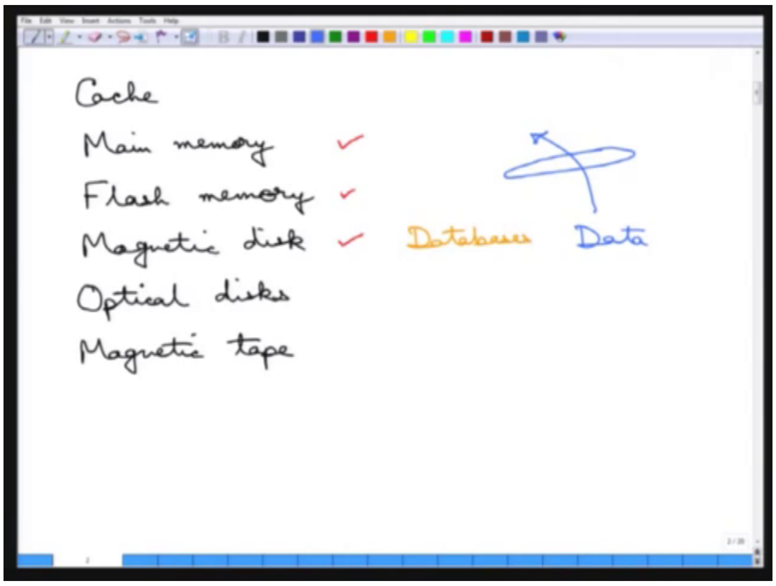
Fundamentals of Database Systems

Prof. Arnab Bhattacharya
Department of Computer Science and Engineering
Indian Institute of Technology, Kanpur

Lecture - 17
Physical Design

(Refer Slide Time: 00:24)



Welcome, so today we will talk about the physical design, so what do we mean by that is the about we will talk about the physical storage medium, where a database is stored. So, we will start off with the cache, cache as you all probably know is very fast, it is faster than main memory, but it is also very costly. And only few registers are more faster than the cache, but cache is very costly, data can be stored in the cache, but the more important problem about the cache is that it is volatile.

So, the contents vanish once the power is off and it cannot be used to persist a database. The next we will talk about the main memory, so once more main memory is quite fast, but also it is again very costly and it is volatile. So, the data cannot be persisted in main memory. And, what generally happens is that databases are quite large and the main memory that is available in a machine may or may not be able to store the entire data.

Even if it is able to store the entire data, the changes that are made in the main memory are not guaranteed to be persisted, because the anything such as power off can happen and the updates can be lost. The next that we will talk about is the flash memory, flash memory the USB drives that we use is a type of flash memory, it is a solid state drive as a D memory. Flash memory very, very importantly is non volatile, so what do we mean by that is once we write something to a flash memory it remains, even if the flash memory is taken out of power or whatever else happens.

So, it is costlier than the hard drive and we will talk about the hard drive later, but it is less costly than a memory than a main memory or a cache and it is slower than a main memory. More importantly flash memory has certain types of, has a problem about it is the read write cycle. So, the flash memory can support only a limited number of read write cycle. So, after about 10 to the over 6 read writes 10 to the over 6 is about 10 lakhs, the flash memory performance degrades.

So, a particular portion of the flash memory may start showing physical degradation and it may not be able to support anymore read writes and we will talk about flash memories a little bit more detail later. The next important physical storage medium that we will talk about is the magnetic disk, this is the hard disk that you normally understand. Magnetic disks are quite cheap compared to all the other kind of things and it, so it can be used to store a large number large amount of data, it is of course, non volatile.

