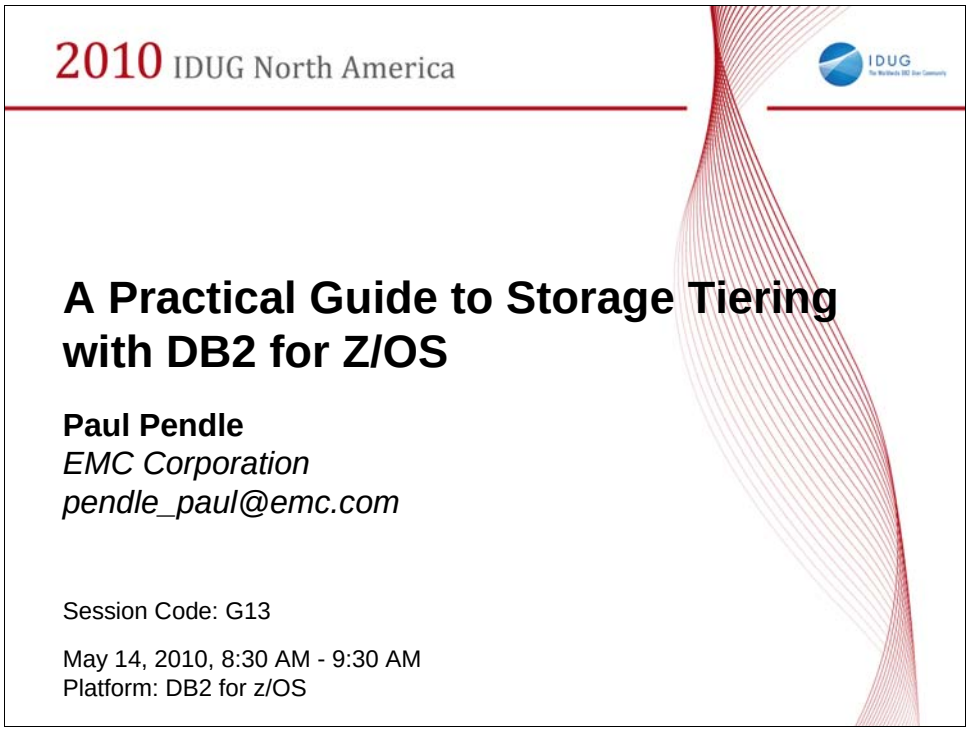


Nowadays, DBAs are presented with multiple different storage options upon which to deploy DB2 systems. The options offer varying performance, availability and economics. The optimal selection for a given DB2 for z/OS system may not be obvious. Moreover, as application usage changes over time, moving to different choice of storage can be challenging.

This presentation discusses practical ways to implement a tiered storage architecture with DB2 for z/OS and presents a view of how this may be handled in the future.

Agenda

- Why should you care about disk?
  - … and how we got here …
- Disk technologies
  - A brief introduction to Flash Drives
- Defining Storage Tiers
- Static Tiering
- Integration with SMS/HSM
- Dynamic Tiering
- Thin provisioning
- Sub-Volume tiering
- Futures …

This presentation leads to the following goals:

•Understanding the tiered storage model

•Understanding skew and persistence in data access

•Practical methods of full volume tiering with DB2 for z/OS

•Understanding partial volume tiering - the real requirements

•A view of the future for the storage tiering model

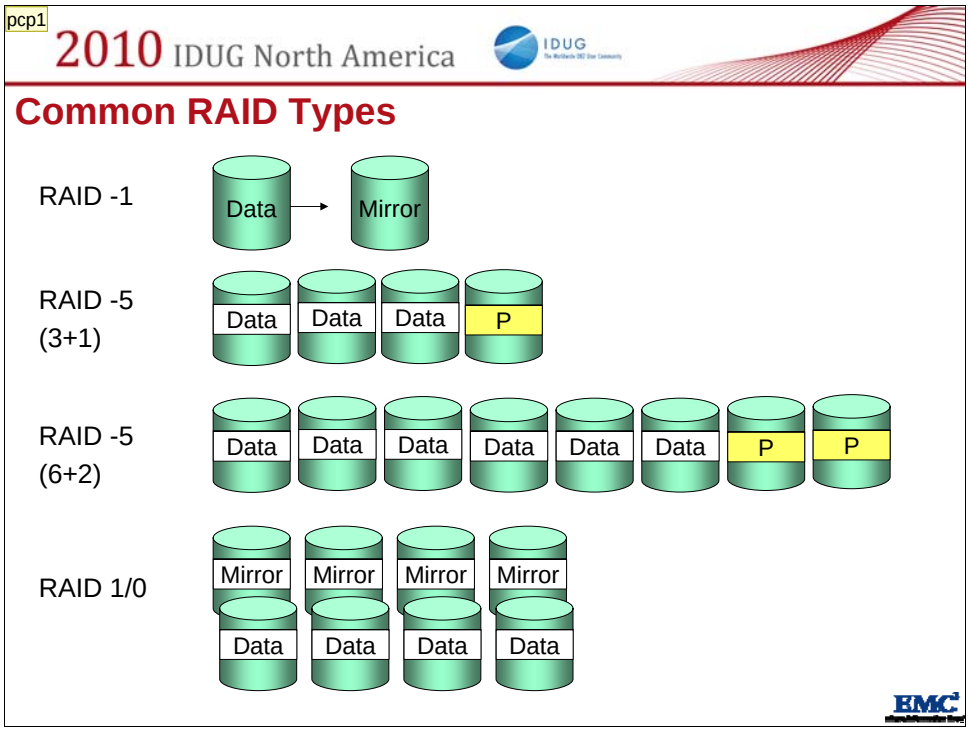
Why should you care?

You need to care about disk. CPUs are getting faster and faster. And when they peak they get more cores and greater parallelism. A zSeries processor can have as much as 1.5 TB of RAM. A total that was inconceivable 10 years ago. And the channels. They have steadily gotten faster and faster since BUS & TAG, through ESCON to FICON. And when the channels are not getting faster the improvements are in efficiciency like zHPF which is very helpful for a 4K channel saturated workload. But hard drives have plateau'd out. There has been no significant performance change in the last seven years. All we are seeing is that they are getting bigger and bigger.

Why Should You Care (contd.)

