Detailed Report for Documentum 5 with A Billion Object Docbase (Loading and Multi-user Retrieval) With Windows 2000, Oracle 9.2, and NetApp Disk Storage

IMPORTANT:

This document requires a non-disclosure agreement between Documentum and the customer before being shared with the customer.

Author: Ed Bueché

Table of Contents

Test Location and Test Team Members

Test: Load throughput and Content retrieval for a Docbase with 1 Billion objects

Location: Documentum Performance Lab, Pleasanton, California

Test Team Members:

Name	Company	Function	Email
Ed Bueché	Documentum	BM planning and execution, and Database design.	Bueche@documentum.com
Bill Kullmann	**Documentum**	Bulk content loader development and execution	Bill.kullmann@documentum.com
Ashraf Elessawy	**Documentum**	ASCII row generator development	Ashraf.Elessawy@documentum.com
Brian Shin	**Documentum**	Webtop search customizations	Brian.shin@documentum.com
Sri Gangoor	**NetApp**	NearStore & Filer technical support during benchmark	Sridhara.Gangoor@netapp.com
John Kim	**NetApp**		John.Kim@netapp.com

We also received helpful assistance throughout the test from various other members of Documentum Engineering, Consulting, and IT too numerous to name. However, we like to give special thanks to Roger Kilday (for Documentum Content Server detailed internal support), and to Documentum's IT staff (especially Rey Gonzales and Tony Smith) for quick response and great support on the hardware throughout the several month project.

Executive Summary

The following test explores Documentum scalability after one billion objects have been loaded into the content repository (docbase). The repository's scalability is defined by its ability to store new documents and retrieve existing ones with little performance degradation compared to a system with few objects. In industry such a large number of content objects in a single repository can arise in several areas. These include "imaging" (for example, the archival of TIFF images of customer forms) and "email archiving" (the archiving of emails for regulatory purposes).

The drop in prices for disk storage and server hardware have allowed customers to increase their "retention" times for their content to a point in which the total number archived could reach billions after a few short years. A load rate of 125,000 images / hour during an 8 hour busy day, could lead to a billion objects in less than 5 years that must be online for regulatory purposes.

The focus of this study centered on several areas:

1. *User response time*: retrieve content based on some reasonably selective range criteria or highly selective criteria in a multi-user environment.

2. *Content storage throughput*: Once a billion objects have been loaded, will subsequent loads take a significantly longer time?

3. *Database maintenance*: How long do typical database activities take? Can Documentum take advantage of database partitioning and how does this help performance?

4. *Mixing and consolidating of applications*: A typical enterprise will have many different content management applications with perhaps significantly smaller sizes of content objects. What are the advantages / disadvantages of mixing these applications into a single large Content repository?

The results of the testing showed that Documentum 5 scales at a billion objects:

1. A real-world, search-and-view application based on Documentum's Webtop user interface delivered excellent response time to over 1,000 concurrent users even when in the face of frequent system-wide queries that returned hundreds of results.

2. The content storage throughput of dual input loaders on the billion object repository was diminished by only 8% over a system with few objects.

3. The system maintained excellent user response while only employing "local indexes" on a partitioned database. Since partitioned database maintenance operations scale well on local indexes (as opposed to "global" ones), this ensures that the content repository could leverage state-of-the-art database technology for ease of maintenance.

In addition, the test demonstrates that Documentum's ability to scale well on inexpensive hardware. All of the servers involved in the test are Windows/Intel-based ones. In addition, the connections to the Netapp Filer was over Copper-Gigabit Ethernet iSCSI rather than the more expensive Fiber channel.

As was expected, however, scaling to a billion objects requires more effort in application design and production maintenance. The most noteworthy observations where that:

1. If a query uses large table scans, then its response time will be in terms of hours rather than minutes given the size of the tables. Hence, we recommend enforcing query resource limits in production (not often done).

2. Pre-testing every application query on a reasonably sized database prior to production is mandatory, not optional.

3. There may be a good business need to mix a smaller content application with one that is based on a billion objects, however, poorly formed queries in the less scrutinized small application might significantly degrade when mixed with the larger repository.

This testing effort did not cover an exhaustive list of scenarios and functionality. In particular, it does not cover:

1. Maximum load rates: Our database server machine was limited to two 1.4 GHz processors. Tests for some larger "busy hour" load rates are covered in other Documentum tests conducted in 3Q2003.

