03 Database Storage I

0-0.16
Oh guys we’re starting
~~兔崽子们，我们开始吧~~
孩儿们，我们开始吧

0.16-0.17
 welcome
欢迎回来
0.17-0.18
again

0.18-0.21
 sorry for being away last week
抱歉，上周我并不在
0.21-0.22
but I mean in town now
但我现在回来了
0.22-0.25
 so real quick we’ll pause around for my our DJ rapper tables
有请我们的DJ出场
0.25-0.30
 so I don’t think people know how bad you are right
~~我并不觉得人们知道自己有多糟~~
so 我觉得有很多人不知道你牛逼与否
0.30-0.35
so once you show the people a little bit someone what you can do write letter i t
来给大伙儿秀一段
0.40-0.48
this is real yes we do have a course DJ right
这是Real，我们这门课的DJ

0.48-0.49
I’ve always won one now I have one
我之前总想找个DJ，但现在我有了Real
0.49-0.50
how was your summer
你们的暑假过得怎样?
0.50-0.51
good
不错
0.51-0.57

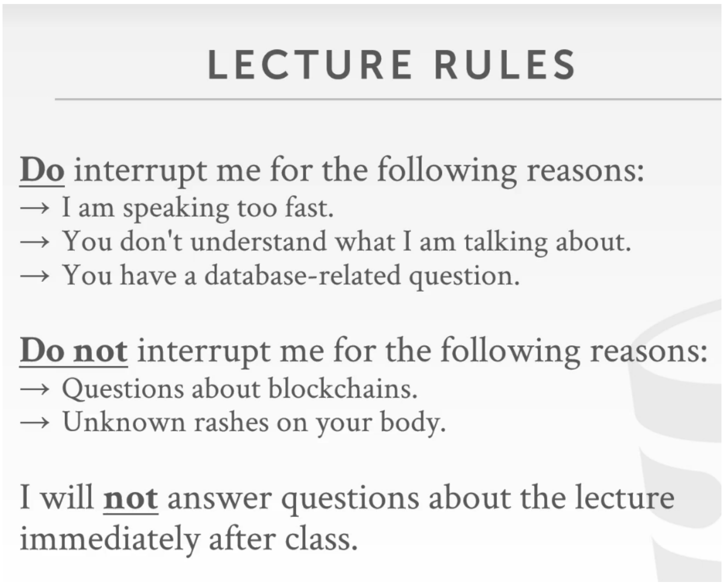
would you out there
有去过哪里
Did you do anything?
你有做什么吗?
0.57-0.59
 okay awesome
Ok，这很棒
0.59-1.01
 yeah all right what wilson do?
Wilson做了啥?
1.01-1.08
 yeah yeah there’s that yeah

1.08-1.13
I mean so we’re still been a database system here at Carnegie Mellon
我是说，so 我们仍然在卡内基·梅隆大学搞数据库系统
1.13-1.16
so we’re still grinding on that’s just taking all my time
我依然在尽全力搞这事情
1.16-1.20
right you know it is yes that is true
你知道，那事是真的
1.20-1.21 （啥也不敢说，啥也不敢问）
I did get somebody pregnant
我老婆确实怀孕了

1.21-1.25
 okay so let’s go through this real quickly
Ok，让我们来快速看下今天的课程

1.25-1.28
all right so I only care about two things in my life
在我的人生中，我只关心两件事情
1.28-1.34
~~I say this every semester~~ number one is my wife number two is databases
首先，我最关心的就是我的妻子，其次则是数据库
1.34-1.36
I don’t give up anything else right
我不会放弃这一切
1.36-1.38
my family vote for Trump I don’t talk to them
我家给特朗普投了票，但我不想和他们说话
1.38-1.40
I you know I have the dog that’s fine
你们知道，我养了一条狗
1.40-1.43
but like I don’t have any other hobbies it’s just databases
我除了数据库以外没有其他爱好了
1.43-1.46

right and again so people think I say crazy stuff
因此，人们觉得我是一个很疯狂的人
1.46–1.52
and so if I start talking fast, I want you to please interrupt me to slow down   right
如果我讲的太快，你们可以打断我，让我放慢语速

1.52–1.54
and just tell me repeat myself
告诉我停下来重复之前的内容
1.54–1.56
I say you’re sounding stupid
~~你们听到这个可能会觉得我有点愚蠢~~
我的意思是说你觉得听不懂的话
1.56–1.58
 you’re going too fast slow down
如果你们觉得我讲的太快，那我可以慢下来
1.58–2.01
you can also interrupt me when you don’t understand what I’m talking about
当你们不理解我正在说的东西的时候，你们也可以打断我
2.01–2.03
and I actually encourage you and I want you to do this
实际上，我也鼓励你们，也想要你们这样做

LECTURE RULES

Do interrupt me for the following reasons:
→ I am speaking too fast.
→ You don't understand what I am talking about.
→ You have a database-related question.

Do not interrupt me for the following reasons:
→ Questions about blockchains.
→ Unknown rashes on your body.

I will **not** answer questions about the lecture immediately after class.
2.03–2.07
because I know this like I don’t know if you guys knew it
因为我知道我明白我讲的内容，但我不知道你们是否了解这些
2.07–2.10
so please stop me，if you don’t understand what I’m talk about or already have a server database related question
如果你不理解我说的，或者你有一些数据库相关的问题，那么请打断我
2.10–2.15
what you can’t stop me about or stupid things like you have a question up blockchain like in previous years
如果你有区块链之类的问题或者其他愚蠢的事情，请不要打断我讲课

2.15–2.17

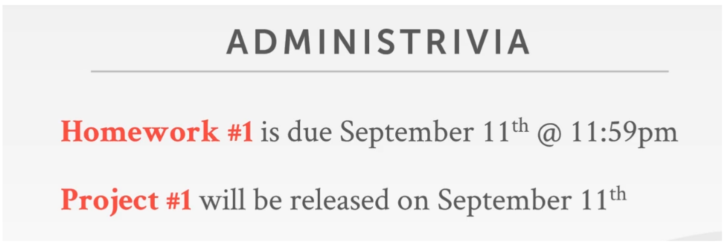
I don’t care about blockchains
我并不关心区块链
2.17–2.17
okay

2.17–2.21
this course is not about blockchains, that’s not we’re talked about here
这门课并不是关于区块链的，我们也不会在这讨论这个
2.21–2.25 （狗命要紧？？？）
if you also have a weird rash on your body, don’t share it with the class in front of the middle of lecture
如果你的身体上也有怪异的皮疹，请不要坐在课堂中间传染给别人
2.25–2.27
this happened the first year
这在大一有发生过
2.27–2.27
it’s weird
这很糟糕
2.27–2.28
don’t do that okay
请不要那么做，Okay？！
2.28–2.31
so all right

2.31–2.32
what does this mean?
这意味着什么呢?
2.32–2.35
so this means that I want you to interrupt me as much as possible
这意味着我想让你们尽可能多的打断我
2.35–2.37
right don’t be embarrassed, there’s no stupid questions
不要害羞，请尽管提问，也不要觉得问题愚蠢
2.37–2.40
so if you have a question though about the lecture
如果你对课程有疑问的话
2.40–2.43
don’t come up in front of me at end of the class
请不要到快下课的时候跑到我面前提问
2.43–2.45
be like Oh what about slide three what did you mean all this
就比如你可以这样说，“我对幻灯片3上你所说的东西有疑问，你能告诉我吗”这种
2.45–2.46
I will not answer these questions
我不会去回答这些问题
2.46–2.49
because if you have a question while I’m talking
因为当我正在讲课的时候，如果你有疑问
2.49–2.50

then somebody else probably is a question
然后其他人可能也会有一个问题想问
2.50-2.56
and I rather just discuss in front of everyone rather than a line of people saying what about slide forward what about slide six over never again
我更想在所有人的面前讨论这个问题，而不是只对某几个人讨论某张幻灯片上的内容
2.56-2.56
okay