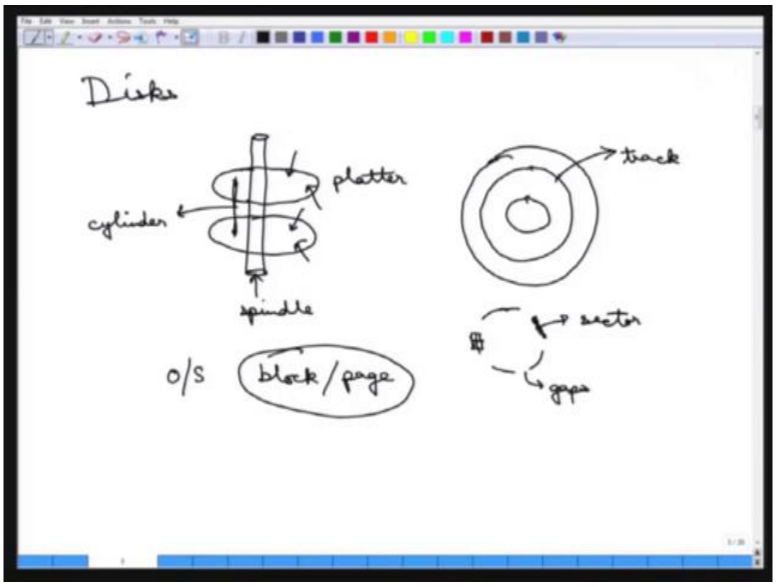
But, it is quite slow compared to a main memory and it is slower compared to a flash memory as well. So, after magnetic disk, what comes next is the optical disks. So, optical disks are your CD, DVD etcetera and these are also non volatile, these are also quite slow and, but the more important problem about the optical disk is that unless it is a rewritable medium, generally CDs and DVDs are write once and read many times. So, you write it once and persist it and you cannot generally rewrite it unless of course, it is a rewritable medium.

The last one in this type of physical storage medium that we will discuss about is called the magnetic tape. So, magnetic tapes are really, really very cheap and it is extremely slow, because you have to go through a magnetic tape completely to reach a particular byte of data. But, it is quite useful while taking backups of an entire institutional, quite while taking backups of large amounts of data and it is used very sparingly and magnetic disks tapes are generally used only for such back up purposes, it is not for day to day access, because it is extremely slow.

So, the important parts that we will worry about is the main memory and the magnetic disk, which are the hard drive and we will also talk about the flash memory a little bit. But generally, what happens is that the databases are generally stored in the magnetic disk, so these are where the databases are stored. But, importantly the operating system cannot access data that is stored in a magnetic disk.

So, the operating system must bring the data back to the main memory for any access. So, the data that is stored in a magnetic disk must be brought to main memory and perhaps move to the CPU registers etcetera depending on the architecture of the machine, but it cannot be accessed directly from the magnetic disk. So, there is an overhead involved in bringing this data back from a magnetic disk to the main memory and this is what is called the disk access time.

(Refer Slide Time: 05:45)



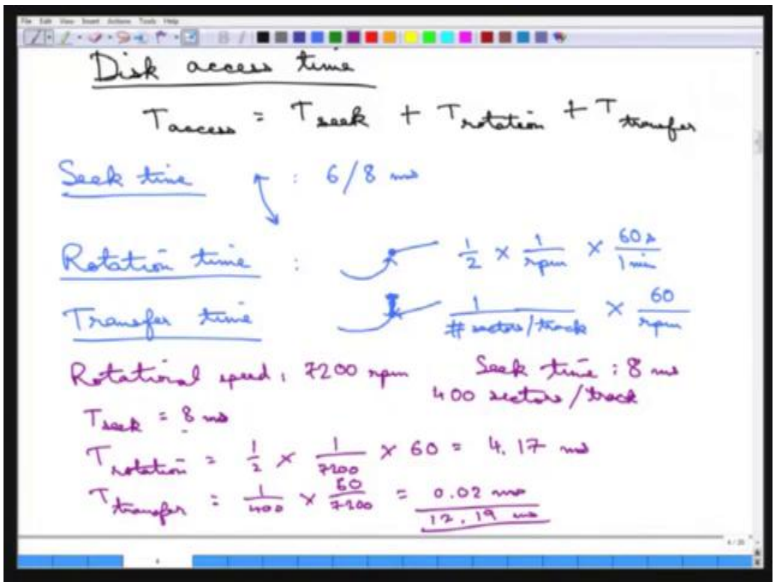
Now, before we analyze what the disk access time are, let us first try to understand what the disks are, the magnetic disks that we will be talking about. Magnetic disks generally consist of a spindle and there are certain types of cylinders attached to it, so certain types of plates attached to it, so these are different plates. And data can be stored on each of these surfaces, so this surface and the back surface, so on, so forth.

And this is called a platter, so each such disk is called a platter, this is a spindle around which the platters rotates, so essentially everything rotates and a particular point in each of this platter consists, what is called a cylinder. Now, how is data stored in a platter? So, each platter if you lay it out it looks like this, this is a complete platter of course, with a spindle in between. So, a platter has two surfaces on each surface there are different tracks on which the data is stored, so this is one track.