2. Time constraints did not allow an exhaustive examination of all Documentum user interfaces, the full e-mail archiving software environment, complex security models, workflow, and full text indexing.

documentum

Benchmark Results Summary Page

Hardware Vendor & OS	Database & Application Svr Vendor	Benchmark Version	Test Start and end Dates	Documentum Versions	System availability Date
HP/Windows Storage: Netapp	Oracle 9.2.0.4 IBM WAS 5.0.1	Webtop 5.2 with a customized advanced search	8/11/2003 to 10/24/2003	Content Server 5.2, Webtop 5.2 , DFC5.2	3Q 2003

Test Name	Docbase Size (# docs)	Metric	Hardware Configuration
LDB-Retrieval-1	1,006,632,000+	1,000 users/30 min	BM drivers: 6 x 750MHz dual, Solaris WAS/Webtop: 4 x 1.4GHz, dual Windows Content Server: 1.4 GHz, dual Windows RDBMS: 1.4 GHZ, dual Windows
Index-Set-Rebuild-10 million objects		60 min	
Compute-statistics-10 million objects		10 min	RDBMS: 1.4 GHz dual processor windows
Dual loaders		106 msec per object	BM Driver: 1.4 GHz, dual windows Content Server: 1.4 GHz dual Windows RDBMS: 1.4 GHz dual Windows
Dual loaders		98 msec per object	

Hardware Tested

Number	System	CPU	Memory	Disks	Purpose
1	HP lp2000r (Windows 2000)	2 x 1.4 GHz Intel	2 GB	> SCSI to four 36 GB internal disks > iSCSI/Gig Ethernet to Netapp Filer with 84 drives and 3.6 TB usable space,	Oracle RDBMS
1	HP lp1000r (Windows 2000)	2 x 1.4 GHz	1 GB	> SCSI to single 36 GB disk > CIFS share over Gig Ethernet to Netapp Nearstore with 58 drives and 8 TB of usable space	Content Server
4	HP lp1000r & HP2000r	2 x 1.4 GHz	1 to 2 GB	Internal drives	Webtop / WebSphere servers
5	Assorted Sun servers	1 to 2 x 750Mhz			Benchmark drivers

Test Summary Description

This test covers several performance aspects faced in an imaging or email archiving production environment once that Content repository (or docbase) has reached over a billion objects. The focus of the testing was on user retrieval response time, content storage throughput, and database maintenance. The goal is to demonstrate that Documentum 5 scales to a repository of a billion objects because it preserves good performance and can still be maintained.

Customer environments are unlikely to reach a billion objects until after many years of operation. The challenge for this test is that we needed to have a docbase setup with a billion objects within several weeks. The current max tested load rates using the standard Documentum benchmark loader (and hence ultimately the standard Oracle server) are in the vicinity of 12 million objects per day. For our Windows test server hardware the maximum rate was more on the order of 4 million per day. At that rate it would take about 8 months to load a billion objects. Coupled with the timing constraints for this test we opted to develop a special "bulk object loader" that would utilize the Oracle Direct Path loader to achieve maximum load rates. The idea behind this approach is to populate both the database and file system as if it had been done by the Documentum Content Server. Then boot a working repository from this environment and conduct the rest of the test. The process of getting the docbase setup with a billion objects is detailed further in Appendix B.

Most of the billion objects are of type "mail_doc". This is a custom type (sub-typed from dm_document) that is used to represent e-mails. The full type definition is provided in Appendix M. Although meant to primarily model the storage of a billion e-mails, this data type can also be used to represent images in an archive. In either case, this object definition implies that eight Content Server tables would grow as we added objects. The rest of the content server tables would remain fairly constant in size. The high growth tables and their relationship with each other are outlined in Figure 1 below. More details on the Documentum Object model can be found in Documentum Server Object Model reference.

We utilized database partitioning as a way to help ensure the enormous database associated with this Content repository was manageable and could deliver great performance. The tables associated with the billion objects were range partitioned by R_OBJECT_ID (10 million objects per partition). Only "local" indexes were used and this helped to ensure that any single partition-based maintenance operation could be completed with a few minutes. For this testing effort we timed various maintenance operations like index creation and statistics computation.

Once the billion objects had been loaded we measured the load rates using the standard Documentum API's and compared the load rates on the Content repository with a billion objects with that of one that had less than 100,000 objects on the same hardware platform. In this portion of the test we grew the number of parallel document loaders until the available hardware reached capacity. Unlike the earlier special "bulk" loading, this part of the test employed the standard Documentum API's to load the content and meta-data.