The spectrum of choices is broader than ever before:

- Different Capacities
  - From 73GB to 2TB
- Different RAID protections
  - RAID 1
  - RAID 1/0
  - RAID 5
  - RAID 6
- Different technologies
  - FC (Fibre Channel)
  - SATA (Serial Advanced Technology Attachment)
  - SSD (Solid State Disk)

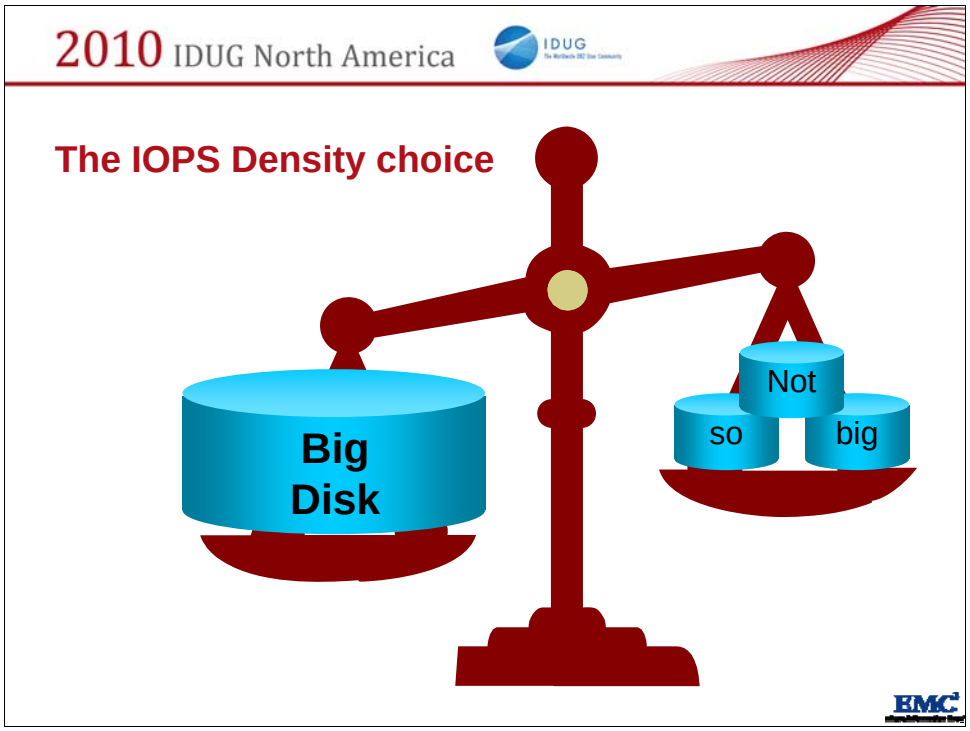


Common RAID Types

Different RAID types have different performance, economy, and availability characteristics. RAID 1 uses up the most of your raw disk capacity; 50% goes for data protection. For this you get the best performance for both reads and writes. RAID-5 3+1 (3 data disks and one parity) striped data with parity protection. Only 25% is used for protection. RAID 5 requires 4 I/Os for every. RAID 6 with double parity and requires 6 I/Os for every write and therefore does not perform well in a high-write environment.

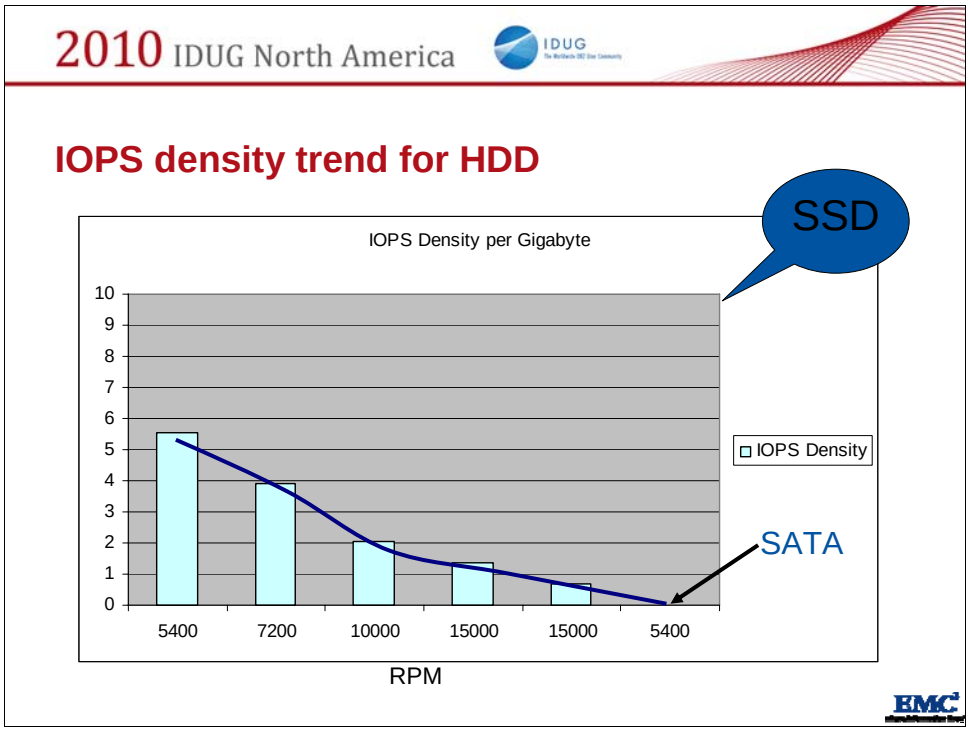
Slide 5

pcp1 ppendle, 4/13/2010



IOPS density

IOPS=**I**nput/Output **O**perations **P**er **S**econd. The reality is that the key metric IOPS density is missing here. In fact it rarely gets any publicity. Let me ask a question if you had the choice which would you take, 500 1GB hard drives or 1 500GB drive. The reality is that drive single thread operations and no matter how fast they are only having a single spindle servicing I/O is inferior to having lots of smaller slower drives. IOPS Density is the effective IOPS of a given drive divided by its capacity. The IOPS density of drives has been falling at a steady rate over the last several years.



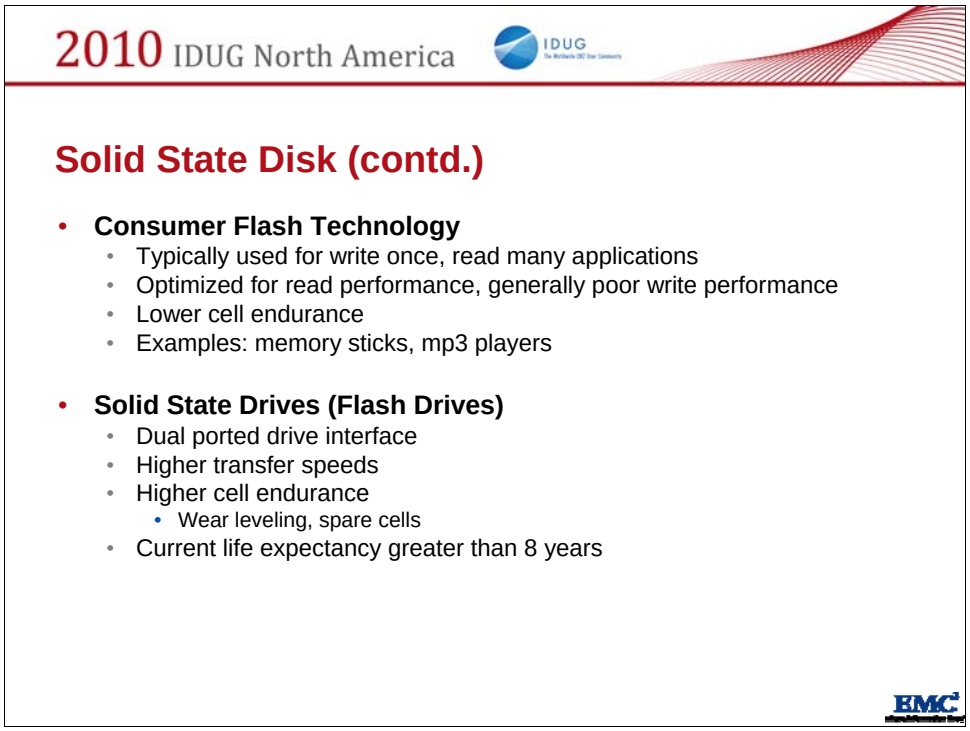
IOPS Density Trend

This chart shows how the IOPS density has fallen over the last 15 years or so. And finally, the last column on the chart for a 1TB sata drive, it is less than 1/10 an IOP per GB. This single factor alone is why disk systems have to be manage carefully today. To be sure storage controllers have gotten smarter, lots of cache, many processors, smart prefetch algorithms etc. But when the rubber hits the road the data must always land on disk.

