10.1

Ans: This arrangement has the problem that Pi and B4i−3 are on the same disk. So if that disk fails, reconstruction of B4i−3 is not possible, since data and parity are both lost.

10.2

Ans:

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 gigabyte 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.

10.5

A.

Header				↑ 4
record 0	10101	Srinivasan	Comp. Sci.	65000
record 1	24556	Turnamian	Finance	98000
record 2	15151	Mozart	Music	40000
record 3	22222	Einstein	Physics	95000
record 4				↑ 6
record 5	33456	Gold	Physics	87000
record 6				
record 7	58583	Califieri	History	62000
record 8	76543	Singh	Finance	80000
record 9	76766	Crick	Biology	72000
record 10	83821	Brandt	Comp. Sci.	92000
record 11	98345	Kim	Elec. Eng.	80000

B.

Header				↑ 2
record 0	10101	Srinivasan	Comp. Sci.	65000
record 1	24556	Turnamian	Finance	98000
record 2				↑ 4
record 3	22222	Einstein	Physics	95000
record 4				↑ 6
record 5	33456	Gold	Physics	87000
record 6				
record 7	58583	Califieri	History	62000
record 8	76543	Singh	Finance	80000
record 9	76766	Crick	Biology	72000
record 10	83821	Brandt	Comp. Sci.	92000
record 11	98345	Kim	Elec. Eng.	80000

C.

Header				↑ 2
record 0	10101	Srinivasan	Comp. Sci.	65000
record 1	24556	Turnamian	Finance	98000
record 2	34556	Thompson	Music	67000
record 3	22222	Einstein	Physics	95000
record 4				↑ 6
record 5	33456	Gold	Physics	87000
record 6				
record 7	58583	Califieri	History	62000
record 8	76543	Singh	Finance	80000
record 9	76766	Crick	Biology	72000
record 10	83821	Brandt	Comp. Sci.	92000
record 11	98345	Kim	Elec. Eng.	80000

10.6 The relation **section** with Three Tuples is as follow.

course id	sec id	semester	year	building	room number	time slot id
BIO-301	1	Summer	2010	Painter	514	A
CS-101	1	Fall	2009	Packard	101	H
CS-347	1	Fall	2009	Taylor	3128	C

The relation **takes** with five Students for each **Section** is as follow.

ID	course_id	sec_id	semester	year	grade
00128	CS-101	1	Fall	2009	A
00128	CS-347	1	Fall	2009	A-
12345	CS-347	1	Fall	2009	A
12345	CS-101	1	Fall	2009	C
17968	BIO-301	1	Summer	2010	null
23856	CS-347	1	Fall	2009	A
45678	CS-101	1	Fall	2009	F
54321	CS-101	1	Fall	2009	A-
54321	CS-347	1	Fall	2009	A
59762	BIO-301	1	Summer	2010	null
76543	CS-101	1	Fall	2009	A
76543	CS-347	1	Fall	2009	A
78546	BIO-301	1	Summer	2010	null
89729	BIO-301	1	Summer	2010	null
98988	BIO-301	1	Summer	2010	null

The multi-table clustering for the above two instances can be taken as:

BIO-301	1	Summer	2010	Painter	514	A
17968	BIO-301	1	Summer	2010	null	
59762	BIO-301	1	Summer	2010	null	
78546	BIO-301	1	Summer	2010	null	
89729	BIO-301	1	Summer	2010	null	
98988	BIO-301	1	Summer	2010	null	
CS-101	1	Fall	2009	Packard	101	H
00128	CS-101	1	Fall	2009	A	
12345	CS-101	1	Fall	2009	C	
45678	CS-101	1	Fall	2009	F	
54321	CS-101	1	Fall	2009	A-	
76543	CS-101	1	Fall	2009	A	
CS-347	1	Fall	2009	Taylor	3128	C
00128	CS-347	1	Fall	2009	A-	
12345	CS-347	1	Fall	2009	A	
23856	CS-347	1	Fall	2009	A	
54321	CS-347	1	Fall	2009	A	
76543	CS-347	1	Fall	2009	A	

10.7

a. Every time a record is inserted/deleted, check if the usage of the block has changed levels. In that case, update the corresponding 6 Chapter 10 Storage and File Structure bits. Note that we don't need to access the bitmaps at all unless the usage crosses a boundary, so in most of the cases there is no overhead.

b. When free space for a large record or a set of records is sought, then multiple free list entries may have to be scanned before finding a proper sized one, so overheads are much higher. With bitmaps, one page of bitmap can store free info for many pages, so I/O spent for finding free space is minimal. Similarly, when a whole block or a large part of it is deleted, bitmap technique is more convenient for updating free space information.

10.10

Ans: hard disk, floppy disks and CD-ROM drives.

10.11

Ans: Remapping of bad sectors by disk controllers does reduce data retrieval rates because of the loss of sequentially amongst the sectors. But that is better than the loss of data in case of no remapping!

10.12

Ans: RAID level 1 (mirroring) is the one which facilitates rebuilding of a failed disk with minimum interference with the on-going disk accesses. This is because rebuilding in this case involves copying data from just the failed disk's mirror. In the other RAID levels, rebuilding involves reading the entire contents of all the other disks.

10.13

Ans: Successfully written sectors which are subsequently damaged, but where the damage has not been detected, are referred to as latent sector errors. In RAID systems, latent errors can lead to data loss even on a single disk failure, if the latent error exists on one of the other disks. Disk scrubbing is a background process that reads disk sectors during idle periods, with the goal of detecting latent sector errors. If a sector error is found, the sector can either be rewritten if the media has not been damaged or remapped to a spare sector in the disk. The data in the sector can be recovered from the other disks in the RAID array.

10.20

Ans: Store the data on a redundant array of independent disks (RAID). It strips data across a number of disks such that when a certain disk fails the data is not lost. It uses disk mirroring technique that duplicates data in the disk.

1.Compute the (a) transfer time, (b) average rotational latency, and (c) disk access time for the following input data for a disk? Rotational speed = 7200 rpm, sector size = 0.5 KB, average seek time = 5.7 ms, transfer rate = 24.5 MB per sec, controller overhead = 1 ms.

(a) **Transfer Time = 24.5/0.5 = 0.020 sec = 20 ms**
(b) **Average Rotational Latency = 0.5/(7200/60) = 0.004 sec= 4 ms**
(c) **disk access time = 2.0 + 5.7 + 4 + 1 = 30.1 ms**