第二章上

* 数据库系统的功能：CRUD 增删改查
  1. Create：创建数据
  2. Read：读取数据
  3. Update：修改数据
  4. Delete：删除数据
* 数据模型：存储数据的结构（关系、文档等）
  1. database数据库
  2. collection文档集
  3. document文档  
     键值对；可嵌套
* 文档数据库的基本功能（MongoDB为例）  
  “\_Id”: objectId(“…”)
  1. 增/Create: db.myCollection.insertOne({…})
  2. 查/Read: db.myCollection.find({…})
  3. 改/Update: db.myCollection.updateOne({…},{$set:{…}})
  4. 删/Delete: db.myCollection.deleteOne({…})
* 文档数据库——数据存储
  1. CPU——内存——硬盘
  2. 文档存储。硬盘空间划分为页，数据存储以页为单位
  3. 文档集存储。文件系统：使用Inode（基本信息、直接指针、间接指针）
  4. 索引。利于文档的查找

**第*1*题：一个文档数据库里面有两类对象，书（book）和人（person）。书和人之间存在一种写作的关系，即某个人是某本书的作者。请问这种写作关系的信息应该如何存放？**

A：在描述书的文档中增加一个“作者（author）”属性，取值为其作者的名字。

B：在描述人的文档中增加一个“著作（writing）”属性，取值为其著作的标题。

C：在描述书的文档中增加一个“作者（author）”属性，取值为其作者的ID。√

D：A和B均可。

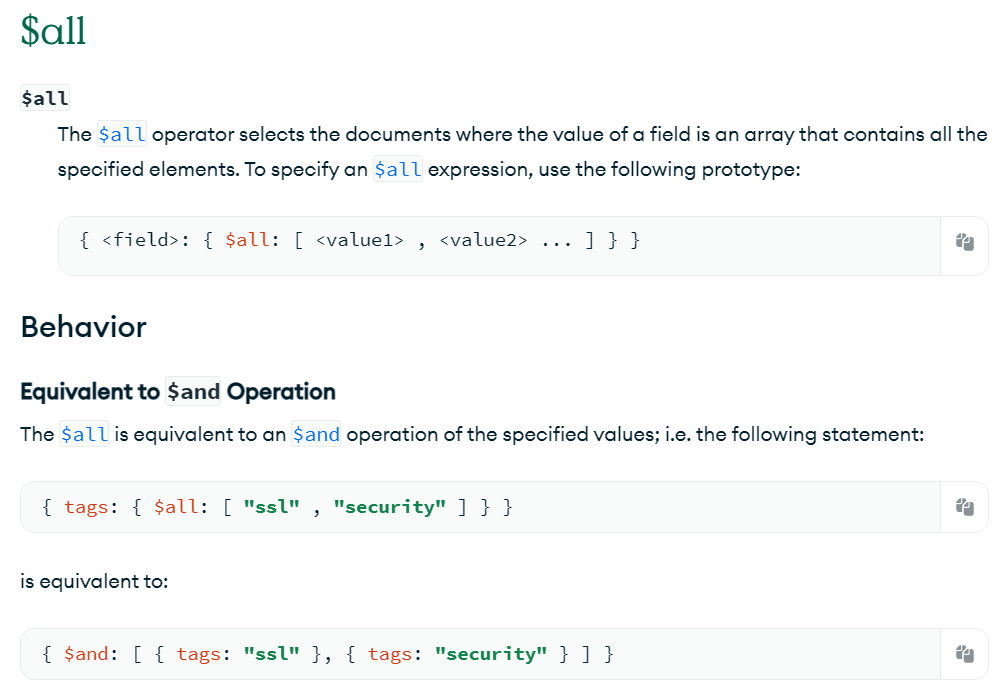
**第*2*题：文档数据库允许一个属性有多个取值，比如{...colors: ["red", "blank"] ...} 或者 {...hobbies: ["red", "blank"] ...}。那么，哪些文档满足以下查询？**

A：仅{... tags: ["computer", "music"]... } √

B：{... tags: ["computer"]... }和{... tags: ["music"]... }

C：{... tags: ["computer", "music"]... }和{... tags: ["computer", "music", "movie"]... }

