Leveraging EMC Unified Storage System Dynamic LUNs for Data Warehouse Deployments

Abstract

This white paper discusses the rationale behind using EMC® unified storage system dynamic (pool-based) LUNs to support the implementation of data warehouses, and provides guidelines for tradeoffs and practical implementation considerations.

September 2010
Table of Contents

Executive summary ............................................................................................................. 4
Introduction ......................................................................................................................... 4
  Audience .......................................................................................................................... 5
  Terminology ..................................................................................................................... 5
Matching storage virtualization with evolving data warehouse trends ........... 5
Storage pools in EMC unified storage systems ............................................................... 6
  Thick versus thin pool-based LUNs ................................................................................ 8
  Pool-based LUN performance consideration for data warehouse ................................. 8
An EMC engineering case study ...................................................................................... 9
  Test environment ........................................................................................................... 10
  Testing approach .......................................................................................................... 11
  Observed testing results .............................................................................................. 11
    Analysis of observed results ....................................................................................... 11
  Initial data load ............................................................................................................ 12
    Analysis of the observed results ............................................................................... 12
Dynamic pool expansion and FAST .............................................................................. 13
  Engineering example of leveraging FAST in the test workload ................................... 14
    Observed test results ................................................................................................ 15
    Analysis of observed results ...................................................................................... 15
Conclusion ....................................................................................................................... 17
References ....................................................................................................................... 17
Executive summary

A primary goal of every CIO and CFO is to effectively leverage today’s wealth of information to develop deeper and useful insights into how the enterprise business has been running, and to help with changing and improving the business. Even though the cost to acquire, store, and manage large volumes of data is expensive, new data warehouses are continuing to spring up at a rapid pace throughout the industry. Properly implemented, a data warehouse provides the foundation of a single source of information that business analysts can use to conduct their analysis, and propose adjustments, to improve the effectiveness and profitability of the business with confidence.

A frequent challenge now facing many of the new data warehouse projects is that business analysts need as comprehensive a collection of data at their disposal as quickly as possible, while the IT functions supporting these users often are struggling with planning and acquiring the server and storage assets needed to cope with these users’ needs. The rate of new information generation and collection has been exploding, with the advent of many modern technologies, through which information is being generated and captured by automated systems, sensors, and machines as opposed to human data entry. With the ever-increasing CPU processing power, and the correspondingly lower cost of CPUs, application analysts can readily justify the attempt to processing with more detailed information to improve the statistical accuracy of their business model analysis. IT functions are expected to be able to fulfill the need by provisioning more resources, in shorter time, to satisfy those expectations.

The very trend of the business need has been a key driving force for many IT functions to seriously explore virtualization and consolidation. Provisioning silos of resources to satisfy different business project needs is no longer practical. Even if IT functions are given the budget to acquire more hardware for each new projects, there is just not enough IT-supporting infrastructure, including people, floor space, power and cooling, and so on, to support such growth.

Virtualization and logically consolidating different projects so they can share and leverage a pool of resources is quickly becoming a necessity rather than a luxury. It is no longer just a question of cost of IT infrastructure. It is really a matter of optimizing for the total cost of operation, for it is just as expensive, if not more so, for the business analysts to be wasting their time and effort conducting the studies with incomplete or inaccurate data, leading to conclusions that may end up hurting rather than helping the business.

EMC as a company has been steadily transforming our entire business and shifting our focus to creating solutions and products that enable IT functions to transition quickly and confidently toward a fully virtualized IT deployment model. In storage products, special focus has been placed on supporting the effective provisioning and management of storage by IT storage professionals through storage resource pooling. The focus is to simplify their tasks, allowing storage administrators to be able to provision any available storage resource that is already in their pool of physical storage assets. This means they do not have to be as concerned about isolation of storage usage, and can create silos to buffer against possible performance interference between different application projects. Storage resource pooling is a key component in the overall goal of enabling full resource virtualization.

Introduction

Understandably, there would be a certain amount of reservation for IT technologists contemplating a need to change from a storage management approach that they have been accustomed to and are comfortable with. This white paper shares some of the engineering testing conducted by EMC engineering that leverages pool-based LUNs (available at FLARE® release 30 and later) against typical data warehouse-oriented workloads. These tests serve as the basis for recommendations for practical approaches, with discussions of some of the pros and cons of certain options and tradeoffs.
**Audience**

This white paper is intended for storage administrators, database administrators, enterprise application administrators, storage architects, customers, and EMC field personnel involved with planning for infrastructure support to enable new data warehouse deployments, or extending the currently deployed data warehouses. It is assumed that the reader is generally familiar with the EMC® CLARiiON® storage system architecture and terms.

