DBMS通过数据目录管理和访问数据模式。

**数据模型对系统性能有什么影响？**

数据模型的要素有数据操作、数据结构以及完整性约束。分为概念、逻辑、物理数据模型。

对系统性能的影响主要体现在：查询效率，I/o次数、表达能力，数据冗余，易于使用，易于理解等方面

1、不同的逻辑数据模型在语义表达能力上有很大差异，有的模型可能导致数据冗余；

2、数据模型的抽象层次决定了用户使用的复杂程度如：编程的复杂度

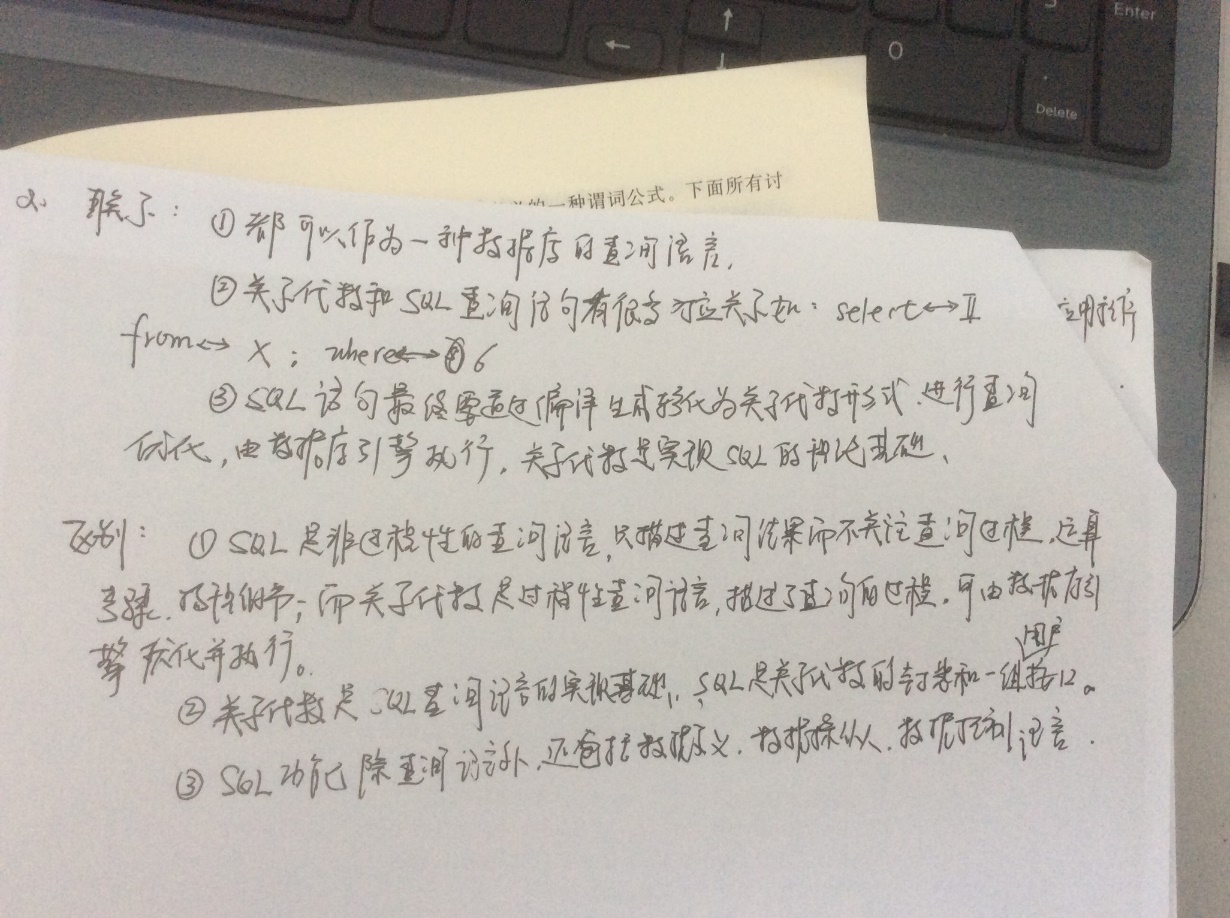
3、当逻辑数据模型映射到物理数据模型后，决定了数据在存储介质上的存储方式。对于数据库系统的查询执行效率有直接的影响。如：记录是否等长，采用直接、堆、或者索引存储，在哪些键值上建立何种类型额索引等。