2.56-2.58
so again I will not be offended
So，我不会生气
2.58-3.00
I don't care just interrupt me if you have questions
如果你有任何问题，请打断我，我不会在意你这样的做法~~，相反我很喜欢这样~~
3.00-3.01
okay

3.01-3.03
so with that any questions
对此，你们还有任何疑问吗?
3.03-3.05
okay

ADMINISTRIVIA

Homework #1 is due September 11th @ 11:59pm

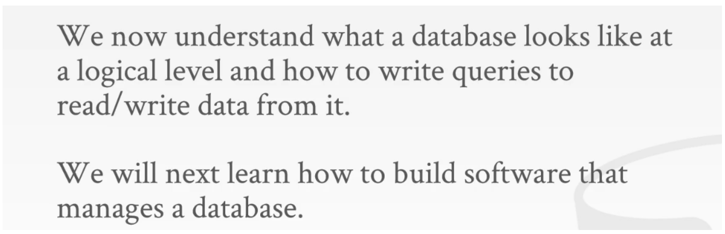
Project #1 will be released on September 11th

3.05-3.12
so I also post on Piazza that homework one is out it'll be due on the on the 11th next week
So，我也在Piazza上发布了作业1，你们需要在下周9月11号之前做完并上交
3.12-3.17
and then we'll also be releasing project one on github next week as well
然后，我们也会在下周那天将Project 1在Github上放出
3.17-3.20
just sort of keep in mind of this is only your schedule with coming up
要注意，这是你们接下来的日程规划
3.20-3.22
who here has finished the first homework
这里有人将第一个作业完成了吗?
3.22-3.23
I think somebody already did and got a perfect score
我认为某些人已经完成并得到了一个非常完美的分数
3.23-3.25
and they're not here
看来他们并不在这里
3.25-3.26

awesome even better
~~算了，就这样吧~~

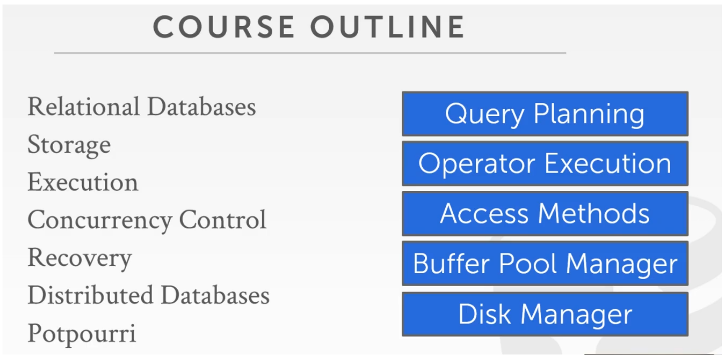
3.26-3.30
okay so again it's we're giving you a SQLite file
我们给了你一个SQLite文件
3.30-3.32
we're making you write some simple queries
我们让你们去写一些简单查询
3.32-3.35
it's sort of to force you to do some of the things we talked about in the second lecture
并强制让你们去实现一些我们在第二节课中所讲的内容
3.35-3.39
and this will actually be the only time you were write SQL query for the rest of the semester
这实际上也是你们在剩下的学期里唯一一次写SQL查询的机会了
3.39-3.42
all the projects you will be doing and it will be in C++
所有的项目，你都需要用C++来编写
3.42-3.44
and all the homeworks will be pencil and paper
并且，所有的作业都是用笔和纸来完成的
3.44-3.45
okay

3.45-3.50
so this I want at least have you take a database course at least you know write some sequel once in your life
因此，我想要让你们最起码在上数据库课程的时候写些SQL语句
3.50-3.50
okay

3.50-3.53
all right so let's jump right into this
好了，让我们上课吧

We now understand what a database looks like at a logical level and how to write queries to read/write data from it.

We will next learn how to build software that manages a database.
3.53-4.02
so last class at least the last two classes we spent time talking about what a database looks like at a high level
在前两节课中，我们花时间讨论了从一个高级层面来看待一个数据库该是怎么样的
4.02-4.05
the logical level what is the application programmers see
~~以及应用程序开发人员在逻辑层所看到的是什么样的~~
以及从逻辑层面来讲，对于应用开发人员看到的应该是什么样子
4.05-4.06

they see relational tables
他们看到了关系表
4.06-4.07
they see SQL
他们看到了SQL
4.07-4.15
and so now what we want to talk about is how are we actually gonna build the software that's gonna manage this database system
So，现在我们想去谈论的是我们实际该如何去构建管理这种数据库系统软件
4.15-4.17
alright this is essentially what the course is about
简单来讲，这就是这门课所要教的内容
4.17-4.23
right how do we actually going to store database and allow queries to execute and derived new data from it
我们该如何在数据库中保存数据，并且允许从数据库中执行查询以及处理得出新的数据



COURSE OUTLINE

Relational Databases
Storage
Execution
Concurrency Control
Recovery
Distributed Databases
Potpourri

Query Planning
Operator Execution
Access Methods
Buffer Pool Manager
Disk Manager

4.23-4.27
so the overall outline for the course again
如图是我们的课程大纲
4.27-4.30
we've already covered the logical part at the top relational databases
我们已经讨论了图中最上面的关系型数据库
4.31-4.38
and now we're starting to be going through different parts of a database and some one-by-one as if they're like as if different layers
现在，我们要对数据库的这些不同部分进行逐个学习
4.38-4.41
so we'll start talking about storage query execution Concurrency control recovery
So，我们会去讨论存储，查询执行，并发控制，恢复
4.41-4.43
and then we'll get to like distributed databases
然后，我们会去讨论分布式数据库
4.43-4.45
and other topics at the at the end
以及在最后会去讨论些其他话题

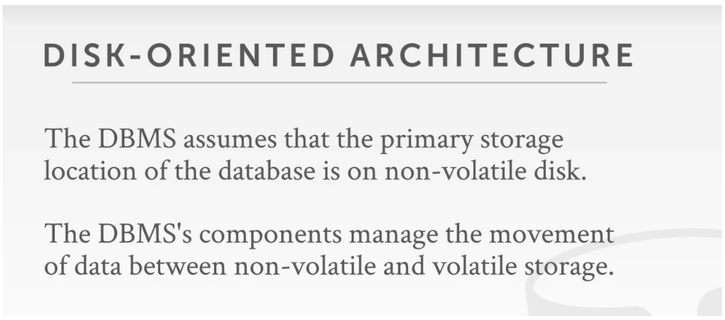
4.45-4.49
and the way to think about this, this is a gross approximation of what a database is looks like
这张幻灯片上所展示的就是一个数据库中应该包含的东西
4.49-4.52

 it’s just a bunch of layers built on top of each other
它就是建立在这些层面之上
4.52-4.53
 right

4.53-4.58
so we’re gonna focus on this lecture and the next lecture on the on the disk manager
这节课以及下节课，我们会集中讲解磁盘管理器
4.58-5.00
 how do we actually store data you know on files on disk
即我们实际该如何将数据存储在硬盘的文件中
5.00-5.05
and then above that once we know what API we’re gonna expose to the upper levels in the system
然后，在此之上，我们需要了解哪些API需要暴露给上层系统
5.05-5.09
we start adding those extra levels to it till at the end we have a full-featured database management system
然后，我们开始逐步添加额外功能，直到最后我们会有一个功能完备的数据库管理系统
5.09-5.12
so that’s the way to think about we’re talking at this point
这就是我们这个课所要讲的东西
5.12-5.15
 so we’re no more sequel stuff no more relational model stuff
因此，我们不会再去讨论SQL和关系模型之类的东西
5.15-5.20 ******
aspects of it are gonna be important for how we make different design decisions in our system
这些方面对于我们如何在我们的系统中做出不同的设计决策而言，将非常重要

5.20-5.24
but we have to figure out how do we actually gonna run sequel queries
但我们必须实际弄明白该如何运行SQL查询
5.24-5.29
 but it’s not like we’re gonna worry about how to write you know complex sequel queries
但我们并不用去担心该如何写出那些复杂的SQL查询
5.29-5.30
because we’ve already done that
因为我们之前已经练过了（注：家庭作业和前两集课程的内容）
5.30-5.30
 ok

5.30-5.37
it’s again we’re focusing on different levels in the system different layers one by one and then you know sort of going up the stack
So，再说一遍，我们会专注于该系统的不同层面，对它们逐个深入讲解

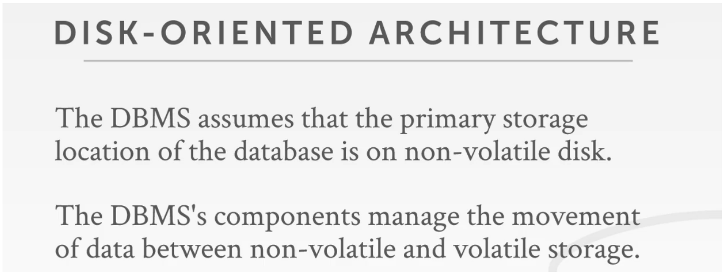

DISK-ORIENTED ARCHITECTURE

The DBMS assumes that the primary storage location of the database is on non-volatile disk.