**Terminology**

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERP</td>
<td>Enterprise Resource Planning</td>
</tr>
<tr>
<td>FAST</td>
<td>Fully Automated Storage Tiering, automatically managed by the storage system</td>
</tr>
<tr>
<td>FAST storage pool</td>
<td>A storage pool that includes physical drives of multiple disk types, including possibly Flash, Fibre Channel, and SATA</td>
</tr>
<tr>
<td>FC</td>
<td>Fibre Channel</td>
</tr>
<tr>
<td>Flash Drive</td>
<td>Also known as solid state disk</td>
</tr>
<tr>
<td>LUN</td>
<td>Logical Unit Number, a storage object from the external storage system that can be referenced and manipulated by host applications</td>
</tr>
<tr>
<td>Pool-based LUNs</td>
<td>LUNs created out of available space in a storage pool</td>
</tr>
<tr>
<td>SATA</td>
<td>Serialized ATA</td>
</tr>
<tr>
<td>Storage pool</td>
<td>A group of physical drives inside a CLARiiON system designated to form a pool of available disk space</td>
</tr>
<tr>
<td>Thick LUN</td>
<td>A pool-based LUN that requires all the space needed to be immediately allocated from the pool when the LUN is created</td>
</tr>
<tr>
<td>Thin LUN</td>
<td>A pool-based LUN that does not require all the space needed for the LUN to be immediately allocated. As the LUN space is consumed, more space from the pool will be allocated up to the maximum allowed, or when the pool is out of usable space, whichever comes first.</td>
</tr>
</tbody>
</table>

**Matching storage virtualization with evolving data warehouse trends**

Data warehouses primarily organized to support reporting, decision support queries, and business analytic queries tend to drive a different I/O profile against the underlying storage system compared to traditional operational database systems. One of perhaps the most significant shifts in the recent year in the use of data stored is that more frequent brute-force scanning through the data has become commonplace. This has primarily been due to a shift in how the stored data is now being used.

Traditionally, data warehouses were used for generating certain comprehensive reports run in batch mode. Experienced SQL programmers, with a good understanding of the actual data distributions and schemas, were responsible for developing well-structured SQL reports optimized through proper data indexing, and data schema architectured to optimize for efficiently generating those reports crucial to the business. The format of the data used tends to be relatively stable, even though the volume of data being reported against may be growing in size.

More and more, structured and well-architected reports are giving way to ad hoc queries by business analysts, who are neither SQL nor physical data schema experts. With the rapid explosion of information collected through highly diversified data sources over the Internet, data schemas also tend to be evolving and changing more frequently. Attempts to optimize data I/O access through proper index organization become increasingly impractical from a time-to-deployment standpoint. Consequently, massively parallel
database engines that leverage a high number of commodity processing components to pour through big volumes of these datasets are becoming mainstream in the data warehouse/business analytic arenas.

To match up with these powerful processors, the underlying storage subsystem must also be structured in a manner to be able to grow and scale with each element of processing added into the mix.

The conventional approach of creating storage objects like LUNs tied closely to physical storage elements such as disk drives may offer more predictable performance characteristics for each object, but would create a significant barrier for scaling.

Instead, pool-based LUNs, created out of storage system pools by grouping together physical drives, with the ability to expand and grow by merely adding more physical drives to the pool without disturbing any of the existing LUNs already deployed, is logically a far more practical approach to satisfy the need.

The most common concern voiced in going to a fully virtualized storage model is performance consistency. The currently popular approach for handling complex queries involves dividing the business data into multiple partitions based on some business data criteria (for example, date ranges for sales data), then breaking down the query into parallel subquery tasks that can be performed in parallel against the different partitions of data. This approach typically relies heavily on the assumption that each partition of data of roughly equal volume can be retrieved from the underlying storage infrastructure at comparable speed. If access to one partition suddenly slows down for whatever reason, it often has the net effect of slowing down the entire query process. Consistent performance from each LUN used to store the data partitions is therefore important.

If the pool is used primarily to house the different LUNs forming the different partitions of the data warehouse (as opposed to housing a variety of LUNs not related to the same data warehouse), then much of this very concern is removed. By relying on the pool management software to evenly distribute all the stored data in the different LUNs holding the different partitions over all the physical drives making up the pool, we are actually ensuring that there would be very comparable and even performance delivered from each LUN when all the LUNs are being concurrent accessed.

In the event a query references only some of the partitions, or accesses some of the partitions more heavily than the others, spreading each LUN over as many physical drives as available actually helps to ensure that in the event of access skewing, the slowest path due to skewed data access can pick up more bandwidth headroom from the physical drives once the activities on other parallel partitions have completed.

Pool-based LUNs have been supported, primarily for thin provisioned LUNs, in the EMC CLARiiON storage platform for a couple of releases. In unified storage system release 30, a number of key enhancements have been added. For new data warehouse projects, the new pool-based LUN approach will benefit from the additional flexibility and ease of use, while delivering the level of performance and scaling required for most of these new projects. The following sections will provide more specific details of what can be expected.

**Storage pools in EMC unified storage systems**

The concept of storage pools was introduced into EMC unified storage systems two major storage firmware releases ago. The addition of pool-based LUNs support in our storage platform is an integral part of EMC’s overarching virtualization strategy to fulfill our vision of enabling customers to readily transform their IT functions by fully harnessing the advantages of an enterprise private cloud. Key enhancements have been added in the current release.