Solid State Disk

- They are good for every kind of workload
- They are expensive
- They are not just a large thumb drive …

Solid state disks agenda



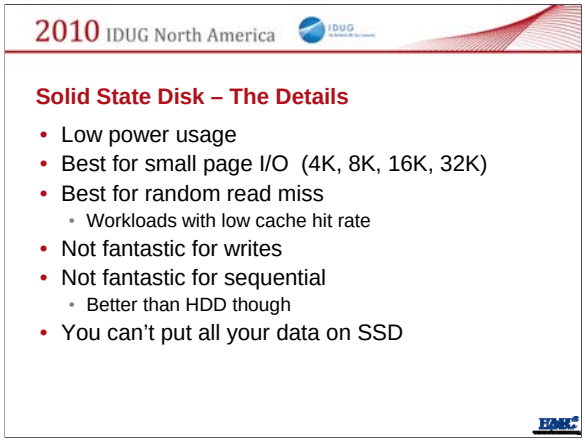
Solid state disk – details.

So the enterprise solid state disks are different from those you would find in a thumb drive. Your thumb drives are not build for tremendous amount of write activity. Eventually cells in the drive wear down if written to consistently. And the write performance is not great.

Enterprise Flash drives are based on NAND technology. The drives come with wear-leveling algorithms, redundant cell replacement (transparently). Their actual life expectancy is a around 8 years which is more than your typical disk drive. They are also dual-ported, have fast data transfer speeds and have on-board cache to assist writes.

Title

Month Year



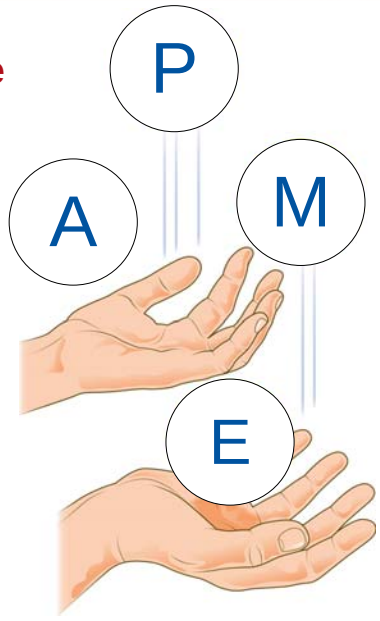
Solid State Disk – The skinny

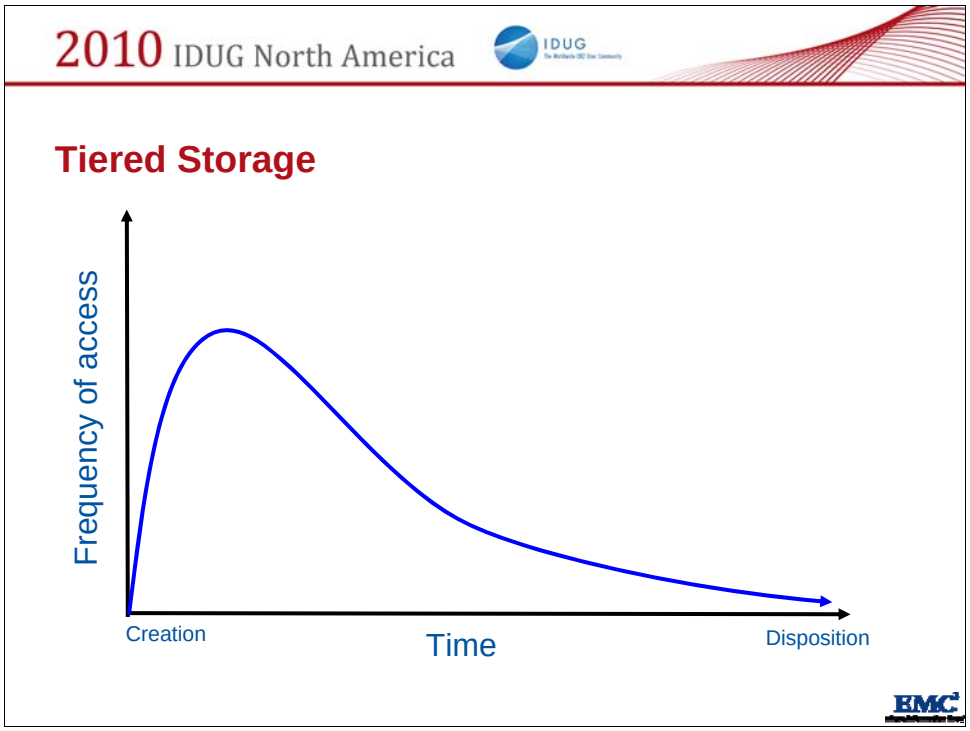
Since it is unlikely you will be placing all of your database on SSD you will need to pick and choose what goes on the SSD and what goes on other tiers of storage. The best fit is for DB2 tablespaces is those tables that are accessed randomly with small pages, 4K and 8K. Anything with heavy writes is already receiving memory type I/Os if you are using an enterprise storage array since they all come with DASD fast write, that is to say the write is placed into battery-backed up memory on the controller and the acknowledgement to the application is immediate. The cache slot is written to the disk asynchronously.

The Juggling Struggle

- Performance
- Availability
- Economics
- Manageability

"The right data in the right place at the right time"