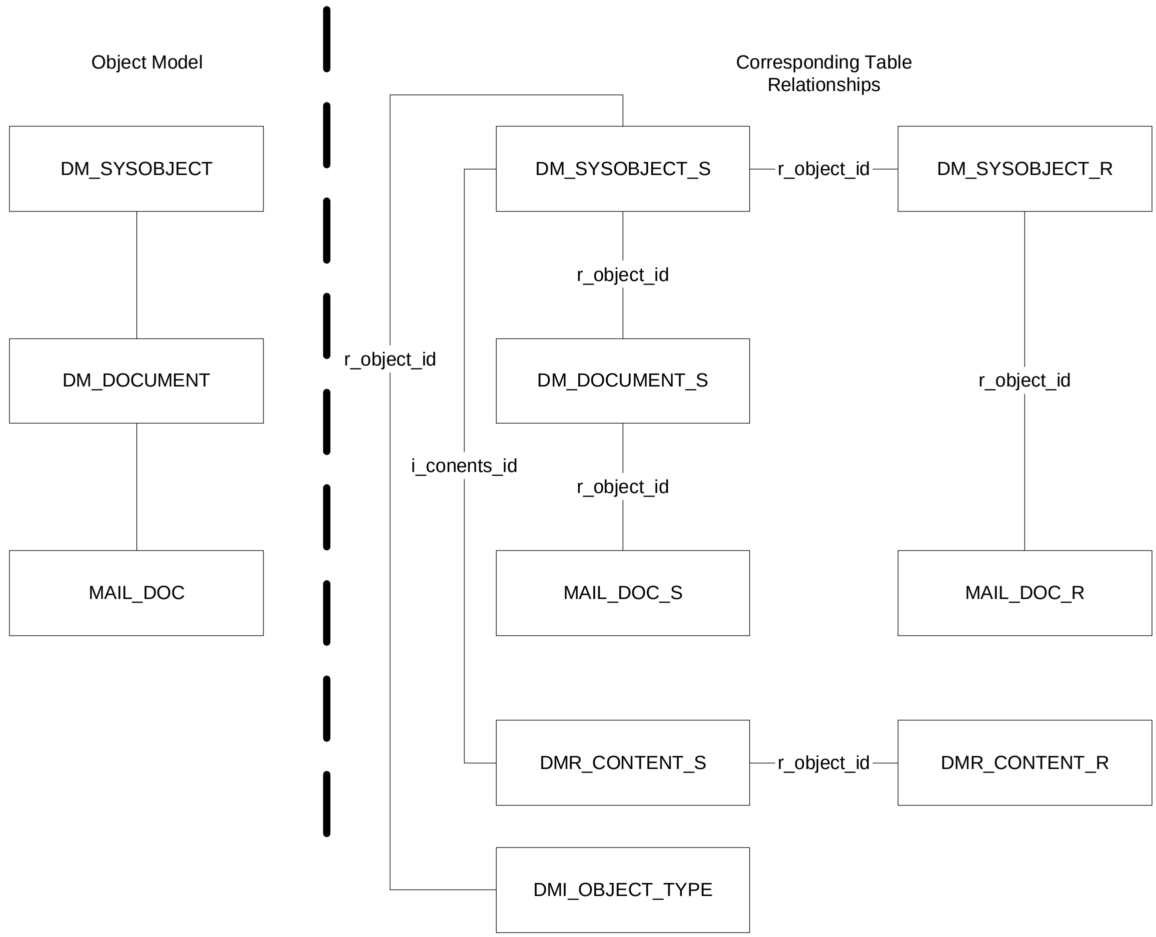


Figure 1: Object Model Outline and High Growth Database Tables

Finally, a multi-user workload was applied to verify that the resulting Content repository could be accessed with excellent response time for a large number of concurrent users. The multi-user benchmark used in these Documentum capacity tests was designed to simulate a workload of Documentum Webtop users that covers the primary components of the Documentum. These products include the Documentum Content® Server, the Web Development Kit (WDK), and the Documentum Foundation Classes.

This benchmark simulates various actions of a number of Documentum Webtop users issuing highly selective and medium selective search queries using a “customized” Webtop Advanced search component, followed by content retrieval. In both types of searches the primary search key is <u>not</u> a database partitioning key, and hence mirrors real customer environments.

The focus of the multi-user benchmark was user response time for their “advanced” searches. Folder navigation is not an extremely useful manner in which to locate content in a repository with a billion objects. In addition, searches need to be somewhat selective (i.e., initial RDBMS scans find few possible hits) else general performance will suffer and response time would be quite long. In the field of “imaging” such searches are typically the norm. Image lookups are driven by customer id and/or date or document type. In the e-mail archiving case, individual selective queries could occur during the process of retrieving the contents of an individual e-mail or range of e-mails generated on a specific day. The basic search / retrieval sequence for each user is outlined at a very high level in Figure 2.