IT functions today are commonly faced with the challenge to support more business users and more complex processing, and store ever-increasing volumes of data, while working with a tightened budget. Furthermore, user requirements are often changing rapidly as a result of changing business needs and conditions, which are often difficult to anticipate fully for IT resource planning. By adopting virtualization,
physical IT assets can be much more effectively pooled and shared among all users, increasing the average utilization of those assets, and allowing a good deal more flexibility in reallocating resources to respond to changing business needs.

Traditionally, EMC storage system LUNs are found by selecting a set of physical disk drives managed by the system, selecting the usable space of each of those drives, and logically organizing the space from the different drives together as a usable storage object. The object is then presented to system servers to be used as a “disk device.” EMC unified storage can also take that storage object and superimpose a file system structure over the usage space, then present that file system as a network file system conforming to different network file system standards, including NFS and CIFS.

With traditional LUNs, it is often possible to follow where the applications are driving I/O into the specific drives inside the storage system. This allows performance issues due primarily to inefficient storage device I/O to be more readily identified and potentially addressed.

However, when a traditional LUN is created in the storage system, the space in the physical drives selected is immediately committed as part of the LUN. If a LUN of 100 GB is created out of five physical drives using 4+1 RAID 5 protection, 25 GB worth of space is used up in each of the five drives. Whether the application assigned to store data in this 100 GB LUN is storing 100 GB worth of data, or just 1 GB of data at the present time, none of the 25 GB from each drive can be used for any other purpose.

The virtual LUN migration utility does allow the content of that 100 GB LUN to be migrated in its entirety from those five disk drives to space from another set of drives. Frequently, this is done as part of information lifecycle management, where data that has aged, and become less valuable for the business, with less frequent access, may be migrated from the more expensive drives, such as Fibre Channel (FC) or Flash drives, into high density, lower cost, and slower drives such as SATA. But the entire LUN content, all 100 GB worth in our example above, would have to be migrated. For large traditional LUNs, this may not always be very efficient to do.

![Traditional LUNs vs Pool-based LUNs](image)

**Figure 1. Traditional unified storage system LUNs compared to pool-based LUNs**

With pool-based LUNs, generally a much larger set of disk drives in an array is selected to form a pool. Instead of slicing off a fixed amount of space from each of the drives to create that same 100 GB LUN needed for our application, the array software finds slices of space at a much lower size granularity from all drives within the pool that have unused space.
It may appear that by going to pool-based LUNs, with smaller slices of data for each LUN spread over all the physical drives, it might be more difficult to troubleshoot application performance issues that are related to the I/O behavior. But even when deploying traditional LUNs, it is often the case that in order to provision new LUNs to applications, it may be necessary to slice spaces out of the same set of drives (forming a RAID group) for more than one LUN. Keeping one LUN on each RAID group (from a set of drives) is usually impractical for most unified storage system customers.

By allocating smaller slices of space, and spreading them as evenly as possible over all available drives, the applications get the advantage of pushing their I/O against more physical drives.

With the storage system tracking and managing the allocated and used space for each LUN as finer-grain disk space slices, data stored in a certain portion of the LUN can be relocated readily by the storage system to different drives with different performance characteristics, such as Flash drives or SATA drives, adjusting to the frequency of the usage of the data stored in those slices of the LUN. This is part of the new Fully Automated Storage Tiering (FAST) enhancements in unified storage system release 30.

**Thick versus thin pool-based LUNs**

Pool-based LUNs can be fully allocated, also known as a thick LUN, as opposed to a partially allocated LUN, also known as a thin LUN. When a thick LUN is created, the amount of space is immediately reserved. A LUN meta structure is created, but the actual data slices are not written into the different drives in the pool until the application starts to actually write data into the LUN.

When a thin LUN is created, a similar LUN metadata structure is also written into physical storage. However, the used space from the pool is only reduced by the amount of space used up for establishing the initial slice of the LUN holding the metadata plus some extra data space headroom. Only as data is written into the LUNs and actual data slices allocated to accept the new data, is the used space information for the pool updated. As a result, overprovisioning is possible with thin LUNs. For example, we can create 100 thin LUNs of 1 TB each from a pool that only has 20 TB of physically usable disk drive space. With thick LUNs, only 20 such LUNs of 1 TB can be created from the same pool.

**Pool-based LUN performance consideration for data warehouse**

For data warehouses, the name of the game for the underlying storage system is typically sustained bandwidth. The practical read I/O pattern is quasi-random, because complex user queries working against large amounts of data are parallelized by the database engine into concurrent reads against multiple data segments stored as partitions in the warehouse. Coupled with the fact that the data warehouse is invariably supporting multiple users running such queries, and the total I/O driven down to the LUNs storing the warehouse data can be expected to be randomly going after different parts of all the LUNs.

To achieve high read bandwidth, it is important that data is stored in relatively large chunks on physical rotating storage, such that each read request from the host can be satisfied with a single drive seek and access.