D：{... tags: ["computer"]... }和{... tags: ["music"]... }和{... tags: ["computer", "music"]... }和{... tags: ["computer", "music", "movie"]... }



**第*3*题：数据库系统的增、删、改、查操作中的改操作（update）通常可以由一个删操作（delete）和一个增操作（create）实现。那么为什么我们不把“增、删、改、查”（CRUD）简化为“增、删、查”（CRD）？以下哪个理由不成立？**

A：先删后增虽然可以代替改，但其性能可能比改差。

B：先删后增的程序写起来比较复杂，没有一个单独的改操作简洁。

C：先删后增是两个独立的操作。如果中间出现状况（比如掉电或者bug），会出现只删未增的情况，导致数据正确性问题。

D：这只是一种习俗，二者并没有什么本质区别。√

**第*4*题：大部分系统都是对存储空间进行分页管理的。请问，分页模式的优势不包括？**

A：有利于减少存储空间的碎片化，提升空间利用率。

B：有利于提升数据访问的性能。

C：有利于提升内存缓存的效率。√

D：有利于减少空间管理的成本（即减少空间管理对CPU和内存资源的消耗）。

**第*5*题：课程中提到，内存由于比较昂贵且无法持久地保存数据，通常只作为数据缓存。那么，什么数据不适合被放在缓存中？**

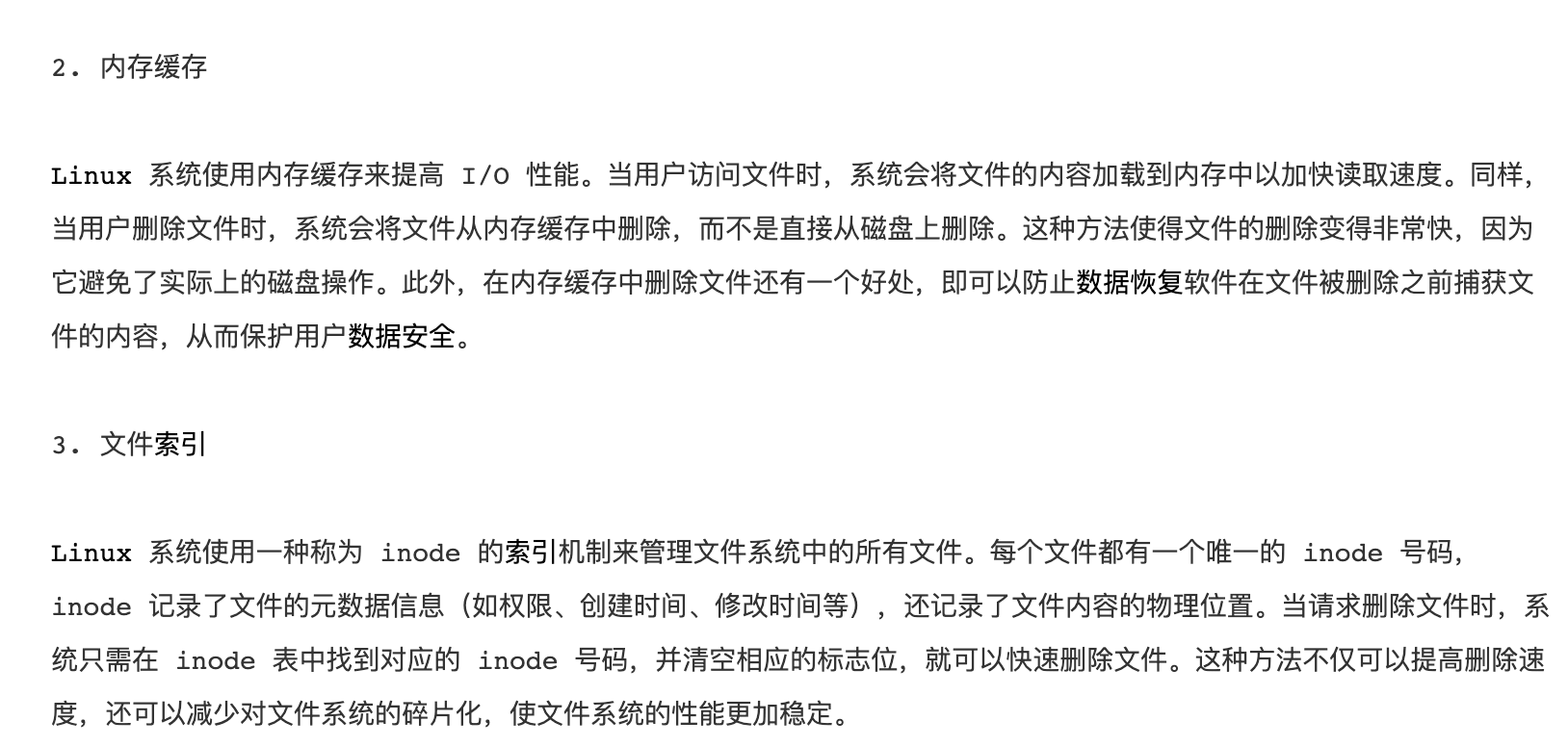
A：经常被修改的数据

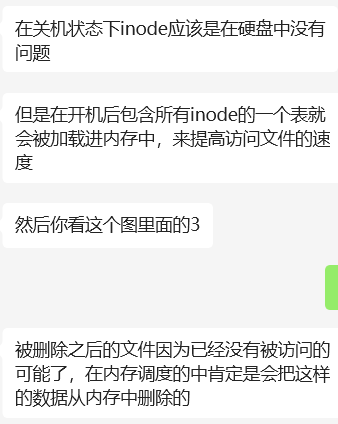
B：像Inode这样的组织结构数据

对于像Inode这样的组织结构数据，通常包含文件系统相关信息，例如文件的权限、大小、创建时间等。由于这些数据较为稳定且很少发生改变，不适合频繁地放在缓存中。相反，它们更适合存储在持久性存储介质（如硬盘）中，以确保数据的可靠性和一致性。