The DBMS's components manage the movement of data between non-volatile and volatile storage.

5.37–5.40
 so as I said the first lecture as well
正如我在第一节课所讲的那样
5.40–5.44
 this course is about building disk oriented database management systems
这门课是关于构建面向磁盘的数据库管理系统
5.44–5.47
so just to reiterate what I mean by that
这里只是要重申下我的意思
5.47–5.54
 if Disk oriented database system is one where the software makes the assumption that the primary storage location of the database is on disk
假设数据库的主要存储位置都是放在磁盘上的，这种软件被称为面向磁盘型数据库系统
5.54–6.03
and so that means, that any single time we have to execute a query it may actually want to run you know to want to access data that's not in memory
这就意味着，每次我们执行查询时，所要访问的数据都不在内存中

6.03–6.04
and we got to go out in the disk and get it
我们要进入磁盘，并取得该数据
6.04–6.14
and there's a bunch of components in how we design our software that are going to be based on this assumption to protect ourselves
在设计我们的软件时，需要基于一些假设来设计一系列组件，以此来保护我们的系统，
from losing data having~~ you know ~~invalid or incorrect data
比如数据丢失，保存无效或错误数据等诸如此类的情况


DISK-ORIENTED ARCHITECTURE

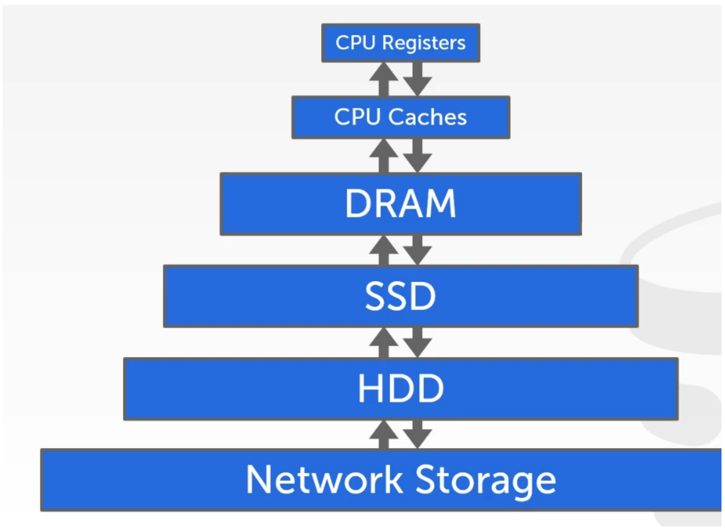
The DBMS assumes that the primary storage location of the database is on non-volatile disk.

The DBMS's components manage the movement of data between non-volatile and volatile storage.

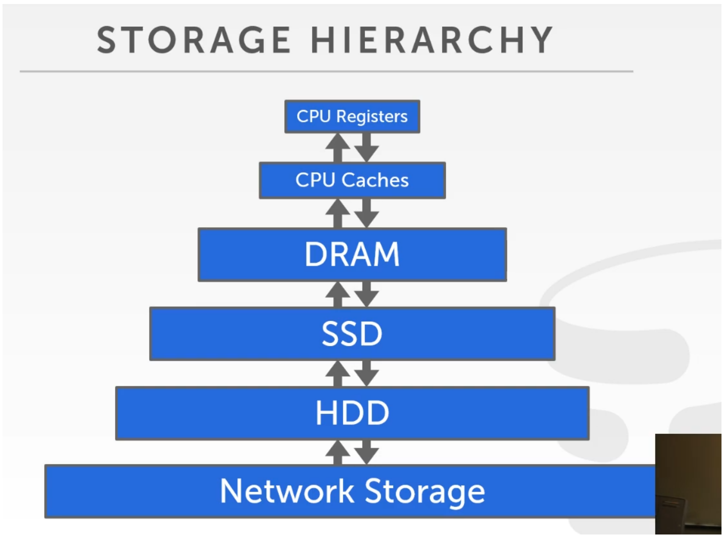
~~6.08–~~

~~我们需要基于这种假设，来防止我们丢失数据，例如，保存了无效或者错误的数据~~
6.14–6.18
right so this is something's gonna permeate all throughout the entire system
这是整个数据库系统都需要考虑的事情
6.18–6.21

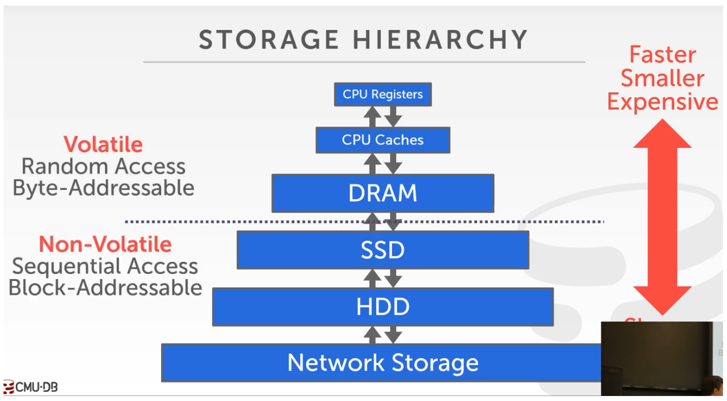
and we need to be aware of at any given time something we’re trying to read is not in memory
我们需要意识到在任何时间我们尝试所读取的东西都不在内存中（知秋注：比如我们做查询操作，我们潜意识里都认为是从硬盘存储的数据中来查找，因为我们都是将它们持久化在硬盘中的）
6.21–6.25
so to understand this a bit
So为了理解这一点
6.25–6.29 ******
further we want to make the distinction between volatile and non-volatile storage
此外，我们要区分易失性存储和非易失性存储
6.29–6.36 *******
so essentially what we’re trying to do is we’re trying to have the data system manage the movement of data from non-volatile storage into volatile storage
简单来讲，我们所要的数据系统管理了数据从非易失性存储到易失性存储的移动
6.36–6.39
so what I mean by that
So，我的意思是什么呢?