As long as pool-based LUNs are using slice sizes large enough to allow the DBMS host reads to pull data off the drives in relatively large sequential chunks, read bandwidth expectations from pool-based LUNs should be quite comparable to isolating data partitions of data warehouses on separate drives using traditional LUNs.

In the current implementation of pool-based LUNs, thin LUNs are implemented using relatively smaller space slicing. The main advantage of using thin LUNs is to optimize for space utilization. Using small size slices avoids wasting space inside a slice, thereby ensuring all available space from the drives in the pool can be leveraged for storing data.

Thick LUNs are currently implemented using the larger (1 GB) slice size. Thick LUNs are intended to be generally provisioned with the expectation that all LUN space will be used (hence, no space oversubscription is allowed with thick LUN provisioning). Thick pool-based LUNs should therefore be used to support implementing data warehouses (as opposed to thin LUNs).
With pool-based LUNs, there is undoubtedly additional overhead in managing the metadata structures. Whereas in traditional LUNs, the 100 GB of the warehouse data partition are known to be stored physically inside, say, five specific physical drives (forming a RAID group), and all I/O to the LUN data will always be directed to those somewhere within those five drives, with pool-based LUNs, to read through all 100 GB of stored data, we will have to look through the metadata to find all slices of data spread over many more drives at different drive offsets.

Again, with thick LUNs being used for implementing data warehouses, some of the performance concern is minimized. Thick LUNs use relatively wide slices, so the number of metadata entries to track even larger LUNs are reduced (for example, the 100 GB LUN is “described” by 100 “slice tracking entries”). The smaller tracking map can then often be cached in storage system memory. If the storage system is further using the new FAST Cache feature leveraging Flash drives inside the storage system as “extended cache,” that would make it even that much more efficient for looking up the tracking data. As the nature of DWs is far more heavily read-oriented, the need to continually expand and add slice tracking entries to the metadata for the LUN is far less frequent and intrusive.

**An EMC engineering case study**

The following system and test scenario was put together by the EMC engineering team to provide a particular data point comparing a sample data warehouse implemented using a traditional LUN versus a thick pool-based LUN approach.

The test workload used for this experiment is a point-of-sales information data warehouse. This is a sample test workload developed by the Oracle Data Warehouse engineering team, and shared with partners like EMC to help with jointly qualifying the effectiveness (and identifying opportunities for enhancements) of a total system solution involving both Oracle and EMC products working in tandem.

The test warehouse is populated with a little over 2 TB of raw data, stored as compressed tables within Oracle. The key point-of-sales information tables that make up more than 90 percent of the total warehouse fact data are partitioned by date ranges, covering multiple years of sales information, and subpartitioned by hash.

The query streams involved a mixture of relatively complex queries in examining trends and correlation of different facts related to different sales scenarios, to specific queries that return with “empty results.” The general idea behind the mix of queries is to provide a more realistic simulation of real-world business queries, where a certain amount of ad hoc queries without necessarily a tightly bound search target is predefined. In other words, some of the queries involve “brute force” plowing through most of the data to determine if there are common threads relating different sales behaviors, in the hope of finding new insights that actually do not end up yielding specific results after all the processing. These types of rich data “explorations for new business insights” are becoming more mainstream with most of the new DW/BI deployments today.

Recognizing that the thrust is really about how well a particular deployment can cater to multiple users, Oracle engineering has organized the testing methodology to focus on the maximum number of concurrent users accommodated by the system under test. Starting with a workstream where only one user running a sequence of queries against the warehouse data uses the entire system and establishing that as a target service time baseline, an “acceptable” user service time level is defined for the system based on that baseline reference. The system is deemed oversubscribed when the overall user experience for the concurrent user workload would take more than 2.5 times as long to finish up all concurrent user job streams compared to the single user baseline.

Note that the key metric of this test workload is the number of concurrent users that can be admitted into the system while observing a required response time service level for the system under test. As such, this is obviously not just about the raw speed and feed of the underlying storage system or storage configuration.
(such as LUN layout choices). But the overall system throughput hinges still on the underlying storage system being able to deliver a steady data feed to the servers to keep all concurrent user queries running and processing within the Oracle database system. A significant degradation in the rate of data delivery due to a revision in the storage configuration would be readily reflected in the total number of users that the system can practically sustain.