So, if this is one track each track is broken up into multiple sectors. So, there are different sectors that are broken up, so each of this is called a sector of data, so sectors are separated by gaps and in each sector consists of a multiple bytes of data, so this is how the entire design is there. So, each sector contains multiple bytes of data, each track contains multiple sectors, each surface contains multiple tracks, each platter contains two surfaces and a complete magnetic disk or a hard drive contains multiple platters and that is how the entire system is there.

Now, what the OS does is that, OS reads a unit of data, which is the block or page of data, this is the unique by which the OS reads it and each sector consists of multiple blocks of data.

(Refer Slide Time: 07:52)



Disk access time

$T_{access} = T_{seek} + T_{rotation} + T_{transfer}$

Seek time : 6/8 ms

Rotation time : $\frac{1}{2} \times \frac{1}{rpm} \times \frac{60 s}{1 min}$

Transfer time $\frac{1}{\# sectors/track} \times \frac{60}{rpm}$

Rotational speed : 7200 rpm

Seek time : 8 ms

400 sectors / track

$T_{seek} = 8$ ms

$T_{rotation} = \frac{1}{2} \times \frac{1}{7200} \times 60 = 4.17$ ms

$T_{transfer} = \frac{1}{400} \times \frac{60}{7200} = 0.02$ ms

$\underline{12.19 \text{ ms}}$

And having understood this, let us now try to understand, what is a disk access time, so a disk access time. So, the smallest amount of information that can be read or written from a disk is a sector, so one must reach a particular sector to read. Now, what happens is that, if we go back to the previous figure in the cylinder there is a read write head, this is the read write head of the disk that moves back and forth across the surface of a track radically, this movement is only radically.

So, the read write head the arm, the head of the read write arm, this is the read write arm, read write arm, the head of the read write arm can place itself on any track. But, having placed itself on any track it cannot access any sectors, so for that the track must rotate. So, that the sector is placed under the particular read write arm and these movements of the read write arm and the rotation are all mechanical movements. So, that is why there is a physical limit, a mechanical limit as to how much faster hard drive can be.

So, coming back to disk access time, the disk access time consists of three kinds of things. So, disk access time T access consists of three different times, time to seek, time to rotate and time to transfer and we will describe each of this in more detail next. So, the seek time is the time for the arm of the read write head to place itself on the particular track that it wants to. So, it is a radial motion of the read write head, this is the read write head and this is the radial motion of the read write head; that is the seek time.

The rotation time is the time for the disk to rotate; such that the particular sector that is needed is placed under the head of the read write arm. So, it can be a complete rotation or it can be a zero rotation with the sector already under the head of this read write arm, so that is the rotation time. And finally is a transfer time, so a transfer time is the time; such that the complete sector of data is read, so suppose this is the sector of data; that is needed to be read by the OS.

So, this complete time that this rotates; such that that the entire amount of data is passed under the read write arm. So, these are the three kinds of things, the typical seek times are generally of the order of 6 to 8 milliseconds, this is milliseconds mind you. The rotational latency is well we can say it is half the time to rotate on average, because it can be either 0 or it can be a compete rotation. Remember, the disks can rotate only in one way, so if the read write arm has just missed it, so it will take a complete rotation to be back.

