CHAPTER 12

Physical Storage Systems

Solutions for the Practice Exercises of Chapter 12

Practice Exercises

12.1

Answer:
In the first case, SSD as storage layer is better since performance is guaranteed. With SSD as cache, some requests may have to read from magnetic disk, causing delays.

In the second case, since we don't know exactly which blocks are frequently accessed at a higher level, it is not possible to assign part of the relation to SSD. Since the relation is very large, it is not possible to assign all of the relation to SSD. The SSD as cache option will work better in this case.

12.2

Answer:
The disk's data-transfer rate will be greater on the outer tracks than the inner tracks. This is because the disk spins at a constant rate, so more sectors pass underneath the drive head in a given amount of time when the arm is positioned on an outer track than when on an inner track. Even more importantly, by using only outer tracks, the disk arm movement is minimized, reducing the disk access latency. This aspect is important for transaction-processing systems, where latency affects the transaction-processing rate.

12.3

Answer:

a. It is stored as an array containing physical page numbers, indexed by logical page numbers. This representation gives an overhead equal to the size of the page address for each page.

b. It takes 32 bits for every page or every 4096 bytes of storage. Hence, it takes 64 megabytes for the 64 gigabytes of flash storage.

c. If the mapping is such that every p consecutive logical page numbers are mapped to p consecutive physical pages, we can store the mapping of the first page for every p pages. This reduces the in-memory structure by a factor of p. Further, if p is an exponent of 2, we can avoid some of the least significant digits of the addresses stored.

12.4

Answer:

This arrangement has the problem that P_i and B_{4i-3} are on the same disk. So if that disk fails, reconstruction of B_{4i-3} is not possible, since data and parity are both lost.

12.5

Answer:

Fewer disks has higher cost, but with more disks, the chance of two disk failures, which would lead to data loss, is higher. Further, performance during failure would be poor since a block read from a failed disk would result a large number of block reads from the other disks. Similarly, the overhead for rebuilding the failed disk would also be higher, since more disks need to be read to reconstruct the data in the failed disk.

12.6

Answer:

a. To ensure atomicity, a block write operation is carried out as follows:
   i. Write the information onto the first physical block.
   ii. When the first write completes successfully, write the same information onto the second physical block.
   iii. The output is declared completed only after the second write completes successfully.

   During recovery, each pair of physical blocks is examined. If both are identical and there is no detectable partial-write, then no further actions are necessary. If one block has been partially rewritten, then we replace its contents with the contents of the other block. If there has been no partial-write, but they differ in content, then we replace the contents of the first block with the contents of the second, or vice versa. This recovery procedure ensures that a write to stable storage either succeeds completely (that is, updates both copies) or results in no change.

   The requirement of comparing every corresponding pair of blocks during recovery is expensive to meet. We can reduce the cost greatly by keeping track of block writes that are in progress, using a small amount

of nonvolatile RAM. On recovery, only blocks for which writes were in progress need to be compared.

b. The idea is similar here. For any block write, the information block is written first, followed by the corresponding parity block. At the time of recovery, each set consisting of the n^{th} block of each of the disks is considered. If none of the blocks in the set have been partially written, and the parity block contents are consistent with the contents of the information blocks, then no further action need be taken. If any block has been partially written, its contents are reconstructed using the other blocks. If no block has been partially written, but the parity block contents do not agree with the information block contents, the parity block's contents are reconstructed.

12.7

Answer:

Reading data sequentially from a large file could be done with only one seek if the entire file were stored on consecutive disk blocks. Ensuring availability of large numbers of consecutive free blocks is not easy, since files are created and deleted, resulting in fragmentation of the free blocks on disks. Operating systems allocate blocks on large but fixed-sized sequential extents instead, and only one seek is required per extent.