1. **结构上数据缓冲池是由若干个数据缓冲区hash链表所构成，每个hash链表中的缓冲区具有相同的hash值。当hash值的取值范围增大时，hash链的个数增加，每个hash链的平均长度减少，在链中的查询时间就会降低。请问hash值的取值范围是否越大越好？为什么？**

答：hash值的取值范围不是越大越好，原因如下：

1. 因为hash值取值范围越大，数据缓冲池中的数据缓冲区hash链表也就越多，由于每个hash链都有一个表头标，所以当hash链较多时，系统为表头标分配的内存空间也就越多，每个hash链上的缓冲区也就越少(缓冲区数量固定的情况下)，有可能每个hash链上就只跟了1、2个缓冲区，却也要为这一hash链分配一个表头标，整个hash链所占用的将近一半的内存被用来分配个表头标，从内存空间的利用效率上来说是非常低的；
2. 同时，hash链越多，给管理数据缓冲池也带来了更多的麻烦，散列函数把缓冲区均匀地分布在一组hash链上；根据该数据块所对应的设备号和块号数据对取余计算其 hashno(散列、杂凑)值，根据其 hashno 的值放入到相应 hash 链表的链头。当hash链越多，每个hash链上缓冲区的数量的均匀性无法保证。
3. **在一个较大的物理磁盘空间上建立文件系统时，一种方法是建立一个单一的采用两级存储结构的文件系统，另一种方法是建立多个采用一级存储结构的文件系统。分析这两种方法各自的优缺点。**

答：（1）单一的采用两级存储结构的文件系统：

文件系统由两级组成，第一级由超级块和若干个柱面组块所组成（如果是根文件系统则还包括引导块）；第二级（即柱面组块）又是由超级块拷贝块、柱面组信息块、i节点表块和数据区所组成。这种存储结构是二维的。

**优点：**1）快速定位数据块，数据读写效率高。在两级存储结构的文件系统中，通过柱面组号、柱面号i节点表等信息能快速查找所需数据块；2）文件系统安全性高。在每个柱面组块中存放有一个超级块拷贝块，使系统在超级块被意外破坏时，能从任何一个柱面组中进行恢复而不致使整个文件系统陷入瘫痪。此外，每个柱面组中的超级块拷贝块的存放位置为安全起见不一定都装在柱面组中的最前面，而是可浮动地装在该柱面组中的任何位置。3）能够快速定位系统中的空闲资源。系统只需要维护一张较小的表，即位示图，就可以快速地检测指定资源的忙闲状态，或快速查找可用的空闲资源。

**缺点：**1）两级存储结构实现复杂，需要将物理磁盘空间划分为若干柱面组块，每个柱面组块又被划分为多个柱面块。2）在每个柱面组块中存放有一个超级块拷贝块，且这些超级块拷贝块浮动的装在柱面组中的任何位置，耗费大量存储资源。3）若文件系统被破坏，那么系统将不能使用。

1. 多个采用一级存储结构的文件系统

一级存储结构的文件系统由超级块、索引节点表块和数 据区组成，（如果是根文件系统，就还包括引导块）。整个文件系统存储结构是一维的。

**优点：**1）结构简单，易于实现2）单个文件系统被破坏时，不会影响其它文件系统的正常使用。 **缺点：**1）单个文件系统安全性不高。对于单个文件系统而言，由于是一级存储结构，没有超级块拷贝块，因此，当文件系统的超级块被破坏时，整个文件系统将无法恢复。

1. **在较大的系统（如大型机、巨型机）中为了提高数据I/O的速度，没有采用单一总线串行进行I/O的结构，而是采用多通道并行进行I/O的结构。请问要在这类系统中建立UNIX文件系统，则文件系统（特别是数据I/O部分）可能（或应该）要做哪些改变？**

答：

单一总线串行I/O在同一时刻只能传输一位数据，而多通道的并行I/O可以在同一时刻传输多位数据；从单一总线的串行I/O到多通道的并行I/O，UNIX文件系统需要改变数据缓冲区池中的缓冲区大小、结构和读入调用方法，每次在同一时刻读取磁盘中的多位数据，将之写入缓冲区中，使得读入前的数据位与读入后的数据位相一致。并且如果能够将缓冲区中数据块的大小按通道数的整数倍来组织，使得每次缓冲区和磁盘间的一次I/O都刚好是一个完整的操作，相应的，在磁盘上也将数据按这种方式组织，将数据块的大小定为一次I/O数据量的整数倍，从而大大提高了数据I/O的效率和速度。