**结构化数据：**即行数据,存储在数据库里,可以用二维表结构来逻辑表达实现的数据。

**半结构化数据：**介于结构化数据和非结构的数据之间的数据，HTML文档就属于半结构化数据。它一般是自描述的，数据的结构和内容混在一起，没有明显的区分。

**非结构化数据：**相对于结构化数据而言,不方便用数据库二维逻辑表来表现的数据即称为非结构化数据,包括所有格式的办公文档、文本、图片、XML、HTML、各类报表、图像和音频/视频信息等等。

**2. SQL与关系代数的联系与区别：**



**3.** (1) 房间：主键-房间号 客人：主键-身份证号 住宿：主键-（身份证号，入住日期），外键-房间号，身份证号

(2) ①SELECT 姓名

FROM 客人,住宿

WHERE 客人.身份证号 = 住宿.身份证号

AND 籍贯=‘南京‘

AND 房间号=301;

②SELECT 姓名

FROM 客人

WHERE NOT EXISTS

(SELECT \* FROM房间

WHERE NOT EXISTS

(SELECT \* FROM 住宿

WHERE住宿.身份证号 = 客人.身份证号

AND 住宿.房间号=房间.房间号);

③SELECT 姓名, COUNT( \* )

FROM 客人, 住宿

WHERE客人.身份证号=住宿.身份证号

AND YEAR(入住日期) = 2014

GROUP BY客人.身份证号

HAVING COUNT(\*) >= ALL

(SELECT COUNT(\*) FROM客人,住宿

WHERE客人.身份证号=住宿.身份证号

AND YEAR(入住日期)=2014

GROUP BY客人.身份证号);

4. 查询条件是合取式。由于没有多属性索引，不能用多键查找，没有多个二次索引，不能用预查找。只能用条件筛选法：

**先按房间号：**按房间号，二次索引，元组数=10000/120=84,由于每个物理块有40个tid,则查询tid的I/O代价为3+[84/40]=6

总的I/O代价为84+6=90

**先按日期范围：**I/O代价为3+[10000/2/5]=1003

综上所述，最好的选取策略是用房间号=301这个条件选出元组，再用日期范围这个条件进行筛选，执行代价是90。

5. 介质失效恢复时，对运行记录中上一检查点以前已提交事务应该redo否？为什么？

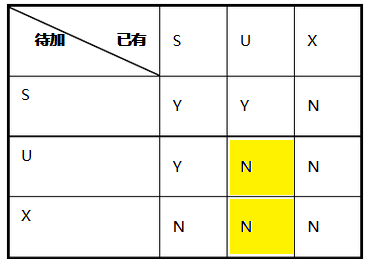
需要redo，

介质失效是指数据库存储介质：磁盘发生故障。磁盘中的数据全部丢失。

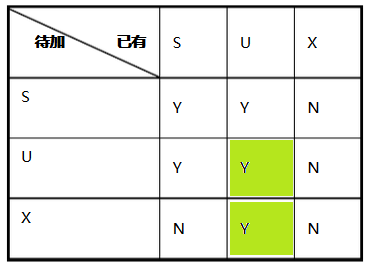
而检查点是为了降低系统失效时的恢复代价，对介质失效没有益处。检查点是将当前提交却没有写如永久介质磁盘的事务相关物理块写入磁盘，当系统失效恢复时，检查点之前的事务不需要重做。

因此，介质失效后只能在新介质中加载最近的后备副本，然后根据日志log文件将提交的事务redo，恢复到最近一致状态。在后备副本之后的检查点根本不会再后备副本中体现出来。