**Test environment**

**Key system software components**
- Oracle Database Real Application Cluster Enterprise Edition, Release 11gR1 (with the partitioning option)
- Oracle Enterprise Linux 5.4 x86_64
- EMC unified storage system release 30

**Key system hardware components**
- Cisco UCS system 5108 with half-width blade servers (four blades)
  - 2 x 4 “Nehalem” CPU @ 2.9 GHz
  - 48 GB RAM
  - Mezzanine card with 2 x 4 Gb FC HBAs and 2 x 10 Gb Intel NICs
  - 2 x Cisco 6120XP fabric concentrators
- Cisco Nexus 5000 40-port NIC switch
- Cisco 9124 32-port SAN switches
- EMC unified storage system family CLARiiON CX4-960 (two systems)
  - 7 disk array enclosures per array, with 15 FC drives of 15k rpm @ 450 GB/drive
  - 120 out of 135 drives used for a data warehouse data store

---

**Figure 2. Test system block diagram**

---

*Leveraging EMC Unified Storage System Dynamic LUNs for Data Warehouse Deployments*

*Applied Technology*
Testing approach

Using the UCS system, two Oracle Database 11gR1 Real Application Clusters (RAC) were deployed. The first cluster (designed by the two green color blades) was configured to run against a 2TB database built out using the 120 drives from the CX4-960 on the right. Forty traditional RAID groups (2+1R5) were configured from the 120 drives, and a separate LUN was created on each RAID group. All 40 LUNs were presented to the green-colored RAC nodes to be used to store the database under Oracle Automatic Storage Management (ASM) control.

From a separate CX4-960 array running the same firmware level, with an identical number of enclosures, front- and back-end connectivity, and same type of drives, a 120-drive storage pool was created. From the storage pool, 40 separate thick LUNs were provisioned, using exactly the same size as the traditional LUNs. The pool-based thick LUNs were then presented to the two red blades in the UCS chassis to be used as the data warehouse store for the same 2TB database. The database from the red storage pool was manipulated by the Oracle RAC nodes forming the red-colored cluster.

After the logically identical (in content) data warehouse databases of 2TB were created and loaded on both clusters, the same concurrent user testing scripts were run on both clusters to determine the maximum concurrent users supported by each cluster.

Observed testing results

The single-user workload stream baseline was 82 seconds (an average of 11 queries executed serially for the user). The business-defined service level was that average stream responsiveness was not to exceed 205 seconds.

Table 1. Traditional LUN vs. thick pool LUN

<table>
<thead>
<tr>
<th></th>
<th>Maximum concurrent users</th>
<th>Average stream response time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Traditional LUN (40)</td>
<td>32</td>
<td>204.3</td>
</tr>
<tr>
<td>Thick pool LUN (40)</td>
<td>28</td>
<td>203.5</td>
</tr>
</tbody>
</table>

Analysis of observed results

The storage unit with the traditional LUNs was carefully configured with a “narrow” 2+1 RAID 5 configuration. For the purpose of driving the storage system to sustain the best bandwidth, the specially built-out LUNs based on that RAID selection are generally expected to deliver the best bandwidth from the drive. The downside to that choice is the higher capacity overhead to accommodate the larger volume of parity data required (33 percent of total physical disk capacity devoted to parity data).

The virtualized disk pool available starting with release 30 attempts to create a better balance between the general overall bandwidth and space utilization. By default, a choice of RAID 5 for the space organized within a pool does not offer the option of creating the pool with a specific RAID configuration. Implicitly, the system defaults to a wider RAID 5 configuration organizing those same 120 physical drives to form the pool.

For the particular test workload, the wider default RAID 5 striping degrades the overall data scanning bandwidth slightly. As a result, the overall system effectiveness in supporting concurrent complex user query streams degraded some.

While it was possible to go through a considerably more manual and tedious process to try to rebuild the pool of 120 drives, carefully crafting the underlying RAID configuration to 2+1R5 to try to match maximum throughput with the tradition LUN configuration, that effort was not attempted for this case study.
One of the main purposes for going to dynamic pool-based LUNs as opposed to using a more tedious and labor-intensive approach to carefully carve out RAID groups and LUNs customized for a particular workload with traditional LUNs is to achieve greater usage flexibility, lower the cost of storage management and provisioning, and shorten the time to provision and deploy.

With the more common practical situations of having multiple types of data from different applications, with different access characteristics, stored as pool-based LUNs sharing the same set of drives in the pool, customizing the configuration of the drives in the pool to try to optimize for one set of data may end up negatively impacting other application data.

EMC engineering’s recommendation, therefore, is to try to avoid special customization when using a pool-based LUN approach. If it is deemed important to compensate for more performance, the flexibility of space pools offers other options to make up that ground. It is simple to expand pools by adding more drives, so that data will end up being spread across more physical drives. In particular, when the pool is augmented by adding a small quantity of Flash drives judiciously, the new FAST feature readily makes up for the bandwidth gaps with no retuning effort. These options will be covered in more details in the following sections.

**Initial data load**

The initial data load for the 2 TB worth of data for the test workload was done by loading through external tables. In each case, the load was done (to the traditional LUN setup, as well as to the thick pool-based LUN setup), using one of the two RAC node instances. In each case, the same files from the same LUN on a separate EMC CLARiiON CX4-120 system were used to drive the load.

- The elapsed time to load the initial 2 TB data to the ASM DATA group on traditional LUNs was 3 hours, 26 minutes.
- The elapsed time to load the initial 2 TB data to the ASM DATA group on thick pool LUNs was 4 hours, 2 minutes.