C：刚被插入的数据

D：刚被删除的数据 √





**思考题1：**当我们对存储空间进行分页管理的时候，页的大小通常是一个设计要点。有的数据管理系统选择使用比较小的页，如2KB或4KB。而另一些系统会使用比较大的页，比如4MB或8MB。请问：小页面对什么情况有利？大页又对什么情况有利？我们确定页的大小时应该考虑哪些因素？

小页：适合较小的文档数据的存储，可以减小数据库大小，提高空间的利用率，降低一次需要读取的数据量，减小IO压力。但是页表大小太大。

大页：适合较大的文档数据的存储，减少一次性需要访问的页数，提高IO效率，提高操作的效率，降低管理成本。但是会分配过多的空间，造成浪费。

考虑因素：

1. 文档大小。根据存储的数据文档大小选择合适的页大小。
2. 操作系统页大小。将数据管理系统页大小设置为操作系统页大小的倍数，提高IO效率。

思考题2：如课程中提到的，内存通常被数据管理系统作为缓存使用。缓存的数据单元可以有不同的选择；可以是页，即当访问完一页后，将整个页继续保留在内存中，以期后面再次访问该页就无需再从硬盘获取；也可以是文档，即当访问完一页中的某个文档后，将这个文档继续保留在内存中，而将页移除，以期后面再次访问该文档时无需再从硬盘获取。请问：页缓存和文档缓存各自的优势和劣势是什么？什么情况下，我们可以考虑使用文档缓存？

1. 页缓存减少对缓存的处理，速度更快。文档缓存需要在访问完页后再将文档留在内存中，将页移除，这些操作降低了访问的效率。
2. 页缓存对局部性友好，当访问其他的文档和原来的文档位置很接近时，很可能也在该页缓存中，可以直接读取不用从硬盘获取。而文档缓存时，访问其他没有被缓存的文档就需要从硬盘获取。
3. 文档缓存节约内存，只需要把访问的文档放在内存中。而页缓存需要缓存整个页，内存使用量大，缓存效率低。

当访问的文件数量较多，而且内存容量较小时，可以考虑使用文档缓存。