6.39–6.43
the way you think about the storage hierarchy of computers is as such
计算机的存储结构层次如图所示
6.43–6.43
right


STORAGE HIERARCHY

6.43–6.47

you sort of start at the very bottom or sorry start at the very top
我们要从最上面的存储设备开始
6.47–6.51
you're gonna have things that are be very fast very expensive they're very small
这些存储设备的速度非常快，也非常昂贵，并且它们的存储容量非常小
6.51–6.52
 so the way to think of this is like a spectrum
你可以把这张图当成光谱来看待
6.52–6.58
at the top you have things that like CPU registers or CPU caches l1 l2 l3
在最上面，我们会有CPU寄存器，CPU L1、L2和L3缓存之类的东西
6.58–7.00
these things are very fast
这些东西的速度都非常快
7.00–7.02
 right but they're doing very small capacity
但它们的容量却很小
7.02–7.04
because they're like literally sitting on the CPU itself
因为从字面上来看，它们都是CPU里面的东西
7.04–7.06
 then we're gonna have DRAM
接着，我们会有DRAM
7.06–7.09
and then below that we'll have SSDs spinning these hard drives in network storage
在它下面我们还有SSD和HDD这些网络存储设备
7.09–7.13
again at the bottom they're very large but they're much slower and they're cheaper
因为它们位于这张图的下方，它们的容量很大，但速度要慢得多，而且它们的价格也要便宜得多


7.13–7.18
so again the dichotomy that we care about is this division line
我们所关注的是这条分割线，它将这张图一分为二
7.18–7.21
here so anything above this line is volatile
这条线以上的是易失性存储
7.21–7.22
 what does that mean
这意味着什么呢?
7.22–7.31
 yes yeah she says it's not persisted when you lose power absolutely

这位同学说的没错，当你供电完全断开时，这些数据就不会持续存在
7.31–7.38
so all these storage devices require constant energy right like electricity to maintain whatever they’re storing
因此，所有的这类存储设备都需要稳定的能量（比如：电能）去维持它们所存储的东西
7.38–7.39
right

7.39–7.44
you pull the power from your computer, everything in DRAM gets wiped everything on your CPU caches get wiped
当你拔掉你计算机的电源时，所有在DRAM以及CPU缓存中的东西都会被擦除
7.44–7.46
everything below this is non–volatile
这条分割线以下的就是非易失性存储
7.46–7.49
meaning it doesn’t require constant power in order to persist whatever was stored in it
这意味着它不需要稳定的能量就能持久存储其中保存的所有内容
7.49–7.53
so that’s at a high level that’s the major thing that we care about
这就是我们站在一个高级层面所主要关心的事情

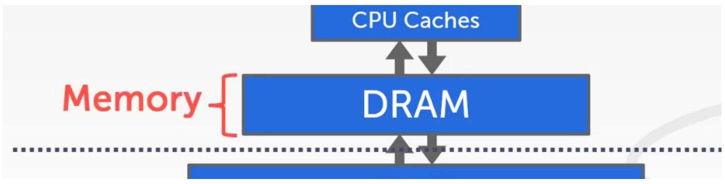
7.53–7.55
we have to move data from here up into here
我们必须将数据从非易失性存储移动到易失性存储
7.55–8.02
there’s other aspect that are going to affect how we design our software
还有一些其他方面会影响我们该如何设计我们的软件
8.02–8.04
and that has to do how’re actually can access this data
即我们该如何真正可以访问这些数据
8.04–8.06
so if it’s in volatile storage
如果数据是保存在易失性存储中的话
8.06–8.09
 it’s gonna support fast random access
那它就支持快速随机访问
8.09–8.13
maybe we can jump to any sort of address location in the storage device very quickly

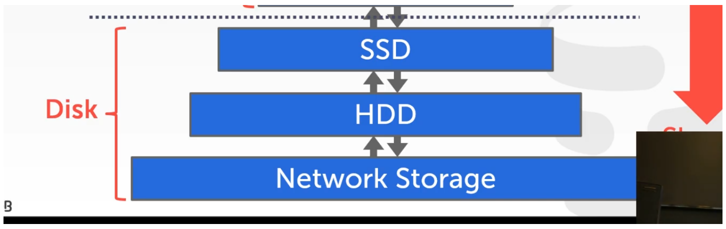
这就意味着，我们可以快速跳转到该存储设备中的任意位置处
8.13-8.20
and we’re gonna get roughly the same performance no matter what order we access the access things
无论我们访问数据的顺序是什么样的，访问的速度都大体一致
8.20-8.24
I mean if I jump to this location, this location and maybe back to another location
我的意思是，如果我跳到这个位置，然后又从这个位置跳转到另一个位置
8.24-8.27
I’m gonna get approximately the same latency the same speed
访问这两个位置的延迟和速度几乎是一样的
8.27-8.34
sorry non-volatile storage they’re gonna have instead of having byte addressable access to and they have block addressable access
对于非易失性存储，它们具备的是块可寻址能力，而不是字节可寻址
8.34-8.39
so in byte addressable access that means, I want to read 64 bits at this stores location
字节可寻址访问的意思是，我想去读取该存储位置的64bit的数据
8.39-8.42
I can just go read just that 64 bits and get exactly what I want
我也只能读到64bit大小的数据，并得到我想要的内容
8.42-8.44
I’m oversimplifying
我说的可能太过简单了

8.44-8.50
but that’s essentially how from the programmers perspective of us as the database system developer that’s what we see
但这实际上就是我们作为数据库系统开发人员所看到的东西
8.50-8.56
in a non-volatile storage, we can’t go get exactly just a 64 bits that we want
在非易失性存储中，我们无法准确的得到我们想要的64bit大小的数据
8.56-9.00
we have to go get the block or the page that has that data that we want
我们所得到的是包含我们想要的数据的块或者页面
9.00-9.01
and we have to get everything that’s along with that page
我们必须得到该页面上所有的内容
9.01-9.03
so I only want to read 64 bits
我只想读取64bit大小的数据
9.03-9.06
and that that you know and it’s a non-volatile storage
但你知道，它是一个非易失性存储
9.06-9.09
I have to go get the four kilobyte page that it’s stored in
我不得不去获取该数据所存放的那个大小为4kb的页面
9.09-9.11
and then go pick out the just the piece that I want

然后再从中获取我想要的那部分数据
9.11-9.18 ****************** ！！！！！！！！！！！！！！！！！
another aspect that is that the these systems also particularly usually have faster sequential access
另一方面是这些系统通常也具有更快的循序访问
9.18-9.25
meaning a bunch of contiguous blocks in the storage device I can do that very more efficiently than just reading random locations
这意味着，在非易失性存储设备中，比起随机读取不同位置上的内容，我可以更有效率地去读取一段连续的块中的内容
9.25-9.30
the easiest way to visualize this is just think about like spinning this hard drive
思考这点的最简单的方式就是去想象一下硬盘旋转的情况
9.30-9.30
right

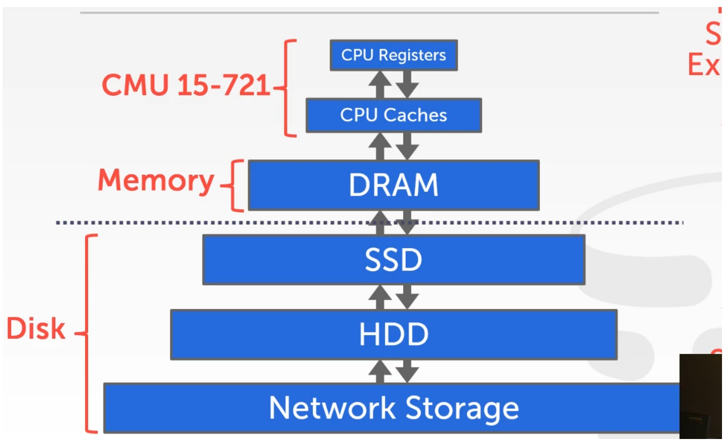
9.30-9.34
most laptops or every laptop pretty much doesn't come with a spinning disk hard drive
大部分笔记本或者说每台笔记本都没有机械硬盘
9.34-9.40
but basically the way it works is that you have this arm that's physically moving on the platter
但基本上它的工作方式是让磁头臂在磁盘上进行移动来访问数据
9.40-9.42 *****
like a turntable and combined
就像转盘那样
9.42-9.46
single time you got to jump to new location you have to pick the arm up and move it to another location
当你每次要去访问一个新的位置的时候，你就必须移动磁头臂到那个位置
9.46-9.49
and that's a physical movement and it's very expensive and slow
这是一种物理移动，它的代价很高，并且速度很慢
9.49-9.50
SSDs don't have this problem
SSD就没有这种问题
9.50-9.52
because it's solid-state
因为它是固态的
9.52-9.55
but there's other issues
但它有其他方面的问题
9.55-9.59 ******************
so in these storage devices we want to try to maximize the amount of data that we can read that's sequential
因此，在这些存储设备中，我们希望做到最大化地按顺序读取的数据量
9.59-9.59
right