**Analysis of the observed results**

To some degree, the load time in each case was constrained by how fast the external table data was read up by the Oracle DBMS engine from the separate CX4-120, since that provided the data needed for the engine to build up the DB pages needed to be stored into the LUNs in the storage system.

Even though the space for thick pool LUNs was fully deducted (and allocated) from the amount of free space in the pool, the 1 GB data slices that were being written were physically allocated from the different drives as the data were actually written to the LUN. For each data slice allocated and written to, a new metadata entry was created to track the new slice.

Consequently, there was more work being done by the storage system supporting the new data writes from the initial data load test into the pool-based thick LUNs.

Had the 40 thick pool LUNs been already written to before, such as having loaded the entire database, dropping the database, and repeating the same data load again, the time difference between a traditional LUN-based load versus the thick pool LUN load would have been considerably reduced. Reusing thick LUN space slices that have already been allocated would incur much less overhead in manipulating the slice tracking metadata. If the database load time window is tight, there may be some advantage in using a simple OS raw data bit copy type utility (such as “dd”) to “pre-zero” the pool LUNs before starting the data load.
Dynamic pool expansion and FAST

Another key advantage in changing over to use pool-based LUNs as opposed to the traditional RAID group associated LUN (that more rigidly binds the data stored in the LUNs to specific physical drives) is the enablement of the FAST feature.

With the pool-based LUN organized as slices of space from all the different physical drives in the pool, maintained by the storage system, there is now the option for the storage system to dynamically adjust and relocate the data slices among the different drives to automatically smooth out disk usage activity hot spots.

In particular, the enhanced release 30 code supports the use of physical drives of different performance characteristics within the same pool.

The ability to accommodate physical storage of different types of drives within the same pool is a major advantage for most data warehouse deployments.

Data warehouses tend to accumulate ever-increasing volumes of stored data over time. Also, the rate of stored data accumulation has been accelerating. In most efforts to mine the massive volume of accumulated data for valuable new insights that can help to transform and improve a business, not all the data is necessarily of equal value, and is used by all analytic queries.

The most common example is business data accumulated over time. In looking for information like sales trends, or changing customer demographics, data that is more current, such as data for the past two or three years, is usually more relevant to examine compared to the data from 10 years ago. Investors looking for new investment options tend to want to compare performance of different stocks or mutual fund trends in the last two to five years as opposed to the lifetime of each option.

Data, especially historical data gathered from a long time ago, tends to decrease in value for the purpose of analytics. The frequency of access for those data elements drops off over time. Effective information lifecycle management is an integral part of any effectively implemented data warehouse.

While it is technically possible to organize partitions of business data to be stored in separate storage LUNs, and leverage EMC unified storage system LUN migration facilities to implicitly move the entire LUN from one tier of storage (for example, stored on FC drives) to a different tier (for example, SATA), it requires that the database must be partitioned and organized in such a way that each logical partition that may have to be moved must reside in its entirety within a single LUN. In many database implementations, this is not always practical to achieve.

If the data in the warehouse goes back 20 years, and we try to organize each month of data to be stored in one logical partition tied to one traditional LUN, we will need 240 distinct LUNs. And for every new month’s worth of data, another LUN would have to be provisioned from storage and added into the database. This quickly becomes difficult to manage from an operational standpoint.

With the new pool-based LUNs, a new LUN can be provisioned from the pool readily without having to worry about adding new storage physical drives. Also, it is possible to expand a LUN, and store multiple logical partitions into the same LUN.

For example, instead of keeping 240 separate LUNs to track each month’s worth of business data for the past 20 years, a new logical approach is to keep 12 LUNs for each month of the year. All the business data from January of the past 20 years can be all logically folded back into that one pool-based LUN holding all the January data.

The different slices of the LUN hold data from different years, and they are spread out to all drives in the pool. So, when all the January partitions for all the years are needed, there will not be a bottleneck for access to data from a limited set of physical drives.
As data ages, the slices that are holding the aged data are visible to the storage system. As activity for these slices drops off, the slices become candidates to be automatically relocated by the system from one storage tier to a lower (slower, but more storage cost-effective) tier. The high access frequency data, such as the data for the past few months, or days, or hours, with heavily skewed accesses, will automatically be targeted for relocation to a faster storage tier, such as Flash drives. No application- or database-level changes will be required to manage this data relocation.

As the volume of data being managed continues to increase, this type of storage-managed automatic data relocation across storage tiers is crucial for both the efficiency of data access and manipulation from the warehouse, and to optimize the cost of storing the data over time.

**Engineering example of leveraging FAST in the test workload**

To further assess the advantage of leveraging the FAST feature, the test environment described on page 11 (using the 40 pool-based LUNs from the 120 FC drives) was revised slightly. Instead of starting with a pool with 120 FC drives, the storage system was revised to use a pool of only 100 FC drives.

Also, instead of the two half-width blades, a single full-width “Westmere” UCS blade with two six-CPU cores at 2.9 GHz was used to drive the test workloads. The server change was made partly with the intent to get an assessment of the new UCS full-width blade with expanded memory running a single-instance Oracle engine compared to running with a two-node RAC cluster.