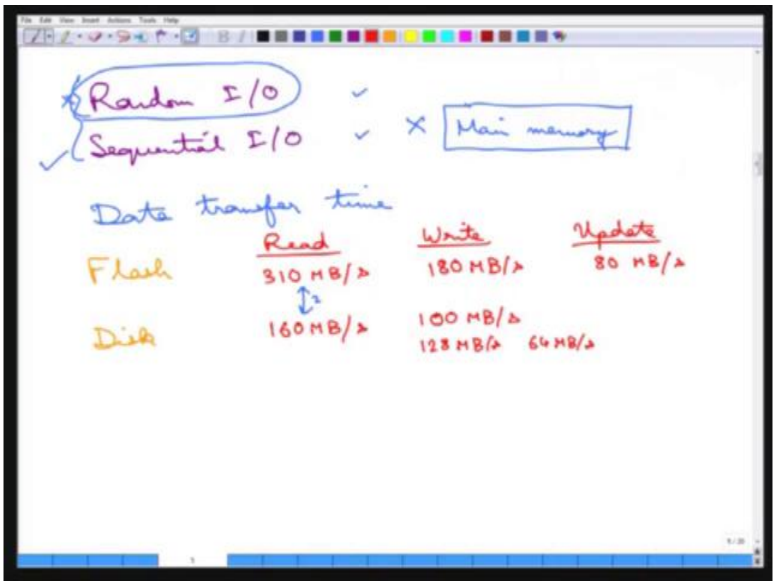
So, it is essentially half into 1 by rpm, this is the revolution power minute. So, to convert it into seconds need to be conversional 60 minutes 60 seconds; that is the rotation time. And the transfer time is the number of sectors, so it to read one sector it is a number of sectors per track into that similar 60 by rpm, that we will do. So, this is the number of sectors; that is, where, so if to read one sector you require this amount of time and to convert it into seconds this is what is being done.

So, what does some typical disk parameters, so this typical disk parameters, let us take an example to do it. So, suppose the rotational speed of a disk is say your 7200 rpm and suppose the seek time given is 8 seconds, these are given by the manufacturer of the disks. Now, what is the typical time to read one sector? We can compute it in the following manner.

So, T seek is your 8 milliseconds; that is given the T rotation will be you can compute it using the formula half into your 1 by rpm, so 7200 into 60 that means, seconds, so this sum out to be around 4.17 milliseconds and the T transfer we can again compute take similarly. So suppose there are 400 sectors per track, so this is 1 by 400 into 60 by 4200, so these are comes out to be 0.02 milliseconds. So, if we add all these up together we get a total time of 12.19 milliseconds; that is the average time to read one sector from the disk.

Now, as you can see the disks are quite slow, because to read one sector this requires 12 milliseconds and this seems to be extremely slow, what happens is that, this is what is called the random IO.

(Refer Slide Time: 13:23)



Random I/O ✓

Sequential I/O ✓ × Main memory

Data transfer time

	Read	Write	Update
Flash	310 MB/s	180 MB/s	80 MB/s
Disk	160 MB/s	100 MB/s 128 MB/s	64 MB/s

So, there is two important things that we must understand one is called the random IO and the other is the sequential IO. What is called random IO is that, if you remember in the last computation we are trying to read an average time for a sector. So, the sector is essentially assumed to be randomly placed anywhere on the disk and we are trying to find the time to do it, so that is why it is called a random IO. So, you are just given the address of a particular sector and you want to read that; that is the random IO.

What happens in the sequential IO is that? Your read a particular sector and the next sector to be read is sequentially placed after that the sector that you are already reading. So, there is no seek time the seek time is 0 there is no rotational time as well, because the read write arm will is already placed on the correct track and just before the sector; that is being read. So, the only time that is required is the transfer time, so there is the big difference between the random IO and sequential IO.

So, to highlight the point once more, suppose you are reading more than two sectors it makes sense to place the two sectors one after another. Because, to read the two sectors, then the first sector needs to be written in a random manner, but the second is just after it, so there is no seek time etcetera, so second sector can be read much faster. So, it makes sense to put the two sectors sequentially on the disk and; that is why sequential access are much fast; that is why sequential access is, what is preferred by the database design.

On the other end if the two sectors are placed randomly anywhere on the disk it will take much more time to read the two sectors. So, if there is a big large table or a big large database it makes sense to layout the layout the data in the database physically in sequential sectors, so that is what the typical thing about this is. So, the data transfer rate etcetera are computed using this sequential IO time and random IO time and sequential IO time, it actually is a little bit more complicated, because there are gaps between the two sectors.

But, it is roughly taken to be the, what we just did the computation, the one thing to note is that the sequential IO is much faster than the random IO, but much more faster is the main memory time. But, main memory is more costly, so the data is generally stored on the disks only, what is being done is that out of these two as I already said most algorithms will try to avoid the random IO time as heavily as possible. So, it will rather do a sequential IO time and not a random IO time, because sequential IO is much faster than main memory.

One more important thing that the database designers generally take into account is that the time to compute in the main memory is ignored. Why is it ignored? Because, the time to bring a particular data to the main memory is, so large compared to the computation time in the main memory. So, nowadays the machines are gigahertz machines, so it is essentially nanosecond or some orders of magnitude larger than nanoseconds time to do any computation on the data; that is brought to the main memory.