9.59–10.02
and these ones we in the volatile storage we don't care as much
在易失性存储中，我们就不用太过于担心这些了
10.02–10.05
so for the purposes of this course
就这门课的目的而言


10.05–10.08
we're just going to say that anything that's in DRAM we're just gonna call this memory
我们将DRAM中的任何东西都称之为内存
10.08–10.13
like that's what we mostly care about how we actually put things into memory
我们最关心的是我们实际该如何将东西放入内存


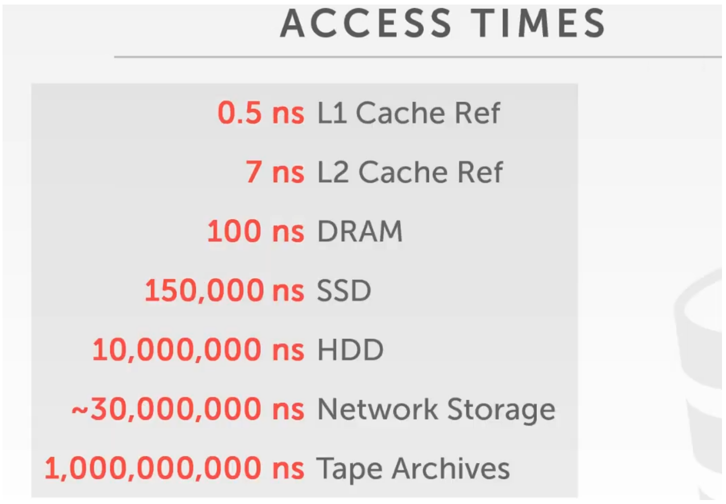
10.13–10.17
and then anything below this line here we're just gonna say this is disk
接着，分割线以下的所有东西，我们都称它为磁盘
10.17–10.24
and for all most of the algorithms and most of the methods were going to talk about in this course we don't care whether it's which one of these it is
对于在这门课中我们要谈论的大部分算法或者方法而言，我们并不在意它们是用于易失性存储还是非易失性存储
10.24–10.29
it's not entirely true when we talk about joins, sequential access matters a lot
~~当我们讨论join和顺序访问时，这点并不完全正确~~
这点在我们讨论join时并不完全准确，按顺序访问非常重要
10.29–10.31
but we'll come to that later
但我们会在之后讨论

10.31–10.34
and so I don't think the textbook talks about network storage
我并不觉得教科书里提到了网络存储这块内容
10.34–10.38
they usually they always when you see these kind of hierarchies always have tape drives at the bottom
通常当你看到这些存储层次时，你会在底部看到磁带机
10.38–10.40
but nobody runs databases on those anymore
但没有人会再往这玩意上面去运行数据库了
10.40–10.42
there's a relief like disaster recovery

~~这是灾难恢复之类的救济~~
可用来做像容灾恢复的事情
10.42–10.45
network storage would be like EBS or s3 on Amazon
此处的网络存储，你可以当做是Amazon的EBS或者s3之类的东西


10.45–10.52
so the reason why we’re not gonna talk about these things at the top is that where you focus on these things in the Advanced Course in the spring
我们之所以不讨论分割线以上这部分内容的原因，是因为这些是你们在春季的高级课程中所要学习的重要内容
10.52–10.57
and for our purposes in this class this course in the semester this is so slow anyway
我们这学期这门课的目的在于学习磁盘存储这块内容，当然这块速度比较慢
10.57–11.03
 that who cares how fast we can be putting things in CPU registers
如果介意速度的话，我们可以将数据放入CPU寄存器中
11.03–11.05
in the advanced class we assume the database is always here in DRAM
在高级课程中，我们假定数据库始终是运行在DRAM中的
11.05–11.07
and therefore these things actually matter
因此，这些东西实际上很重要
11.07–11.11
but for this entire semester we won’t really talk about worrying about things sitting in CPU caches
但在这整个学期，我们不会去讨论关于将数据存入CPU缓存相关的东西
11.11–11.13
 because it doesn’t matter cuz going to disk so slow now
因为目前我们并不介意将数据存入速度缓慢的磁盘中
11.13–11.23
 I always talk about this every year this is actually a new class of storage devices that sort of straddles the line called non-volatile memory
我在每年都会说，实际上在在这条分割线处还有新的一类存储设备，它被称为非易失性内存
11.23–11.25
who here has heard of this before
在座的同学之前有听过这个吗?
11.25–11.28
 who here has heard Intel Optane memory
有谁之前听说过Intel的傲腾内存?

11.28-11.30
 one two three all right
有3个同学
11.30-11.34
yeah so that’s so Intel is actually the first manufacturer the hash tag put released
实际上，Intel是第一个发布这种内存的生产商
11.34-11.37
 this everybody’s been working on this for like 15 20 years
其实，对于这种技术已经有了15-20年左右的研究了
11.37-11.40
 Intel actually put the first one is put out the first devices
实际上，Intel是第一个发布这种设备的公司
11.40-11.45
so it’s like DRAM where it sits in the DIMM slot and yet it’s byte addressable
它就像DRAM那样，可以插在DIMM槽内，但它具有字节可寻址的能力
11.45-11.46
but it’s like an SSD
但它也像SSD
11.46-11.50
meaning if you pull the power on the machine it persists all your data
这意味着，如果你断开机器的电源，它能够持久保存你的所有数据
11.50-11.51
alright so this is super cool
So，这非常酷！
11.51-11.54
this is like the future of what computer is gonna be look like
这就是计算机存储方面未来的样子
11.54-11.58
and eventually we’ll have to rewrite this class to take this an account
最终，我们会将这个存储层次重写为这个样子
11.58-12.01
but we’re not there yet it’s not widely available
但目前来说，这种存储设备并没有被广泛应用
12.01-12.07
now again this is something I’ve been working on it for a while, so this is a book that I wrote with my first phd student
这本书我已经写了有一段时间了，它是我和我的一位Phd学生一起完成的
12.07-12.10
 we it was basically his dissertation they put into book form
我们将他的论文写成书这样形式进行发布
12.10-12.12
 but I think this is the future
但我认为这种技术就是未来
12.12-12.14
we’re just not there yet it’s not widely available
但现在并未广泛使用
12.14-12.15
 but you can’t get on Amazon yet
目前你无法在Amazon上买到这本书

12.15-12.20 （这段话有beeeeee发生，之后再看）
but eventually a lot of talk about what could go away if you have non-volatile memory
有很多关于非易失性存储器可能会消失的讨论
12.20-12.21
all right

ACCESS TIMES

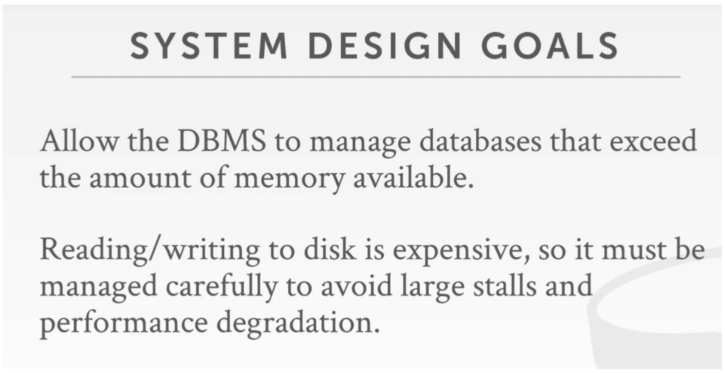
0.5 ns	L1 Cache Ref
7 ns	L2 Cache Ref
100 ns	DRAM
150,000 ns	SSD
10,000,000 ns	HDD
~30,000,000 ns	Network Storage
1,000,000,000 ns	Tape Archives