For the baseline comparison, 120 FC drives continued to be used to form the 2+1R5 traditional LUN.

![Diagram of main lab network](image-url)

**Figure 3. Testing with a pool augmented by Flash drives**
Observed test results
The service level target was for user stream responsiveness to fall within 205 seconds per query for different queries in the stream.

Table 2. Traditional LUN vs. thick pool LUNs (with FC and Flash drives)

<table>
<thead>
<tr>
<th></th>
<th>Maximum concurrent users</th>
<th>Average stream response time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Traditional LUN (40) 120 FC drives</td>
<td>30</td>
<td>205.4</td>
</tr>
<tr>
<td>Thick pool LUN (40) 100 FC drives</td>
<td>24</td>
<td>203.8</td>
</tr>
<tr>
<td>Thick pool LUN (40) 100 FC drives + 5 x 73 GB Flash drives</td>
<td>28</td>
<td>199.7</td>
</tr>
</tbody>
</table>

Analysis of observed results
The full-width UCS Westmere CPU-based blade with 12 higher-speed CPU cores and 96 GB of RAM actually matched up well to running with two-node RAC using the two half-blades, with a total of 2 x 4 x 2, or 16 total CPU cores, and a total of 2 x 64 GB, or 128 GB of total memory, using the traditional LUNs.

When dropping off to using a FC drive-only pool with 100 physical drives, the I/O bandwidth sustainable dropped off further.

However, by adding back into the pool five additional drives of 73 GB Flash drives (which the storage system implicitly organized as 4+1R5, yielding, in essence, around 260 GB of “tier 0” usable space incremented to the existing FC-only pool), the overall system throughput was brought back up close to matching that of the full 120 drives used in organizing the 40 traditional LUNs.

One may well argue that on the surface, we had exchanged 20 FC drives (using the traditional LUN approach) for five Flash drives, just to bring the overall system throughput for this test workload back to about even. This may not appear to be particularly attractive from a cost tradeoff standpoint.

But there is a bit more than just trading 20 FC drives for five Flash drives.

Flash drives generally have much lower access service latencies, because there are no rotating and mechanical parts.

When the complex query is broken down by the DBMS query optimizer into parallel processing subtasks, the time for the query is determined usually by the subtask that takes the most time.

At the current cost point of Flash drives, it is obviously impractical to try to store all the warehouse data exclusively on Flash drives.

But when we have only the room to store part of the data, the potential improvement of execution time for a query is now much more a function of the amount of data involved in that query. When the data for all the subtasks can be completely promoted and housed in the Flash drives, the task time for each parallel subtask improves significantly. However, if only some of the parallel subtasks manage to speed up due to the data slices they work against having been promoted to the Flash drives, but the other subtasks continue to have to pull their data from the FC drives, the overall query time for the entire query may not improve as significantly, if at all. Perhaps the key effect of having promoted some portions of the data to Flash drives is that it takes some of the I/O pressure off the remaining FC drives holding the data not promoted, thereby helping to improve read access service for the remaining data.

In the set of queries used in this particular test workload, most of the queries attempt to access the majority of the 2 TB data stored. As a result, only about 25 percent of the data typically used ended up being
promoted. The data pages were stored as compressed pages by Oracle, averaging to about 50 percent compressed. So, within that 260 GB worth of Flash pool space, about 500 GB of the warehouse data resulted in being promoted.

Other real-world workloads that involve index lookups, manipulation of dimension cubes, more frequent single data row targeted search, and so on, mixed in with these large parallel queries, potentially can benefit far more significantly from having the data slices holding indices, cubes, temporary tables, and so on promoted to Flash drives.

Furthermore, if the pool is also used to hold other data over and above just the data partitions of this data warehouse, it may turn out that the other data slices get promoted, which are far more I/O service latency-sensitive compared to the data warehouse data partitions. That can help to relieve the physical drive access contention against the warehouse data access, thereby improving the overall data analytic workload against the data stored.

It is useful to keep in mind that when it comes to purely driving I/O with the focus on read bandwidth, if the host I/O is being pushed in large chunk sizes down to storage, minimizing the amount of physical drive head seeks on rotating disks, FC drives (and even SATA drives) can deliver quite impressive MB/s read per drive (readily in the range of 30 MB or more with chunk random reads using 512 KB or bigger I/O sizes). As opposed to that, solid state Flash drives used typically are just delivering read bandwidth in the order of 200 MB/s practically. Flash drives are much better leveraged to drive I/O latency-sensitive subsets of tasks in data queries to improve those areas of I/O access (such as index reads, summary and partitioned temporary tables built to facilitate execution of correlated queries, and other areas).