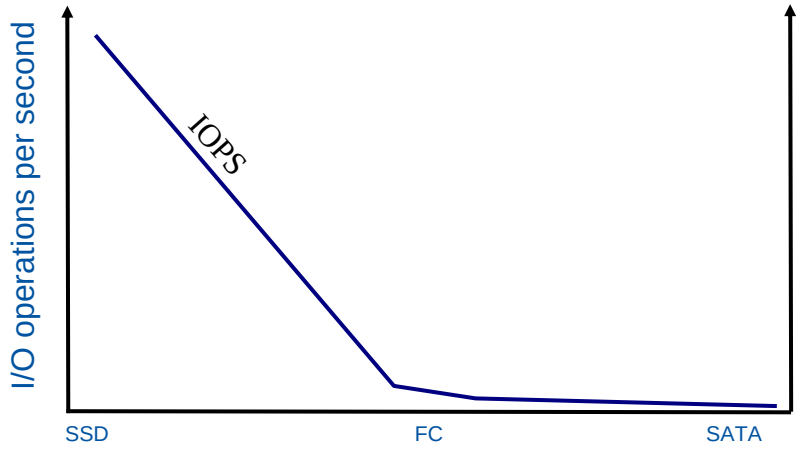


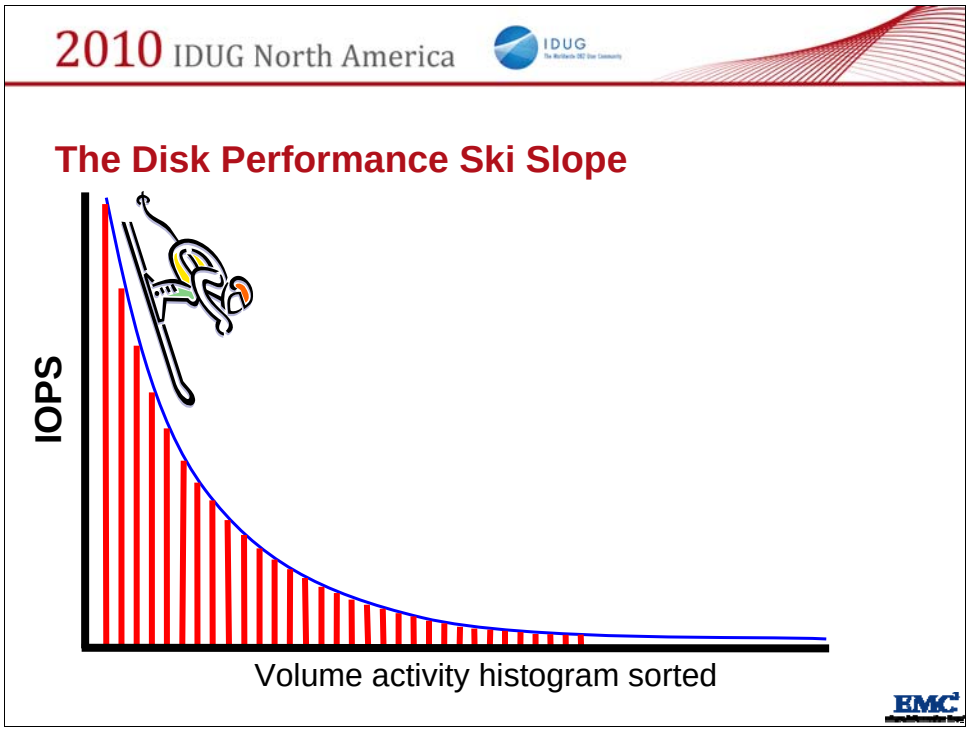


Tiered storage – the reality

The reality is that life is not so simple. Rarely is it when data's importance does not change. And even within a single table data changes over time. The graphic here depicts a typical piece of information from it's creation to it's disposition and the frequency with which it is accessed. Take your telephone bill for example. You might get a telephone bill this month and call the TELCO about it. You might even call them next month. But how often do you go back 3 months, or more? So over time your billing information is accessed less and less and then it is put to nearline storage or archived.

Tiered Storage





Typical performance profiles

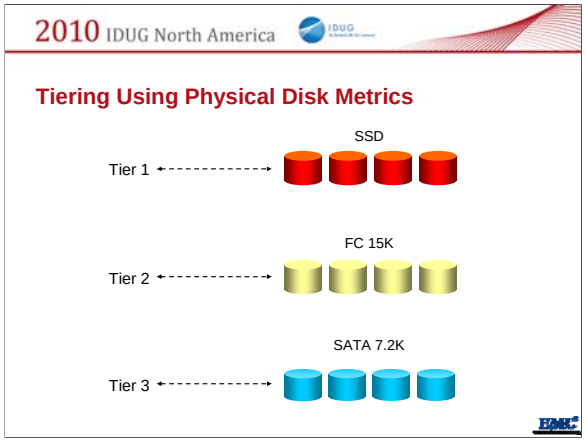
If you were to graph the IO activity of all volumes in your array and histogram them, and order them by most active to least active, I guarantee that you would end up with a ski slope. The downward slope of the curve is such you will see about 20 percent of the disks performing 80% of the workload. This is so typical it is not even funny. The gradient of the slope might change somewhat from account to account but the ski slope is always there.

The trick is how to balance the I/O across all the disks evenly. The only customer configuration that I have ever seen that was evenly balanced was at a customer site running IBM Transaction Processing Facility (TPF). Something within the TPF OS automatically causes all devices in the configuration to be equally loaded.

So if you are not running TPF – how do you balance your system?