12.21-12.23
so let's talk about how slow these things are
So，让我们来讨论下这些东西到底有多慢吧
12.23-12.31
so there's different tables on the internet that have different numbers，but these approximately the same you know roughly in the same ballpark
这是网上的一张关于各种存储设备的访问耗时表，不同的表可能在数字上有所差异，但大体相同
12.31-12.35
the thing that matters most is the orders of magnitude that's different between these storage devices
最重要的是这些存储设备之间的访问耗时的不同
12.35-12.39
so let's say I need to read a 64 bits from different storage device
假设，我现在需要从这些不同的存储设备中读取一个64 bit大小的数据
12.39-12.44
if I'm in l1 cache, then it's like half a nanosecond
如果我是在L1缓存中进行读取，那大概需要0.5纳秒即可
12.44-12.47
if it's in l2, it's seven nanoseconds
如果是在L2缓存中，那就需要7纳秒
12.47-12.57
and so forth to get to really long just long delays
如图所示，从上到下的读取时间会越来越长
12.57-13.00
so again this is why we spent all this time in this class worrying how we can try to minimize the impact of reading data from disk
这就是为什么我们会将这门课全部的时间花在如何最小化从磁盘读取数据的影响了
13.00-13.00
right

13.00-13.05

because there’s a pretty big difference between 100 nanoseconds and 150,000 nanoseconds
因为100纳秒和150000纳秒给人带来的差异是完全不同的
13.05-13.10
and so for every single query if we always have to go out the disk，then we’re screwed
如果我们每次查询都要从磁盘中获取结果，那就很糟糕
13.10-10.13
 right the thing is gonna be essentially grind to a halt and we’re not gonna get any work done
~~因为我们就得停下来等查询完成，不然没法做任何事情~~
没错，事情基本上会停下来，这期间我们不会做任何工作
13.13-13.18
and I realized putting this in terms of nanoseconds is hard for us to as humans to wrap our heads around
对于我们人类而言，我们很难感受到以纳秒为单位的东西
13.18-13.21
 so if you just replace nanosecond with seconds
但如果你将纳秒换成秒
13.21-13.24
 then you start to realize how long these numbers actually are
那么你就会意识到，这些数字所代表的时间的意义了
13.24-13.31
 right so the way to think about this is like in another metaphor you like to use from Jim Gray a famous database researcher
我们可以像著名的数据库研究员Jim Gray那样去思考这种问题
13.31-13.36
is that say that I want to read a book I want to read a page in a book
假设，我想去读书中的某一页
13.36-13.38
so if it’s in l1
如果它是在L1缓存中的
13.38-13.41
then it’s just like reading the book right in front of me on this table
这就像是直接阅读放在我面前桌子上的书那样
13.41-13.42
 if it’s in l2
如果它是放在L2缓存中的话
13.42-13.44
then maybe it’s going across the room to read it
那就像是你需要去隔壁房间拿到这本书，然后阅读它
13.44-13.46
if it’s in DRAM
如果它是放在DRAM中的话
13.46-13.47
that I got to walk to the library
那就是我得去图书馆读这本书
13.47-13.51
and then now you start to these you know larger and larger orders of magnitude
现在你理解了这些，随着数量级越来越大

13.51–13.53
and if you have to read from a tape drive
如果你是从磁带机中读取数据
13.53–13.56
 it's like flying the Pluto to read a page in a book
那么你读完书中一页的时间可能就像是飞到冥王星那么久了
13.56–13.57
 it just takes forever
~~这就可能需要一辈子才能读完这一页子~~
这可能永远读不完了
13.57–14.00
so again this is why people don't want to store data on this
这就是人们为什么不想将数据存在这上面的原因了
14.00–14.01
but note is that they had to
但注意这是他们不得不做的事情
14.01–14.02
 all right



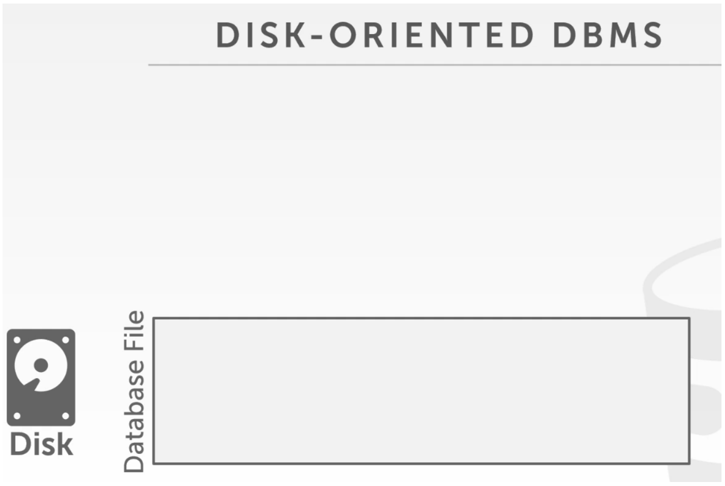
SYSTEM DESIGN GOALS

Allow the DBMS to manage databases that exceed the amount of memory available.

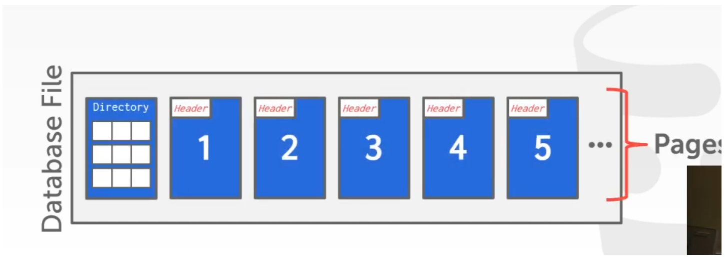
Reading/writing to disk is expensive, so it must be managed carefully to avoid large stalls and performance degradation.

14.02–14.14
so the goal of what we're trying to do in our databse system is that we want to provide the illusion to the application, that we have enough memory to store their entire database in memory
我们所要试着在我们的数据库系统中达成的目标是给应用程序提供一种错觉，即我们能提供足够的内存将整个数据库存入内存中
14.14–14.18
essentially like you know there's a fine amount of memory on our machine
正如你知道，在我们的机器上有一定量的内存
14.18–14.23 ******
and we want to store a database that exceeds the amount of memory that's available to us
我们想要存储的数据库超出了我们可用的内存量
14.23–14.28
but without having to grind to a halt every single time we read something or write something
但是我们不必每次停下来去读取或者写入某些东西
14.28–14.28
right

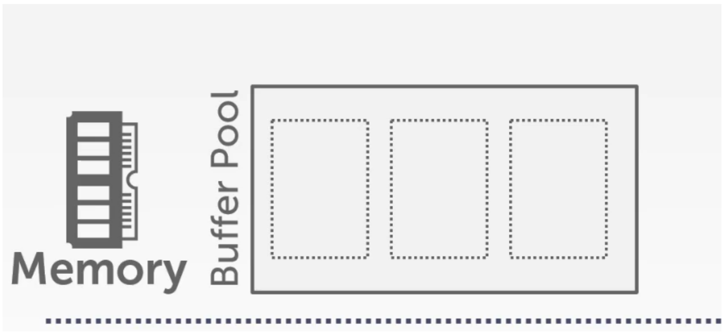
14.28–14.31
so again that's what focus in this course is

这就是这门课的重点
14.31–14.40
and the next three lectures is really about how can we be careful on any single time we’ve got to read something from disk or run a query that we minimize that impact
接下来三节课是关于我们如何谨慎的最小化每次从磁盘读取内容或运行查询时所带来的影响
14.40–14.43
and we’re gonna do a bunch of different tricks to mitigate this problem
我们通过一系列不同的技巧来减轻这种问题带来的影响
14.43–14.46
by allowing different threads or different queries are run at the same time
例如，允许在同一时间运行不同的线程或者不同的查询
14.46–14.47
by caching things
或者缓存某些东西
14.47–14.49
by precomputing some data
或者提前计算出某些数据
14.49–14.58 ********
there’s a whole bunch of slew of tricks that we’re gonna have in our database system that we have in essentially around avoiding this long problem
在我们的数据库系统中，我们将有很多技巧可以避免这种缓慢问题