In some cases, it may be more cost-effective to expand the pool with more FC drives that would be comparable in cost to a handful of Flash drives, if the objective is to increase overall MB/s of data read from the LUNs in the pool. The relevant question for storing data warehouses in LUNs from the pool is whether there are distinct portions of the warehouse that are expected to have heavier access frequency, or needing generally lower read I/O service latencies, compared to the bulk of the data, usually considered “fact” data, that would warrant being promoted and stored in the Flash drives. For example, if the data usage model is such that the most recent 12 months’ worth of data out of the many years will generally have the majority of usage because of the business model, then it would make sense to add enough Flash drives to the pool to accommodate about 12 months’ worth of data. As the last 12 months’ worth of data get accessed repeatedly, the storage system’s internal usage heuristic monitoring will trigger those 12 months’ worth of “LUN slices” to be promoted into the Flash drives. When the new month 13 data enters into the warehouse, and starts to be used by the business queries, the slices holding the first month of “hot data” will get displaced from the Flash drive-based pool space, replaced by the new month 13 data slices.

The key lesson from this test exercise is the difference in how to try to improve our total system throughput. The more customized approach of carefully designing the data schema, then laying out the storage using traditional LUNs, resulted in a total system stack that theoretically can support slightly more concurrent users.

Using the same data schema and starting with pool-based LUNs, but trying to keep the number of physical drives relatively constant, we witnessed a somewhat lower total system throughput.

By adding a handful of Flash drives to expand the pool, and without changing the database or server OS, or anything else, the total system throughput was brought back up to compare more favorably against the carefully designed and planned out storage model using traditional LUN.

In addition to being able to get a new warehouse launched and deployed quickly so that the business data analysts can get on with their job of looking for that key business insight to change and boost the business, a second common challenge for new data warehouse projects is to accommodate rapid and often unanticipated growth.
Business information analytics and business intelligence derived from data assets stored in data warehouses are generally strategic projects. When a key new business insight discovered through the one project has demonstrated success and added value to the business, it is natural that executives would want to do more with the data assets, and to spend more on these projects. Hence, many data warehouse projects, and new projects, often expand in scope very rapidly and unexpectedly, feeding on their very initial success.

By going with the pool-based LUN approach, it is considerably faster to expand the supporting infrastructure to contend with the rapid change in these project requirements. While simply adding more physical drives into the pool, and dynamically provisioning more pools and LUNs, may not always yield as high a level of performance for the system as one that is carefully designed and storage layout finely tuned, being able to respond quickly to new requirements with an expanded infrastructure that is “good enough” is often more crucial to ensure that the business data analysts can sustain their effort and momentum. Businesses engage in pursuing these strategic projects to ensure the company remains competitive and prosperous in a highly competitive and rapidly changing marketplace, and timeliness for producing new business insights means everything to such projects.

Conclusion

Under release 30 of EMC unified storage systems, EMC has put new emphasis on pool-based LUNs, and many of the new features including FAST based on those LUNs. Pool-based LUNs are an integral part of EMC’s strategic vision of Journey to the Private Cloud.

The agility of IT to provision and reallocate resources quickly to accommodate rapidly changing needs of the business data analysts mining the ever larger and richer data assets collected in new data warehouses is quickly becoming a necessity. Embracing a private cloud approach is therefore a natural consequence to meet this key challenge.

As with most forms of virtualization, there is always some concern for the amount of performance compromise. The characteristics of I/O common in most complex business analytic query processing systems actually mitigate some of these concerns.

The ability to include multiple drive tiers in the same pool, and the automatic assignment of all the LUN slices to the space available in different parts of the pool, based on heuristically detected changes in the usage pattern of the different slices, provide true and complete support for managing the vast volume of accumulated warehouse data through their usage life cycle.

By adding more drives, including extremely high-performance drives such as Flash drives, to expand a pool, to boost both performance and capacity of the pool, without the need to change how the data is logically stored inside the storage system for a rapidly growing data warehouse, many of the IT challenges of adjusting the storage layout quickly to accommodate the demand imposed by the rapidly changing need of the successfully deployed data warehouses are avoided.

As the pool-based LUN support is part of the overall EMC virtual IT data center vision, continued engineering development investment can be expected to improve all aspects of the new features, including FAST, service level management, usage accountability, security, and other key areas crucial to a successful IT transition to the private cloud.

Hence, EMC is strongly recommending that the new pool-based thick LUNs be considered seriously as the storage methodology to adopt for deploying new data warehouse projects.

References

- **EMC CLARiiON and Celerra Unified FAST Cache - A Detailed Review** white paper
- **Leveraging EMC CLARiiON CX4 with Enterprise Flash Drives for Oracle Database Deployments** white paper

---

Leveraging EMC Unified Storage System Dynamic LUNs for Data Warehouse Deployments
Applied Technology

17
- Maximize the Performance Benefit from Enterprise Flash Drive (EFD) by Sharing Through Virtual Provisioning (VP) in the EMC white paper
- CLARiiON Virtual LUN Technology white paper
- EMC CLARiiON CX4 Model 960 Networked Storage System specification sheet
- Introduction to the EMC CLARiiON CX4 Series Featuring UltraFlex Technology white paper
- EMC CLARiiON Best Practices for Performance and Availability: Release 30.0 Firmware Update white paper