Title

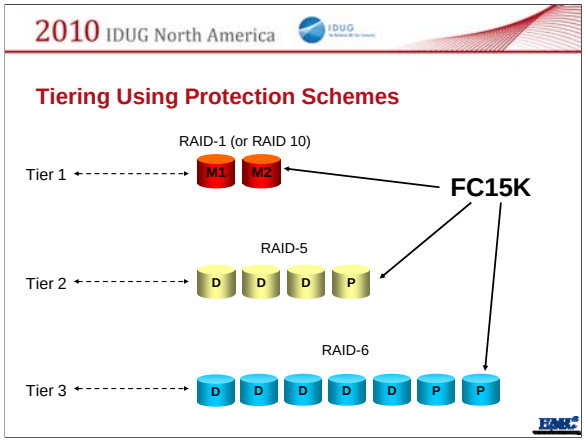
Month Year



15

Title

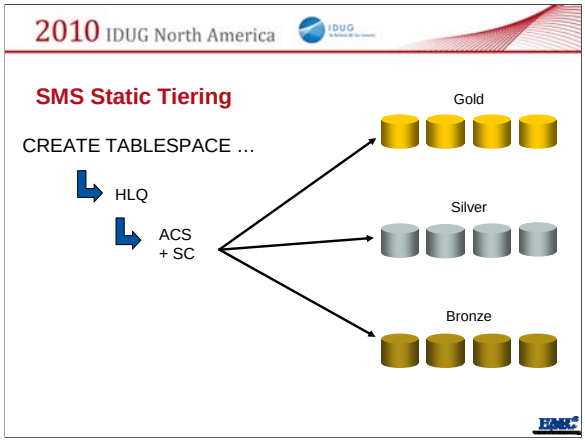
Month Year



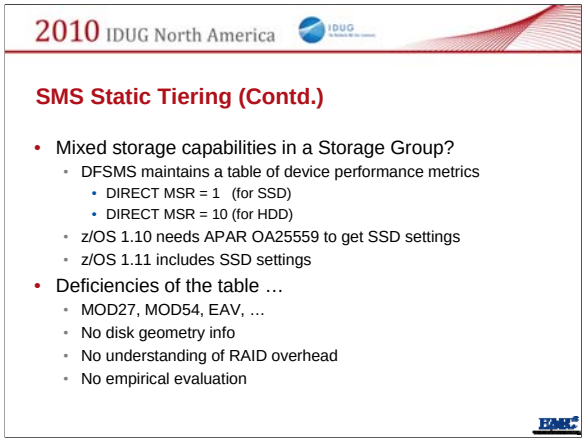
16

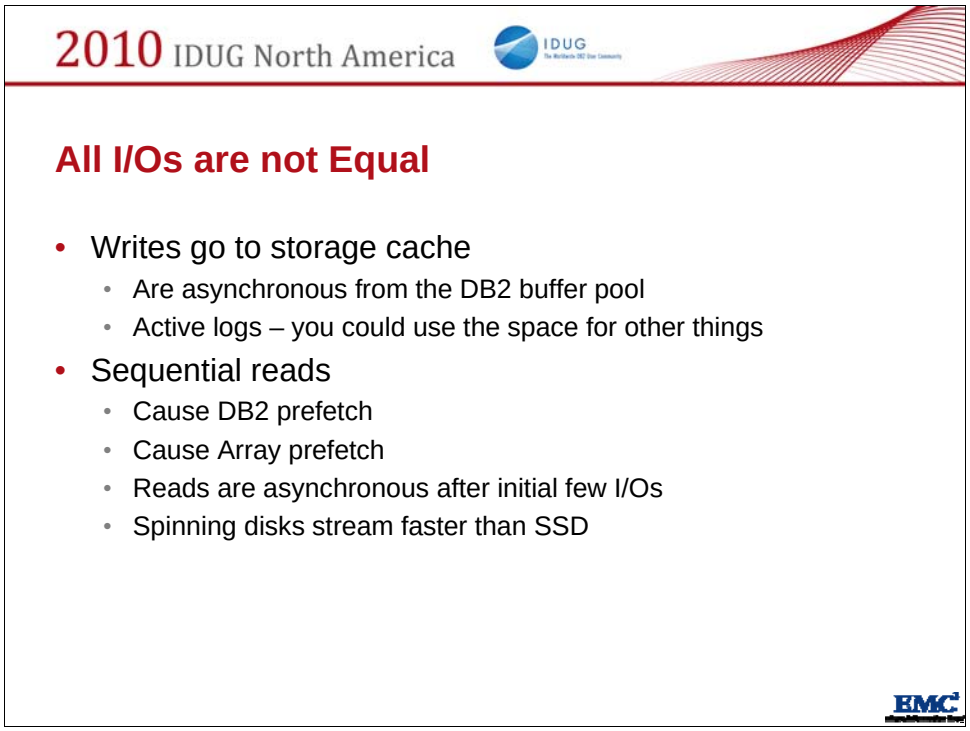
Title

Month Year

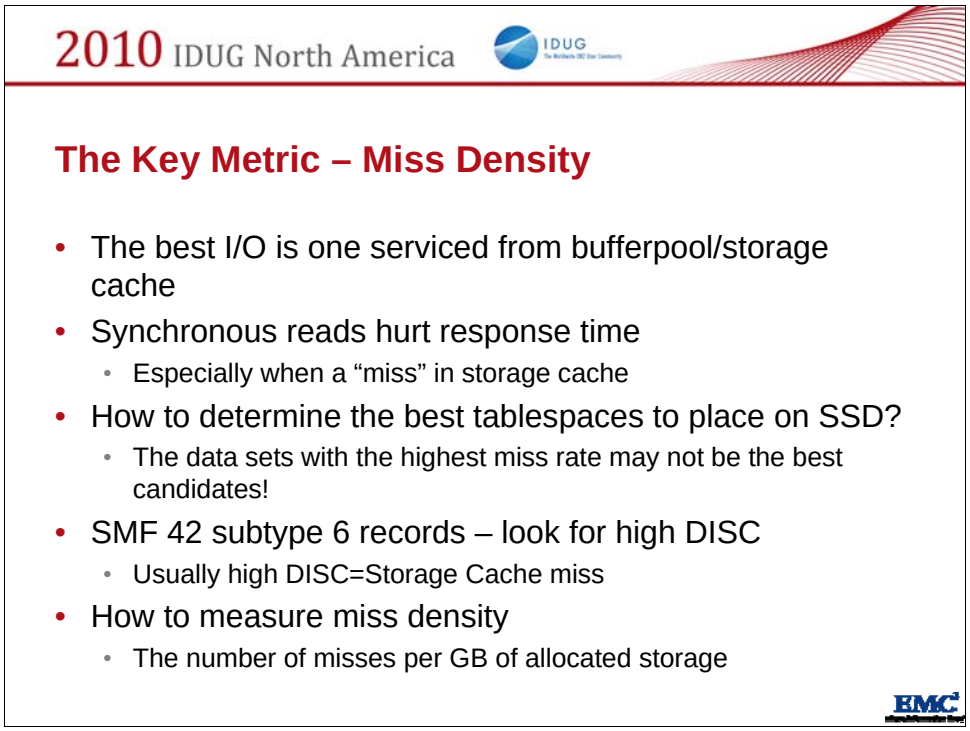


17


2010 IDUG North America
SMS Static Tiering (Contd.)
• Mixed storage capabilities in a Storage Group?
◦ DFSMS maintains a table of device performance metrics
▪ DIRECT MSR = 1 (for SSD)
▪ DIRECT MSR = 10 (for HDD)
◦ z/OS 1.10 needs APAR OA25559 to get SSD settings
◦ z/OS 1.11 includes SSD settings
• Deficiencies of the table …
◦ MOD27, MOD54, EAV, …
◦ No disk geometry info
◦ No understanding of RAID overhead
◦ No empirical evaluation

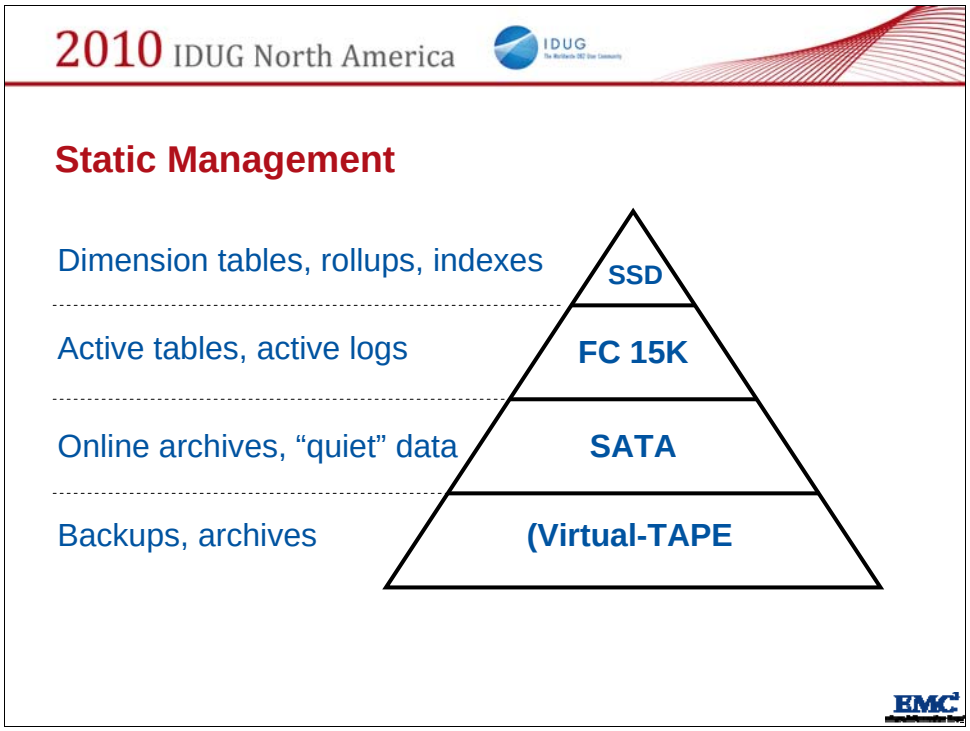


All I/Os are not Equal



Miss density

Miss density is the key metric that allows you to select which datasets are good candidates for flash drives. The ones with the highest miss density, that is to say the highest ratio of storage cache misses to gigabytes of dataset are the best candidates for Flash drives.