14.58–15.07
so let’s look at a high level diagram of what a database system looks like from the perspective that we care about at this point in the semester
So，在本学期中，我们以我们所关注的角度来看一个数据库系统的高级示意图
15.07–15.13
and then we’ll see how the rest of lecture how actually to fill in and design these things
接着，我们会在这节课的剩下时间内看下该如何填充这张图，以及设计这些东西
15.13–15.16
so again at the lowest layer we have the disk
在最底层，我们有磁盘
15.16–15.20
and we have our database file or files it doesn’t matter
接着，我们还有数据库文件

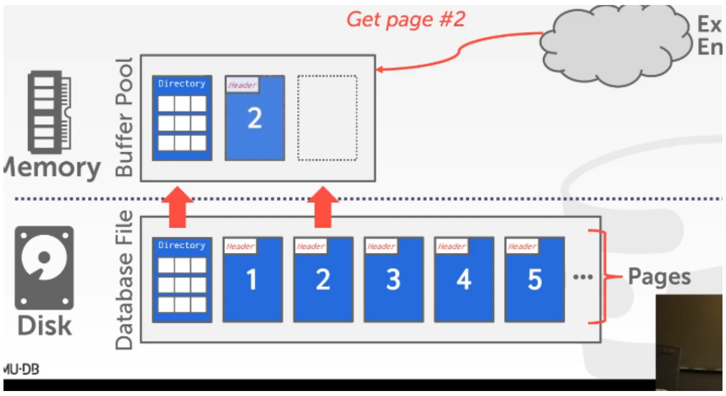


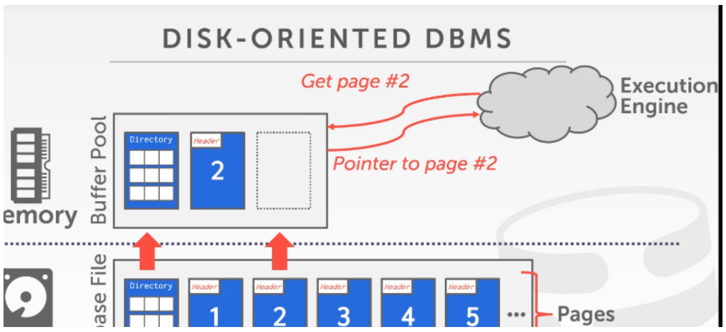
15.20-15.24
and then we're going to represent these through different blocks or pages
然后，我们通过不同的块或者页面来表示这些东西
15.24-15.28
 right a page is the canonical term you use to describe this
页面是你用来描述这个的规范术语
15.28-15.30
 sometimes ~~slip and~~ say block
有时会说成块
15.30-15.32
 but at a high level they mean the same thing
但从高级层面来讲，它们是一回事

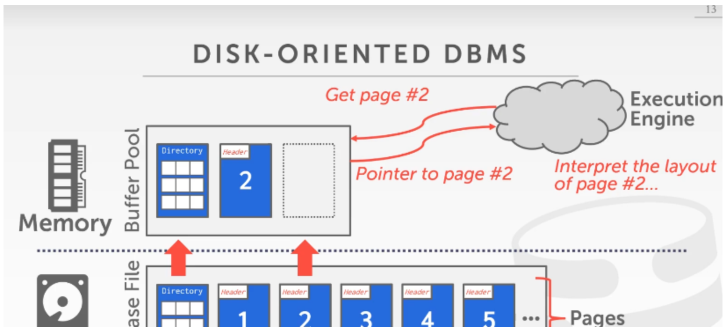


15.32-15.35
so now a memory we're gonna have what was called a buffer pool
接着，在内存中我们有一个成为buffer 缓冲池的东西
15.35-15.38
and this will focus on this and the lecture next week
我们会在下节课着重讲 它
15.38-15.43
and this is what you'll be implementing in the in the first assignment the first project
这也是你要在第一个项目中所要实现的东西
15.43-15.47
so there's some higher-level layer in the system and execution engine a query engine we don't care what it is
系统中一些更高层的比如执行引擎，查询引擎，我们不在乎它是什么
15.47-15.50
but it's gonna make requests to our buffer pool
但它会去向我们的buffer 缓存池进行请求

15.50-15.51
and say hey I want to read page 2
并表示，我想要读取page 2的内容
15.51-15.53
 page 2 is not in memory
但page 2并不在内存中

15.53–15.56
so we got to go look in the page directory on disk
因此，我们要去磁盘上的page目录中进行查找
15.56–15.57
and say here's the list of the pages that I have
在page目录中，我们有page列表


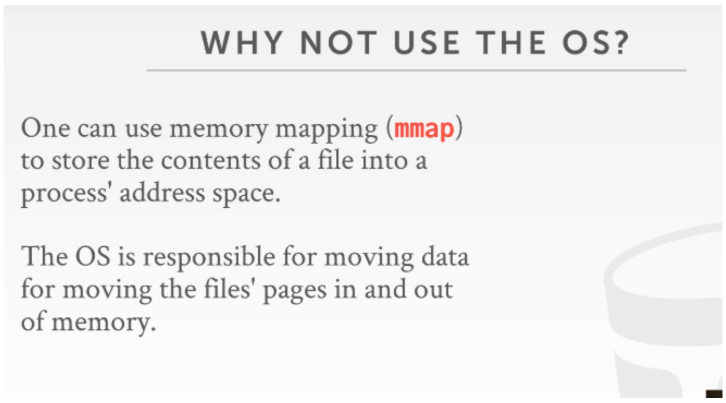
15.57–15.58
 here's where to find them
我们从这里面找到它们
15.58–16.01
so I'd now I can go find where page 2 is I bring it into memory
现在，我们可以从里面找到page 2，并将它放入内存

DISK-ORIENTED DBMS

16.01–16.05
and now I hand off to my execution engine
现在，我将它提交给我的执行引擎
16.05–16.007
here's the pointer to page 2 in memory
这里有一个指向内存中page 2的指针
16.07–16.10
and then it can interpret it do whatever at once we don't care
然后它可以解释并立即执行任何我们无须关心的操作
16.10–16.14
alright so this is what we're focusing on here as for the next three lectures
这就是我们在接下来三节课中要讲的内容

DISK-ORIENTED DBMS

16.14–16.16 ************
 that was how we actually build this part here
实际上这就是我们在这里对此部分进行构建的方式
16.14–16.17
 all right

16.17–16.23
so today and next week we’ll do discuss what the data files on disk
因此，今天以及下周，我们会去讨论磁盘上的数据文件
16.23–16.24
 next week will be the buffer pool
下周的内容则是buffer 缓冲池
16.24–16.28
 and then later and so forth we’ll talk about how we actually represent the directory
之后我们会去讨论我们该如何表示目录
16.28–16.29
 ok

16.29–16.32
so what does this look like
So，这看起来像什么呢?
16.32–16.38
you take an operating system course what does it sound like
你们上过操作系统相关的课程，那它看起来像什么呢?
16.38–16.41
 I’m trying to make it appear that I have more memory than I actually do
我试着让它的内存看起来比我们实际有的内存更加多
16.41–16.43
virtual memory exactly
没错，虚拟内存
16.43–16.46
so now I may be thinking all right I’ve taken OS course here
我假定你们在座的人都上过操作系统这门课
16.46–16.49
why do I want to have my databases manage memory like this
为什么我想让我的数据库像这样管理内存呢?
16.49–16.50
 it seems like a big waste of time
这似乎很浪费时间
16.50–16.51
and they OS can already do this right
操作系统已经能够这么做了
16.51–16.56
well it’s not a good idea
Well，这并不是一个好的想法

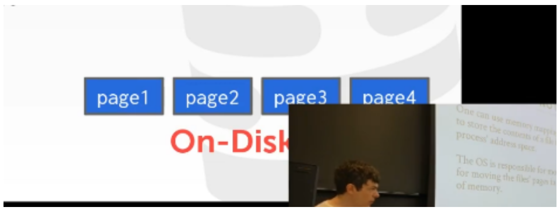

WHY NOT USE THE OS?