However, the time to bring the data is milliseconds, so it is let us say microseconds as compared to milliseconds, microseconds on the CPU computation time on the main memory versus milliseconds time to bring the data. So, the microsecond's time can be ignored and it is generally ignored by the database designers. So, the database designers essentially do not take care of how to reduce the main memory time, because it is only optimizing on the microseconds it rather ties to tackle more on the milliseconds time and more on the random IO time.

So, the algorithms are designed, so that they reduce the random IO time this is one very important point that we need to understand from this physical design is that the database algorithms are typically designed; such that the random IO time is minimized. This was the traditional database story; nowadays as we have already mentioned quickly that there is a flash memory that has come into the picture.

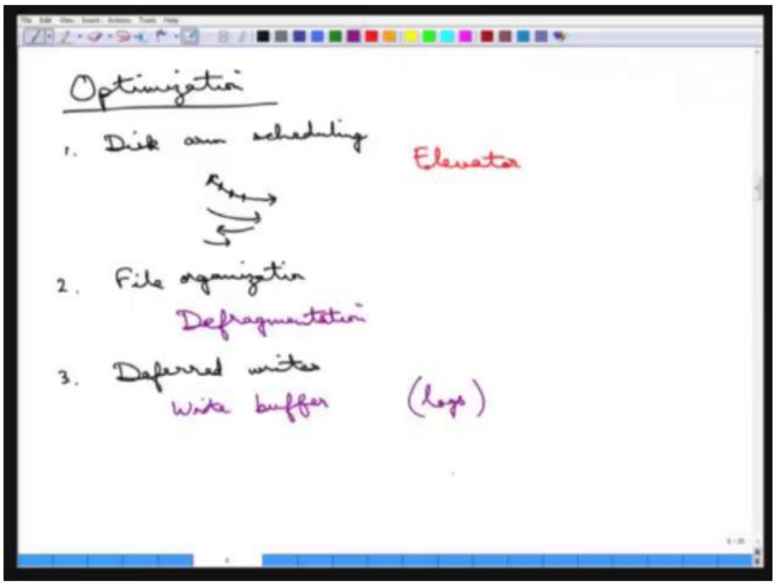
So, the flash memory it is, now becoming cheaper in the sense that some of the databases can be stored on the flash or at least some portions of the database can be stored on the flash and flash memories are typically faster than a hard drives or the magnetic disk that we are talking about. So, let us take some typical numbers, so flash drive read time for flash can be around the read throughput can be around 300 megabytes per second this is a typical time.

So, I mean different flash drives will do different the typical read write times are 180 megabytes per second and flash drives can update, but update is much slower it can be only 80 megabytes per second. As opposed to disks, disks are read times typical read times can be 160 megabytes per second and write times can be depending on sequential IO or this IO, let us say about 100 megabytes per second and update times are essentially just twice of this 100 megabytes per second.

So, you see there is a cost ratio of about two times here and this can be depending on the whether it is a sequential MB or this thing. So, this can be as large as 128 megabytes per second or 64 megabytes per second, so there is again a ratio of about 2 to 3. So, flash drives are about twice as fast as magnetic disk, but it is also quite costly. So, the database designer can now optimize as to how much data to put on the flash drive.

So, there is a cost versus access time optimization that, that database designer can do also flash drives has this problems that data needs to be deleted in much larger blocks than magnetic disks and it erodes after 10 to the over 6 read write cycles.

(Refer Slide Time: 19:40)



Optimization

1. Disk arm scheduling Elevator

2. File organization
Defragmentation

3. Deferred writes
Write buffer (logs)

So, the disk block access that we are talking about, so there are two kinds of optimizations, then we done about this disk block access the first is the disk arm scheduling. So, what it does is that this kind of algorithm first tries to see, what are the sectors that are going to be read or written by the database and then, it schedules such read and write unless they are dependent on each other assuming that independent, such that the disk arm is moved from one sector to another from one track to another.