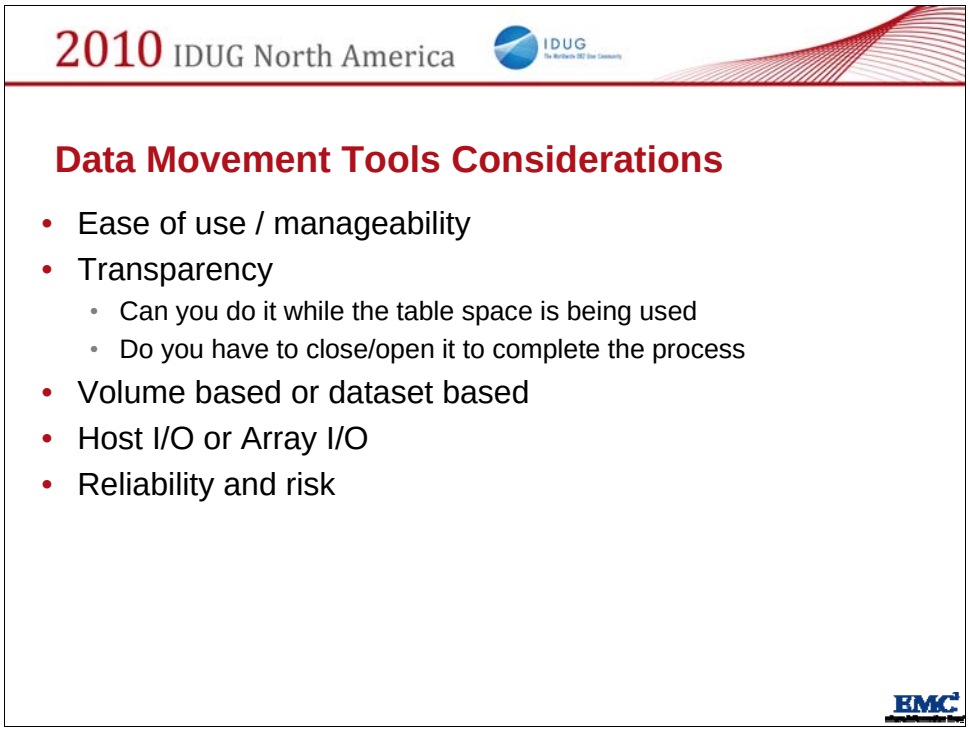
Tiered Storage static management

If you know your data very well and your access patterns exhibit persistently skewed access you might be able to apply a static tiering methodology in your environment. You will need to determine which tablespace and which data belongs in each tier and statically assign these database components to the tier. This of course requires a detailed knowledge of the underlying storage infrastructure and that you be on very good terms with your storage administrators.

Of course this diagram is somewhat oversimplified. There is a place in the hierarchy for NAS, for near-line storage, WORM storage etc.

Tiered storage – the reality

I can tell you that there are very few tools to help you get to a tiered storage configuration. They are single tools that can help you move data like those listed here, and some of them do not require down time. But they can be labor intensive and error prone. Plus they assume a prior knowledge of data access paths and these are not always predictable. zDMF is the z/OS dataset migration

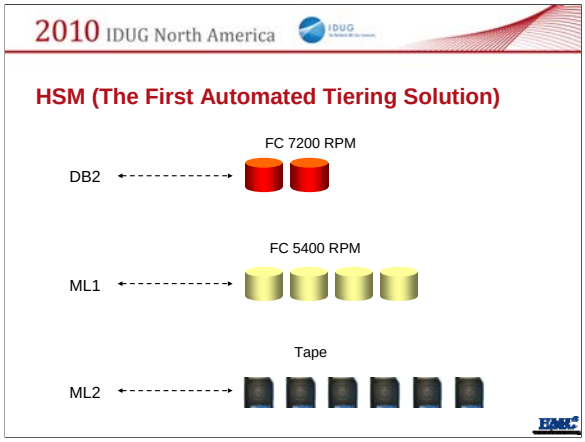


Tiered storage – the reality

I can tell you that there are very few tools to help you get to a tiered storage configuration. They are single tools that can help you move data like those listed here, and some of them do not require down time. But they can be labor intensive and error prone. Plus they assume a prior knowledge of data access paths and these are not always predictable. zDMF is the z/OS dataset migration

Title

Month Year

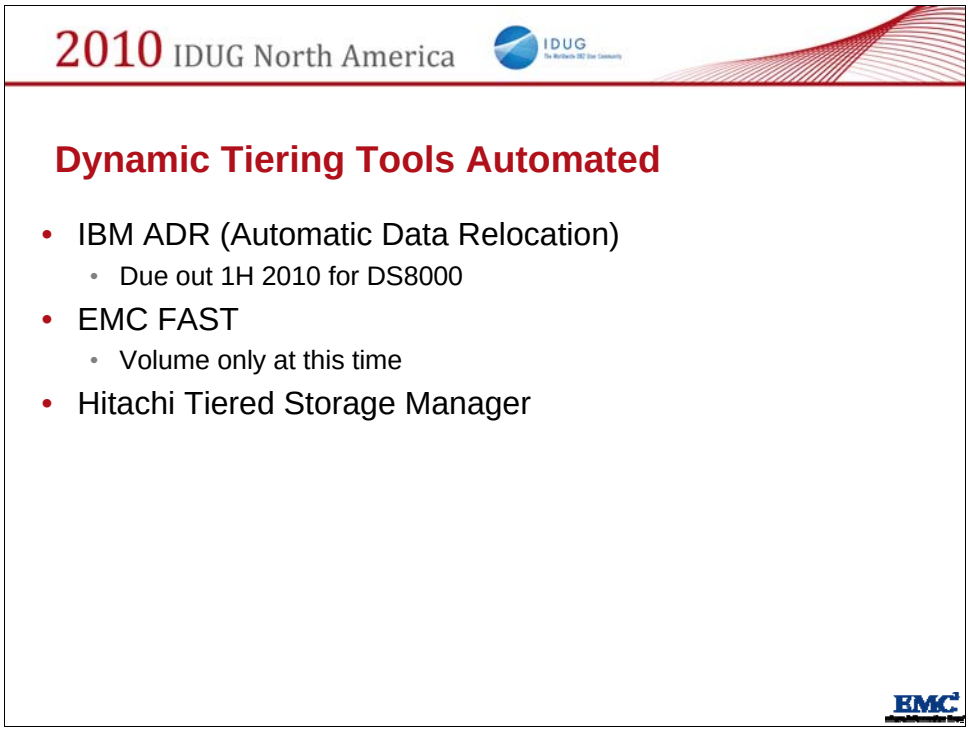


HSM – the original tiering solution

HSM migration might be the "original" storage tiering solution. It allowed for data that was not accessed to be moved from production devices to a staging device and compressed to save space. The archive area on disk is called ML1. Typically ML1 devices/storage is less expensive/less performing than the primary devices, resulting in a cost savings to the customer. When the data is accessed that is on ML1 it goes through a process of "recall" and it is decompressed and restored to the source volumes. If the data resides on ML1 for a determined period of time it can be migrated to ML2 (typically tape). This whole process can be summarized as a "Space management" process and is therefore only a small part of what tiered storage is about. Also the unit of granularity is a dataset (or a tablespace where DB2 is concerned) which could make a single reference to a single row of a table, cause the recall of a very large tablespace.