One can use memory mapping (mmap) to store the contents of a file into a process' address space.

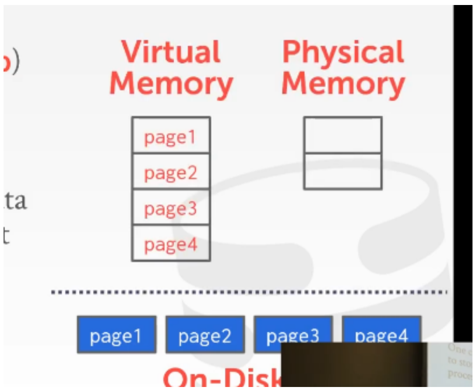
The OS is responsible for moving data for moving the files' pages in and out of memory.

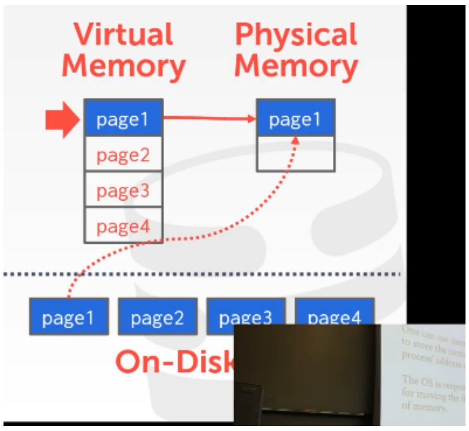
16.56–16.57
here’s what
这里是什么呢?
16.57–17.05
so in an operating system parlance we would call this memory map files or there’s a syscall called mmap in POSIX
在操作系统术语中，我们将它称为内存映射文件，或者在POSIX中，我们称它为mmap
17.05–17.12
and essentially what this does is it takes a file on disk and you tell the operating system map the files pages into the address space for my process
本质上来讲，这是从磁盘上获取文件，然后你告诉操作系统将文件页面映射到我的进程的地址空间中
17.12–17.17
and now I can read and write to those memory locations
现在，我就可以对这些内存地址进行读写
17.17–17.20
and it’s not memory the OS brings it in
这并不是操作系统所引入的内存
17.20–17.21
I can write to it
我可以对它进行写入

17.21–17.27
and then eventually if I can tell OS to write it out for me I can do an N Sync and write it up back out the disk
~~接着，最后我可以让操作系统对其进行写入，并将其写回磁盘~~
接着，最终我可以让操作系统将之写出，通过执行一个 Sync并将其写回到磁盘上
17.27–17.32
so we’re essentially giving up control of the movement of memory back and forth of data back and forth we disk of memory
So本质上来讲，我们放弃了数据在上内存以及硬盘上来回移动的控制权
17.32–17.34
and letting the operating system manage this for us
而是让操作系统来帮我们管理这些

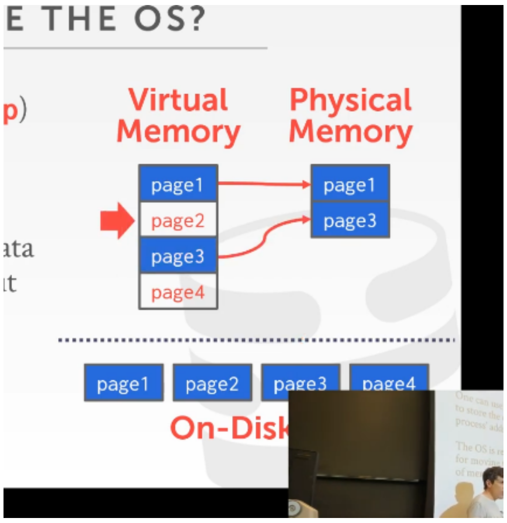


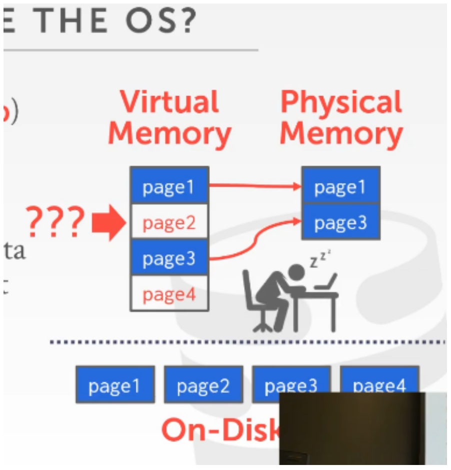
17.34–17.40

right so again at a high level it looks like this
从高级层面来讲，它看起来像这样
17.40–17.42
we have a bunch of pages on disk file
在磁盘文件中，我们有一些page


17.42–17.46
and then in memory the OS has its virtual memory page table and we have physical memory
接着，在内存中，操作系统有它自己的虚拟内存页面表以及物理内存
17.46–17.47
so what happens?
接下来会发生什么呢?
17.47–17.49
is the application says hey I want to read page 1
假设应用表示要去读取page 1
17.49–17.51
it looks in the virtual memory
它就会去虚拟内存中进行查找
17.51–17.53
we get it we get a page fault
我们得到了个Page Fault（缺页异常）
17.53–17.55
and say this thing’s not backed by physical memory
并表示这些东西在物理内存中并没有备份
17.55–17.56
it’s not it’s still out on disk
它仍然在磁盘里面
17.56–17.59
we go fetch it back it into a physical memory page
我们要将它取出来，并将它放到物理内存页面中去


17.59–18.03
and then update our page table to now point to that memory location
然后，更新我们的page 表让它去指向该内存地址

18.03–18.05
 so if I come along and I want to read page 3
So，如果我继续走下去，想去读取page 3
18.05–18.07
I go through the same process
我会经历相同的过程
18.07–18.08
 I fetch it into memory
即我将它放入内存
18.08–18.11
and then the application can do whatever want
接着，应用程序就可以做它想做的事情了


18.11–18.14
but now let’s say that I read page 2
但现在假设我要去读取page 2
18.14–18.15
 what’s the problem
那现在的问题是什么
18.15–18.22
there’s no free physical memory page to put the particulars put this page in
现在没有空闲的物理内存来存放这个page


18.22–18.26

 so I need to make a decision of which of these pages to remove
因此，我需要判断这些页面中我该移除哪个
18.26–18.33
 and while I’m doing this, eventually have to stall the database system install my thread that requested this page
当我做这件事的时候，我不得不让数据库系统停止请求该page 的线程
18.33–18.40
because now disk scheduler for the operating system is gonna go out to pay the disk fetch it and bring into memory
因为，现在操作系统的磁盘调度程序要从磁盘中拿到这个数据，并将它放入内存中
18.40–18.47
~~so you there’s tricks~~ there’s ways to figure out from the application perspective and might have to read something that’s not in memory
我们可以从应用程序的角度来弄清楚，数据库可能要去读取某些并没有放在内存中的东西
18.47–18.49
 so maybe I could hand it off to another thread
因此，我可以将它交给另一条线程去做
18.49–18.50
 so it stalls and not me
so 这条线程会停下来，我不会
18.50–18.52
 because I always want to try to keep doing useful work
因为我始终想去试着做有用功
18.52–18.56
because I want to mitigate the stalls，when I have to go out the disk
因为我想减少停顿的时间，所以我不得从读取磁盘的工作中抽身出去
18.56–18.57
right

18.57–19.01
but essentially the operation doesn’t know exactly what the hell we’re doing
但实质上来讲，操作系统并不知道我们要做什么
19.01–19.02
doesn’t know anything about what the database system is doing
它并不知道数据库系统正在做什么
19.02–19.05
it just sees a bunch of reads and writes to pages
它只是看到了对这些页面所进行的一系列读写操作