6. 将SUX锁的矩阵从



改为



则如下过程能否正常运行？（18分）

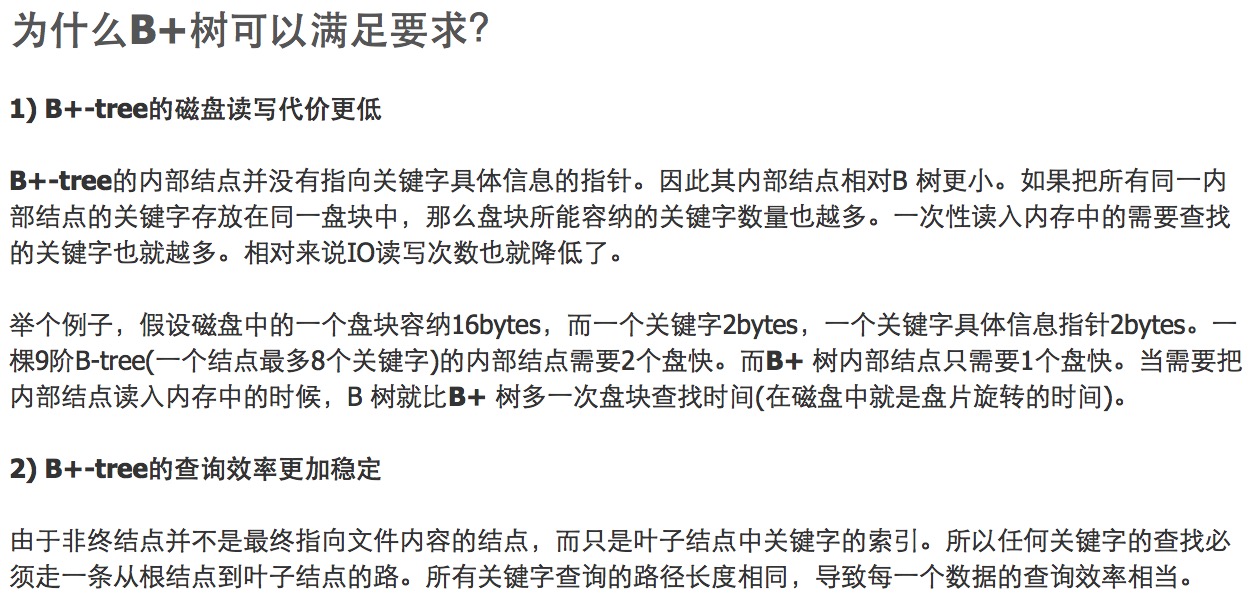
 (1)、都是读操作  可以

(2)、都是写操作  可以；

如果只有写操作的话，就只加X锁，题目中对于X锁的协议没有改变吧

(3)、一部分读操作，一部分写操作可以

只加X，S锁，题目中对于X,S锁的协议没有改变吧



需要redo，  
介质失效是指数据库存储介质：磁盘发生故障。磁盘中的数据全部丢失。  
而检查点是为了降低系统失效时的恢复代价，对介质失效没有益处。检查点是将当前提交却没有写如永久介质磁盘的事务相关物理块写入磁盘，当系统失效恢复时，检查点之前的事务不需要重做。  
因此，介质失效后只能在新介质中加载最近的后备副本，然后根据日志log文件将提交的事务redo，恢复到最近一致状态。在后备副本之后的检查点根本不会再后备副本中体现出来。

**数据库系统中的多级数据模式对数据独立性的影响？**

◆ 概念模式:是数据库中全部数据的整体逻辑结构的描述。它由若干个概念记录类型组成。概念模式不仅要描述概念记录类型，还要描述记录间的联系、操作、数据的完整性、安全性等要求。

　　◆ 外模式:是用户与数据库系统的接口，是用户用到的那部分数据的描述。

　　◆ 内模式:是数据库在物理存储方面的描述，定义所有的内部记录类型、索引和文件的组成方式，以及数据控制方面的细节。

　　◆ 模式/内模式映象:这个映象存在于概念级和内部级之间，用于定义概念模式和内模式间的对应性，即概念记录和内部记录间的对应性。此映象一般在内模式中描述。

　　◆ 外模式/模式映象:这人映象存在于外部级和概念级之间，用于定义外模式和概念模式间的对应性，即外部记录和内部记录间的对应性。此映象都是在外模式中描述。

　　◆ 数据独立性:在数据库技术中，数据独立性是指应用程序和数据之间相互独立，不受影响。数据独立性分成物理数据独立性和逻辑数据独立性两级。

　　◆ 物理数据独立性:如果数据库的内模式要进行修改，即数据库的存储设备和存储方法有所变化，那么模式/内模式映象也要进行相应的修改，使概念模式尽可能保持不变。也就是对模式的修改尽量不影响概念模式。

◆ 逻辑数据独立性:如果数据库的概念模式要进行修改(如增加记录类型或增加数据项)，那么外模式/模式映象也要进行相应的修改，使外模式尽可能保持不变。也就是对概念模式的修改尽量不影响外模式和应用程序。

正如在在三级数据模式中的介绍中所述，概念模式描述数据间的逻辑，外模式是用户看到的自己有权访问的数据部分的逻辑和数据，内模式是所有数据的逻辑和包括物理上的存储，多级数据模式满足了不同用户和管理员对数据的需求，简化了访问和管理，避免了数据冗余和修改时可能的不兼容，使得数据的存储与使用独立，管理员与用户间数据模式独立，修改和使用独立。