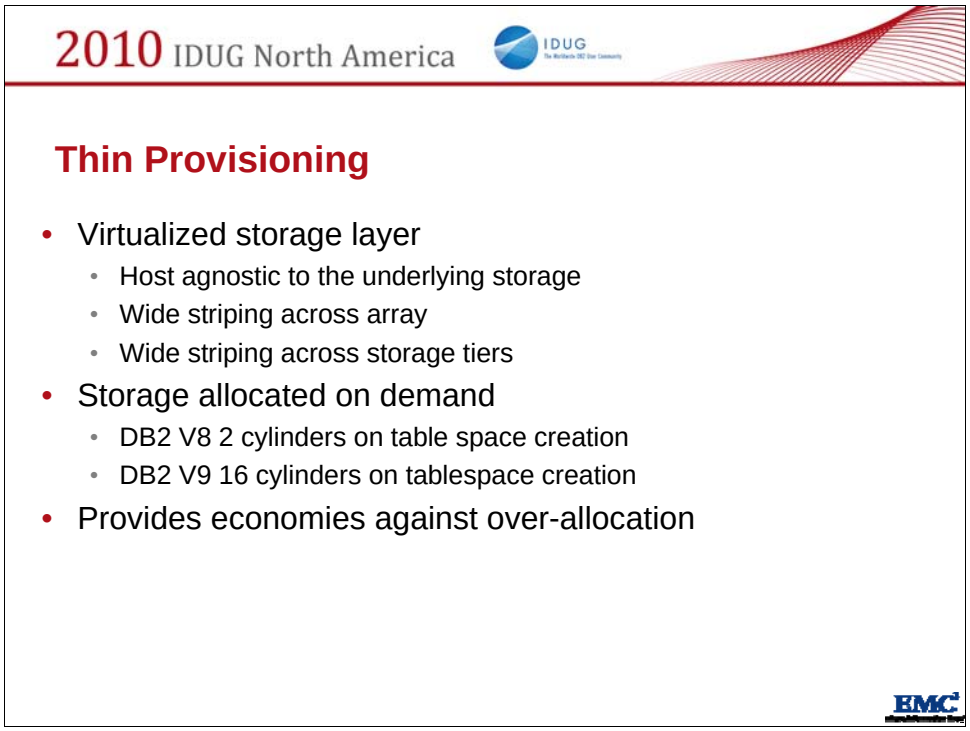
In addition, this process is not really suitable for a DB2 system, since the migrate process is based on dataset access and after the dataset is open DB2 keeps the dataset keeps it open (unless the 99% of the DSMAX limit is reached). In addition the recall for DB2 tablespaces can have an egregious effect on response time. HSM archiving of test and development DB2 systems may be a possibility though.

24



Tiered storage – the reality

I can tell you that there are very few tools to help you get to a tiered storage configuration. They are single tools that can help you move data like those listed here, and some of them do not require down time. But they can be labor intensive and error prone. Plus they assume a prior knowledge of data access paths and these are not always predictable. zDMF is the z/OS dataset migration



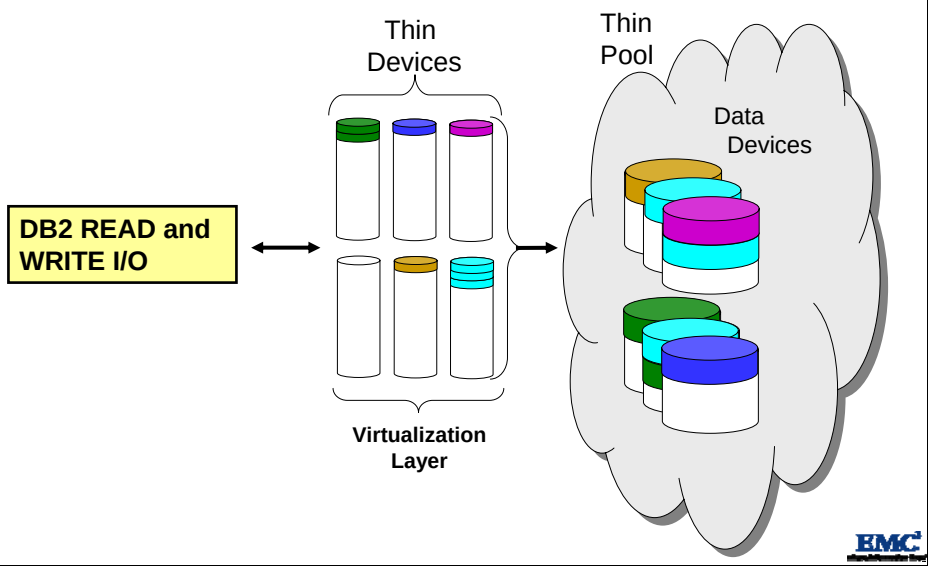
Thin Provisioning

Thin provisioning in an enterprise storage environment allows administrators to maintain one or more free space buffer pools to service the data growth requirements of all databases and applications. Storage is allocated on demand as it is written to. This avoids the poor utilization rates, often as low as 10%, that occur on traditional setups where storage capacity is over-allocated to individual databases, but remain unused (i.e. not written to).

With thin provisioning, storage capacity utilization efficiency can be automatically driven up towards 100%, without heavy administrative overhead. Organizations can purchase less storage capacity up front, defer storage capacity upgrades in line with actual business usage, and save the operating costs (electricity and floorspace) associated with keeping unused disk capacity spinning.

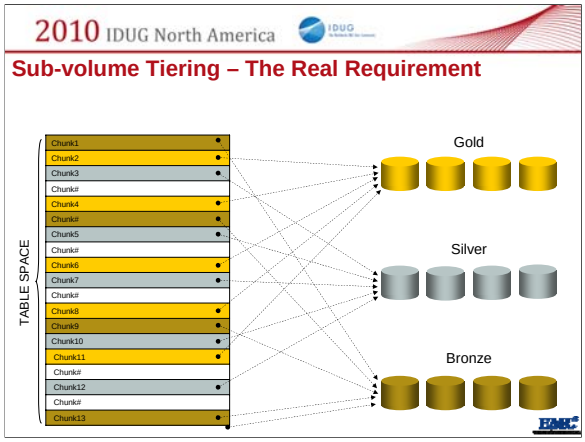
Previous systems generally required large amounts of storage to be physically pre-allocated because of the complexity and impact of growing volume space. Thin provisioning enables over-allocation or over-subscription.

Thin Provisioning (contd.)