1. **inode中包含文件的数据块索引表，用以标识文件占用的数据块的位置。如果用数据块链表（即每个数据块都有一个指向下一个数据块的指针所构成的链表）来代替数据块索引表，请问其优缺点分别是什么？**

答：

数据块索引表，可以只通过一次就可以获得要读取数据的文件系统块号，最坏情况下需要四次读写硬盘，才能获取数据。数据块索引表用于检索本文件占用的数据块。它包含12项 直接索引表目和3项间接索引表目。根据要读写的数据在文件中 的位置可计算出该数据所在的逻辑块号，查索引表就可找到逻 辑块所在的文件系统块号。 系统根据计算出来的逻辑块号判断是否包含在直接索引表 中，如果是，则取出直接索引表中的文件系统块号；如不是， 则看是否包含在一次间接索引块中，否则再寻找二次和三次间 接索引块。

而采用数据块链表的方式，将标记为inode的文件的所有占用数据块都用指针指向构成链表，来标识文件占用数据块的位置；当文件占用的数据块较少时，数据块索引表优势明显；采用数据块索引表的方式可以直接在前面的12项直接索引表目中根据计算该数据所在的逻辑块号，查索引表就可找到逻辑块所在的文件系统块号，查找速度更快，而使用数据块链表的方式需要从第一个数据块往后遍历查找，查找速度较慢。

但当文件较大，文件占用的数据块较多时，使用数据块链表有时更大；数据块索引表中必定存在一次间接、二次间接甚至三次间接块号表，在数据块定位上面，需要逐层查找，一层层地定位磁盘块，再从磁盘块中取出块号，然后定位，这无疑会降低查找的速度；而使用数据块链表则只需要顺序查找，速度更快；而且随着文件的增大，一级、二级间接满足不了要求后，需要链接到三级间接块号表上或更高级间接块号表上，这使得查找和追加数据块都变得更加复杂，效率也就更低，而数据块链表一直是顺序查找，而且追加数据块也只需要直接在链表表头追加，效率也就更高。

1. **如果在UNIX系统中增加一类新的进程——实时进程，你认为这类实时进程应该具有什么样的运行模式？**

答：（1）实时I/O调度

对于同步I/O，进程需要等待当前这次I/O执行结果方可继续运行。而在异步I/O时，可以实现应用作业和I/O任务并行执行，提供更高的效率和性能。进程发出I/O请求后，可以立即执行后面的指令而无需等待本次I/O的结束。I/O结束后，系统通过信号灯的机制通知该进程。

（2）共享内存和内存上锁

共享内存和内存映射机制可以使实时进程之间的相互通信没有系统调用的开销，从而加快速度；进程可以在初始化时通过映射制定共享内存块，这样在进程运行过程中，不同进程之间可以通过共享内存交换信息而不需要通过系统的调用。

（3）固定的优先级调度策略

采用固定的优先级调度策略，可以保证实时功能不会因为后台作业而延迟，具体可采用时间共享、先进先出和循环调度等策略。

（4）增加时钟和定时器类型

为了支持实时应用，应增加一次性定时器和阶段定时器，使定时器的时间可以是绝对时间，也**可以**是相对时间。

1. **当某事件发生时，核心将唤醒所有睡眠在该事件上的进程，并使他们进入到就绪状态下排队等待，但最终只有其中最先进入运行态的那个进程获得相应的资源，而其他进程虽然陆续进入运行态，但因所需资源已被占用而又重新进入睡眠态。下一次被唤醒时又可能重复上面的状况，从而出现极端的情况：某个进程一直这样循环，始终不能获得所需的资源。请问这种情况会发生吗，为什么？**

答：不会，每个进程都有一个优先权字段，用于进程的调度；核心在三种情况下计算进程的优先权：

（1）核心给一个即将进入睡眠态的进程赋予一个特定的优先权。

（2）核心调整从核心态返回到用户态的进程的优先权。

（3）时钟中断处理程序每隔一个“时钟滴答”调整一次所有用户态进程的优先权。

在就绪状态下排队等待某一资源时，当这一资源被释放，而核心态下的进程返回用户态或被抢先，核心调度就绪态下优先级最高的进程进入核心态下运行，核心在一定的时候重新计算进程的优先权，由于优先权值是关于它最近使用CPU时间的函数，所以，随着时间的推移，在就绪态下的时间越长，优先权值越小，优先级也就越高，同时，对于具有相同优先级的进程，会首先调度最先进入就绪态下的进程获得资源，所以，不会出现一个进程一直在睡眠态和就绪态下切换而始终不能获得所需资源的情况。