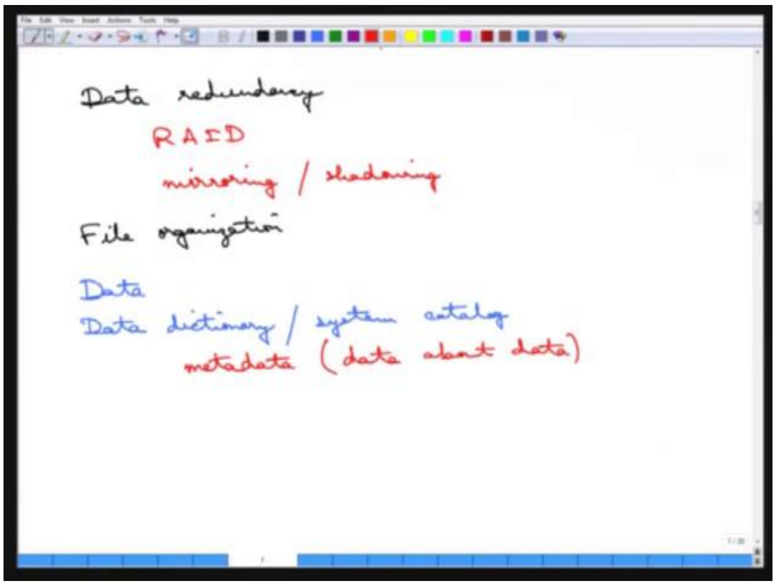
So, it moves surface the first time this data; that is sitting on the first track is read or written, then this is accessed, then this is accessed and so on, so forth. So, that there is only one disk arm movement rather than the back and forth movements. So, that is the disk arm scheduling operation and then, there is an algorithm called the elevator algorithm, which tries to do that.

Then, there is of course, the other kind of this is one kind of operation the other kind of optimization is the file organization and we will talk about this more detailed, but the file organization it tries to do this we all have seen these terms, somewhere in the defragmentation. So, what it tries to do is if a file is fragmented if a file is broken into pieces, then it goes into different parts of the disk.

So, then there are more random IOs that are needed to read the complete file, instead if it is defragmented the entire file is stored together and after one random Io the rest are sequential IO, so that is faster for a database, so that is what it being done. Then, there are other kinds of optimization that it can do it can use this something called a deferred writes, so essentially the databases is use this write buffers.

So, as soon as a write is available the database does not go and write it back to the disk it puts it back into the write buffer and only after sufficient writes are being buffered it, then goes ahead and writes them one after another sequentially in that thing. So, this mean times logs etc and we will see how this is done later, but that is the basic idea is that it is the writes are deferred it is not one after another it is not immediate; that is the main point. Fine and then, let us talk about the related issue, which is on data redundancy.

(Refer Slide Time: 22:17)


Data redundancy
RAID
mirroring / shadowing
File organization
Data
Data dictionary / system catalog
metadata (data about data)

So, data redundancy is done to improve the reliability. So, instead of storing the data in namely one disk or one place it is stored in multiple places; such that even if one copy is lost due to some mechanical problem etcetera, it can be retrieved from the other things. And there are this famous terms are called raid, so redundant arrays of independent disk.

So, this is essentially independent the disks are maintained independently and each one is a redundant copy of the other, then this mirroring is allowed and you have probably heard the term in terms of web mirror.

So, inter website is mirrored or shadowed, so that one can access it from any of those sites. This is more or less all about the data redundancy, then there are this problems of how to store the files the issue of the file organizations. So, the file organization the files can be stored in a sequential manner or it can be hashed depending on the block address etcetera. So, the database stores the data actually about the data, then it also stores something called the data dictionary or the system catalog, which stores the data about data, so the term for that is called meta data, this is data about the data.

So, how is the primary data stored, so how are the tables organized, what are the tables mean, what are the schematics of the table, what are the types of the table etcetera, etcetera. Now, one thing to notice that, why is this being highlighted, because meta data is accessed much more often than a particular piece of data. So, whenever anything about the table is read the meta data about the table needs to be written.

So, it makes sense to have some algorithms that can read the meta data faster. So, it can be stored in a faster memory for example, a particular. So, it can be brought to main memory and persisted in main memory, because generally meta data does not change much it does change a little bit about the statistics of the data, but the type of the data does not change much. So, the meta data can be brought into main memory, because it is accessed much more often.

So, this is all about the data, so of course, what I meant to say is that statistics about the data is also maintained. So, and that is all about this storage of physical design and next we will cover the indexing.