Compute the number of blocks required for storing each of the tables in the database using each of the 3 table

Table	Record Size	Record per full block	Record per 2/3 full block	Number of full block	Number of 2/3 full block
Counselor	10+4+100=114	8192/114=71	71*(2/3) =47	100/71=2	100/47=3
Client	10+4+50=64	8192/64=128	128*(2/3)=85	5000/128=40	5000/85=59
Meeting	10+4+4=18	8192/18=455	455*(2/3)=303	10000/455=22	10000/303=34
MeetingDay	10+4+4+8=26	8192/26=315	315*(2/3)=210	15000/315=48	15000/210=72
Notes	10+4+4+4+2500+8=2530	8192/2530=3	3*(2/3)=2	2000000/3=666667	2000000/2=1000000

Number of Blocks for each Table Structure

	Clustered Heap	Clustered Sort	Clustered B-Tree
Counselor	2	2	3
Client	40	40	59
Meeting	22	22	34
MeetingDay	48	48	72
Notes	666667	666667	1000000

1. List all clients (by name) that meet with a counselor specified by id.

There are 10000 meeting for 100 counselors, meaning there is approximately 10000/100 = 100 clients that meet with each counselor.

A. Clustered Heap – The whole Meeting file must be scanned to find all clients (ids) for the given counselor id. The first block requires a random I/O for 5ms, and then the remainder is read sequentially requiring 21*0.1ms = 2.1ms.
This totals to 5ms+2.1ms=7.1ms. Next the whole Client file must be scanned to find all names for those client ids. The first block requires a random I/O for 5ms, and then the remainder is read sequentially requiring 39*0.1ms = 3.9ms. This totals to 5ms+3.9ms=8.9ms. The overall total is 7.1ms + 8.9ms = 16ms.

B. Clustered Sort – The block in the Meeting file for the given counselor id must be found, there are 22 blocks. So, number of random I/O required is $\log_2 22 = 5$ random I/O's for a total of 5*5ms = 25ms. There are on average 100 clients for each counselor, and each block contains approximately 455 records so it is likely that all of these records are in the same block. Next the names for these Clients must be found from the Client table. Unfortunately, the client ids we retrieved from the Meetings file can be randomly dispersed throughout the client table so we will have to do a scan through the entire table rather than a binary search for each block. The first block requires a random IO for 5ms, and then the remainder is read sequentially requiring 39*0.1ms=3.9ms. This totals to 5ms+3.9ms=8.9ms.
The overall total is 25ms+8.9ms=33.9ms.

C. Clustered B-Tree – Finding and reading the first block requires 2 random I/O's for 2*5ms = 10ms. There are approximately 100 clients for each counselor, and each block contains approximately 315 records so it is likely that all of these records are in the same block. Next the names for these clients must be found from the Client table. Again the clients are likely randomly dispersed across all 59 blocks, so it will simply be easier to scan the entire table rather than search each block individually. The first block requires a random IO for 5ms, and then the remainder is read sequentially requiring 58*0.1ms = 5.8ms. This totals to 5ms+5.8ms=10.8ms. The overall total is 10ms+10.8ms=20.8ms.

2. List the days that a client (specified by id) meets with a counselor (specified by id)

There are 15000 MeetingDays for 10000 Meetings, meaning there is approximately 15000/10000=1.5 days or approximately 2 days for each client-counselor meeting.

A. Clustered Heap – The entire MeetingDay table must be scanned. The first block requires 5ms, and then the rest are read sequentially requiring 47*0.1ms = 4.7ms. The total is 5ms+4.7ms = 9.7ms.

B. Clustered Sort – The first block must be found, there are 48 blocks. So, the number of random I/O required is $\log_2 48 = 6$ for a total of 6*5ms = 30ms. There are approximately 2 days for each client-counselor meeting, and each block contains roughly 315 records so it is very likely both meetings are in the same block. The total time is then just 30ms.

C. Clustered B-Tree – Finding and reading the first block requires 2 random I/O's for 2*5ms = 10ms. There are approximately 2 days for each client-counselor meeting, but again each block contains roughly 210 records so it is very likely both meetings are in the same block. The total time is then just 10ms.

3. Add a new note.

A. Clustered Heap – Record is inserted at the end of the file, must read the last block, modify it and write back. The read and write operations are each a random I/O for a total of 2*5ms = 10ms. This ignores the rare occurrence when a new block must be allocated for the new entry.

B. Clustered Sort – Finding the location for insertion requires approximately log2 (666667) = 20 random I/O's for a total of 20*5ms = 100ms. On average 50% of notes (333334 blocks) must then be shifted over to keeps the sort order. If there is enough memory to do all of the updates at once, this requires 333334 blocks to be read sequentially in 333334*0.1ms=33333.4ms and can be written in 33333.4ms. This totals to 100ms+33333.4ms + 33333.4ms = 66766.8ms

C. Clustered B-Tree – Finding and reading the leaf block requires 2 random I/O's for a total of 2*5ms = 10ms. Writing the modified block with the added note takes one more random I/O for a total of 10ms+5ms=15ms. Splits are rare so they can be ignored.4. Delete a meeting day for a client (specified by id).

4. Delete a meeting day for a client (specified by id).

A. Clustered Heap – It takes 1 random I/O to find the start of the MeetingDay file, and 47*0.1ms = 4.7ms to read the remainder of the file. The total time to read the file is then 5ms+4.7ms = 9.7ms. To rewrite the file back takes the same amount of time of 9.7ms. The total time is then 9.7ms + 9.7ms = 19.4ms.

B. Clustered Sort – It takes $\log_2 48 = 6$ random I/O's to find the block with the entry to delete. This takes 6*5ms = 30ms to complete. We must then update about half of the blocks on average, or approximately 24 blocks, to keep the file in sorted order. If there is enough memory to do all of the updates at once, this requires 24 blocks to be read sequentially in 24*0.1ms=2.4ms and written sequentially in 2.4ms.
The overall total is then, 30ms+2.4ms+2.4ms=34.8ms

C. Clustered B-Tree – Finding and reading the leaf block requires 2 random I/O's for a total of 2*5ms = 10ms. Writing the block back with a deleted meeting day requires 1 more random I/O for a total of 10ms+5ms=15ms.