Title

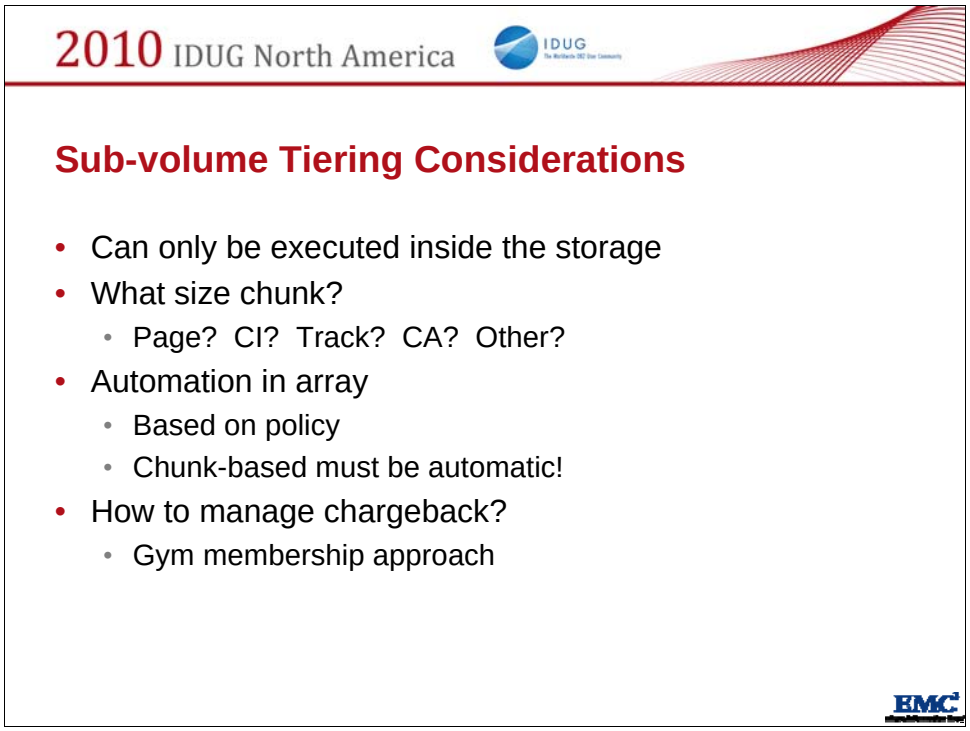
Month Year



HSM – the original tiering solution

HSM migration might be the "original" storage tiering solution. It allowed for data that was not accessed to be moved from production devices to a staging device and compressed to save space. The archive area on disk is called ML1. Typically ML1 devices/storage is less expensive/less performing than the primary devices, resulting in a cost savings to the customer. When the data is accessed that is on ML1 it goes through a process of "recall" and it is decompressed and restored to the source volumes. If the data resides on ML1 for a determined period of time it can be migrated to ML2 (typically tape). This whole process can be summarized as a "Space management" process and is therefore only a small part of what tiered storage is about. Also the unit of granularity is a dataset (or a tablespace where DB2 is concerned) which could make a single reference to a single row of a table, cause the recall of a very large tablespace.

In addition, this process is not really suitable for a DB2 system, since the migrate process is based on dataset access and after the dataset is open DB2 keeps the dataset keeps it open (unless the 99% of the DSMAX limit is reached). In addition the recall for DB2 tablespaces can have an egregious effect on response time. HSM archiving of test and development DB2 systems may be a possibility though.



What is needed?

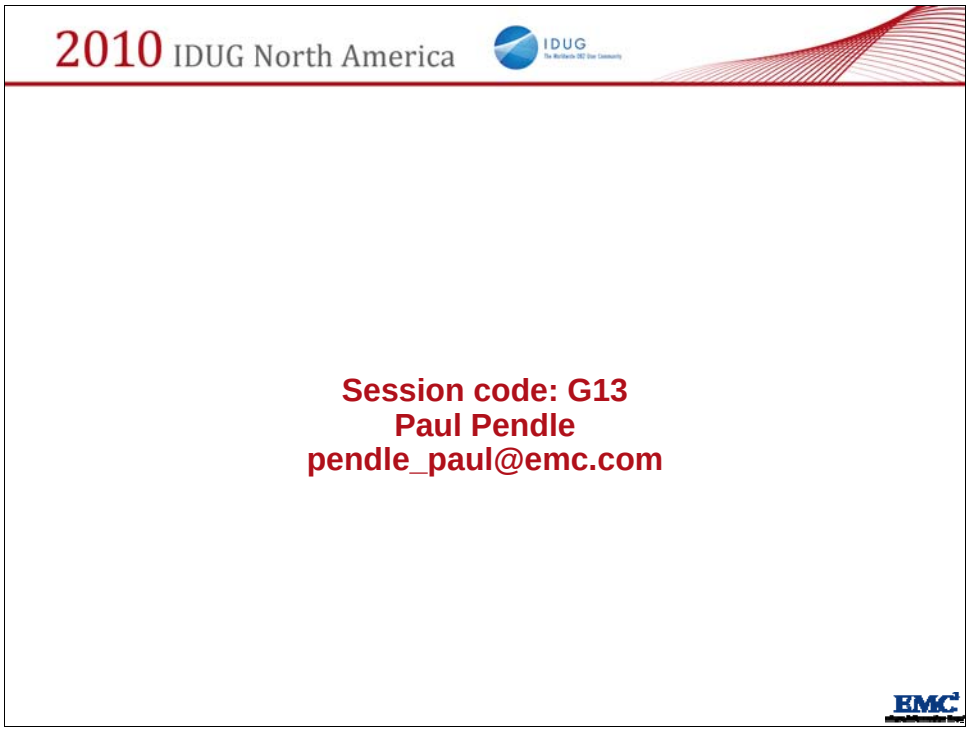
What you need are mechanisms that can move data between tiers, easily and transparently and the ability to control these through policies. It needs to have a finer granularity than a table, tablespace or LUN since the I/O demands on parts of these can vary and will vary.. Plus you need to be able so set policies regarding data that is to be treated as exceptional. And since your data may change it's access frequency over time and based on business imperatives, it needs to be self-learning and heuristic if you will. It needs to be automated in such a way that it runs without excessive management. And it needs to be transparent to applications/databases and operating procedures. If you can get all of this .. You can have a workable tiered storage system that will give you the greatest ROI and reduce your Total cost of Ownership (TCO).

And if it can be automated that would be even better

What is needed?

What you need are mechanisms that can move data between tiers, easily and transparently and the ability to control these through policies. It needs to have a finer granularity than a table, tablespace or LUN since the I/O demands on parts of these can vary and will vary.. Plus you need to be able so set policies regarding data that is to be treated as exceptional. And since your data may change it's access frequency over time and based on business imperatives, it needs to be self-learning and heuristic if you will. It needs to be automated in such a way that it runs without excessive management. And it needs to transparent to applications/databases and operating procedures. If you can get all of this .. You can have a workable tiered storage system that will give you the greatest ROI and reduce your Total cost of Ownership (TCO).

And if it can be automated that would be even better



Presenter Bio:
Paul Pendle, an employee of EMC Engineering based at Hopkinton, Massachusetts. Paul Pendle has over 30 years of experience in databases, hardware, software and operating systems both from a database administrator perspective and from a systems administrator/systems programming perspective. Paul started working with DB2 for z/OS with version 1.2.