1. **用三个进程和三个管道建立一个数据通讯环，请写出其基本的建立流程。**

假设在系统中共有3个进程prc1、prc2、prc3，它们之间的进行通讯的3个管道为pip12、pip23、pip31；要使用这3个进程和3个管道建立一个数据通讯环，过程如下:

关闭进程prc1的标准输出文件(文件描述符1)，并复制(系统调用dup)从管道pip12返回的写文件描述符，因为在进程prc1文件描述符表中的第一个空槽是刚刚由关闭操作腾出来的，所以内核将管道写文件描述符拷贝到文件描述符表中的第1项中，这样，标准描述输出文件描述符变成了管道pip12的写文件描述符；进程prc1以类似的操作将标准输入文件描述符替换为管道pip31的读文件描述符。与此类似，进程prc2关闭它的标准输入文件(文件描述符0)，然后复制(系统调用dup)管道pip12的读文件描述符；由于文件描述符表的第一个空槽是原先的标准输出槽，所以prc2的标准输入变成了管道的pip12的读文件描述符，进程prc2做一组类似的操作使它的标准输出变为管道pip23的写文件描述符。同样的，进程prc3关闭它的标准输入文件(文件描述符0)，然后复制(系统调用dup)管道pip23的读文件描述符；由于文件描述符表的第一个空槽是原先的标准输出槽，所以prc3的标准输入变成了管道的pip23的读文件描述符，进程prc3做一组类似的操作使它的标准输出变为管道pip31的写文件描述符。最后3个进程都关闭从管道调用返回的文件描述符。上述操作的结果是，当进程prc1向标准输出写时，实际上是写向pip12——向进程prc2发送数据，而进程prc2则从它的标准输入读管道。当进程prc2向它的标准输出写时，它实际上是写入管道pip23——向进程prc3发送数据，而进程prc3则从他的标准输入接收来自管道pip23的数据，当进程prc3向它的标准输出写时，它实际上是写入管道pip31—向进程prc1发送数据。至此，3个进程通过3个管道组成了一个数据通讯环，实现了两两之间的通信。

**8.设置数据缓冲区高速缓冲的根本目的，主要就是为了平衡高速运行的内存和低速运行的外存之间的速度差。如果把外部机械磁盘更换成最新的快速固态硬盘（SSD），数据缓冲区高速缓冲还有存在的必要性了吗？为什么？**

答：

数据缓冲区高速缓冲还有存在的必要，这是因为数据缓冲区将数据块暂存在内存当中可以减少磁盘IO，虽然SSD的读写速度很快，但还是跟内存的读写速度有不小的差距，可见数据缓冲区同样可以提高数据的总体读写效率。还有数据缓冲区高速缓冲提供了对磁盘块统一的存取方法，消除了用户对用户缓冲区中数据的特殊对齐需要。另外如果没有了数据缓冲区高速缓冲，还会使固态硬盘的读写更频繁，会大大降低固态硬盘的使用寿命。

**9.对于一个已被打开的文件来讲，如何判断当前是否还有进程在使用该文件？**

答：

活动节点i节点表中的每一个表项对应一个当前正被使用的文件状态信息，所以对于一个已被打开的文件，在活动i节点表中必定有指向它的一项。又由于多个进程要共享一个被打开文件的inode，而每个进程的读写指针都不同，故不能放在inode表中。于是只能在用户打开文件表和活动i节点表之外再建一个系统打开文件表来存放读写指针。因此要判断当前是否还有进程在使用该文件，只需要查看系统打开文件表即file表中有无其他项指向此i节点表中的该项，该项即对应被打开文件的状态信息，若file表有其他项指向，则有其他进程在使用该文件，若没有，则没有其他进程在使用该文件。如果无法访问file表，则需要通过文件描述符来判断，使用该文件的进程越多，文件描述符的值越大。

**10.每一种外部设备通常都有对应的特殊驱动程序，可否设计出一种通用的设备驱动程序，以适应各种外部设备？**

答：

不能设计出该通用的设备驱动程序。这是因为核心与驱动程序的接口是由块设备开关表和字符设备开关表来描述的，而每一种设备类型在开关表中都有若干表项，这些表项在系统调用时引导核心转向适当的驱动程序接口。硬件与驱动程序的接口，是由与机器相关的控制寄存器或操作设备的I/O指令，以及中断向量组成，当一个设备发出中断时，系统识别出中断的设备，并调用适当的中断处理程序，如果是通用的设备驱动程序，将无法识别。