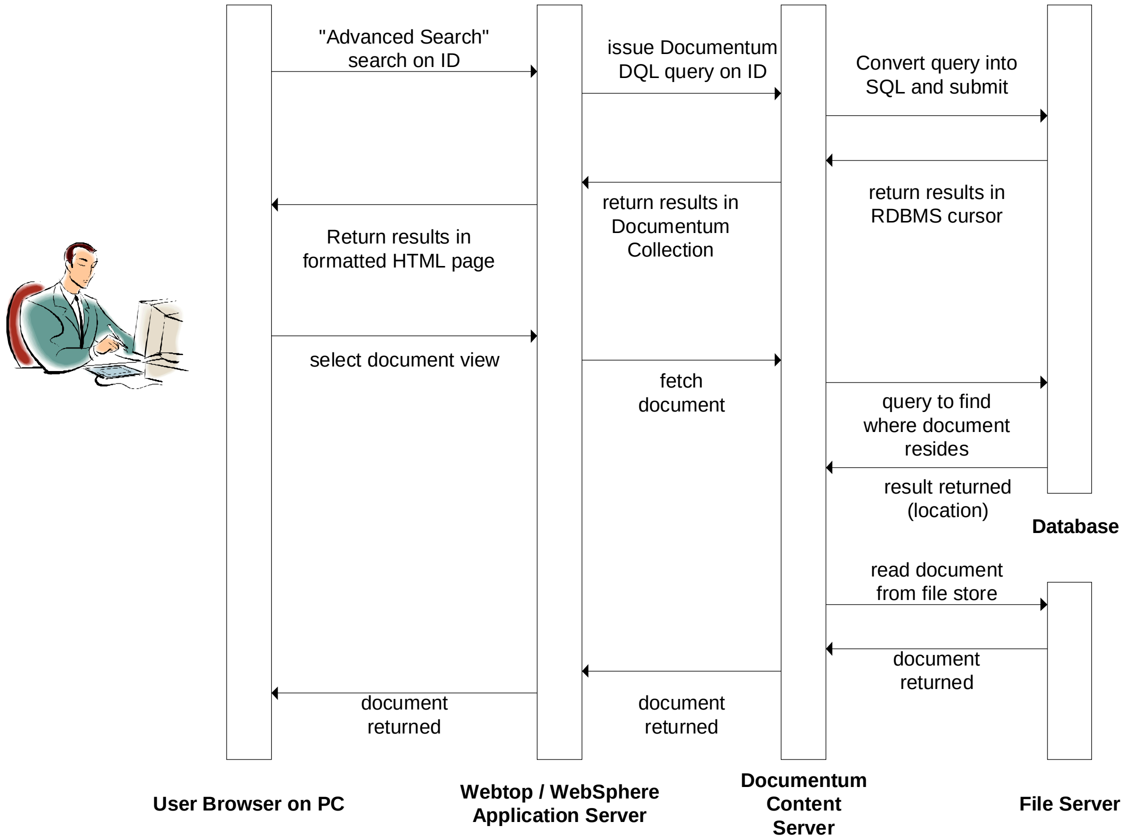


FIGURE 2: Multi-user test scenario

The primary metric for the multi-user tests are the number of concurrent Webtop users that can be supported with an average response time that does not exceed the specified limits. After the login each user performed 6 major operations at random times (5 selective lookup's on a unique ID and 1 content lookup on date). The measurement period is 30 minutes.

Each operation (or task) consists of several HTML screens that are dynamically generated from WebTop/WDK plus the content from the Docbase (e.g., WORD and Text files). Acceptable response time is, in general, considered to be no more than 1 to 4 seconds per screen. After factoring in the relative weights of all operations and the number of screens the average response time per screen is typically 1.5 seconds or less. These operations and their associated maximum average response times are outlined below.

Task	Number of screens	Total acceptable Average Response time (seconds)	Total acceptable response time per page (seconds)	Note
Login	1	4	4	This operation includes establishing the connections/sessions and ends with a Cabinet display for the user.
LOOKUP_BY_ID	5	8	1.6	Starting from cabinet display: 1. move to advanced search, 2. change drop downs to match mail_doc and advanced criteria. Initiate the search 3. view the search results, 4. view the content 5. display the cabinet view again
LOOKUP_BY_DATE_RANGE	5	8	1.6	Similar to the above except that the use queries on a date-driven search rather than a selective ID.

TABLE 1: LDB-Retrieval Response time requirements

The benchmark driver consists of multiple processes that send and receive HTTP messages in a way that simulates a browser. Each benchmark process handles at most 125 users and logged these on every 10 seconds (so 120 users are fully logged on in 20 minutes). So, for instance, a 400 light user test will have 400 users logged on in 15 minutes. A 1,000 concurrent user test will also have all 1,000 users logged on in 20 minutes.

Once a user logs on they start and end their work randomly throughout the measurement period. They will burst the HTTP associated with their high level "operation" and then sleep for a random period (negative exponential distribution with a mean of 60 seconds). The time between requests for a user affect resource consumption and response time due to session management techniques (pooling and inactivity timeouts).

Configuration

Hardware Configuration

The hardware used in these tests is shown below. The first configuration represents the hardware used in the multi-user tests. The second is for the throughput tests (Billion docbase vs. Billion_NOT). The hardware configuration for the Bulk loading phase is described in Appendix B.

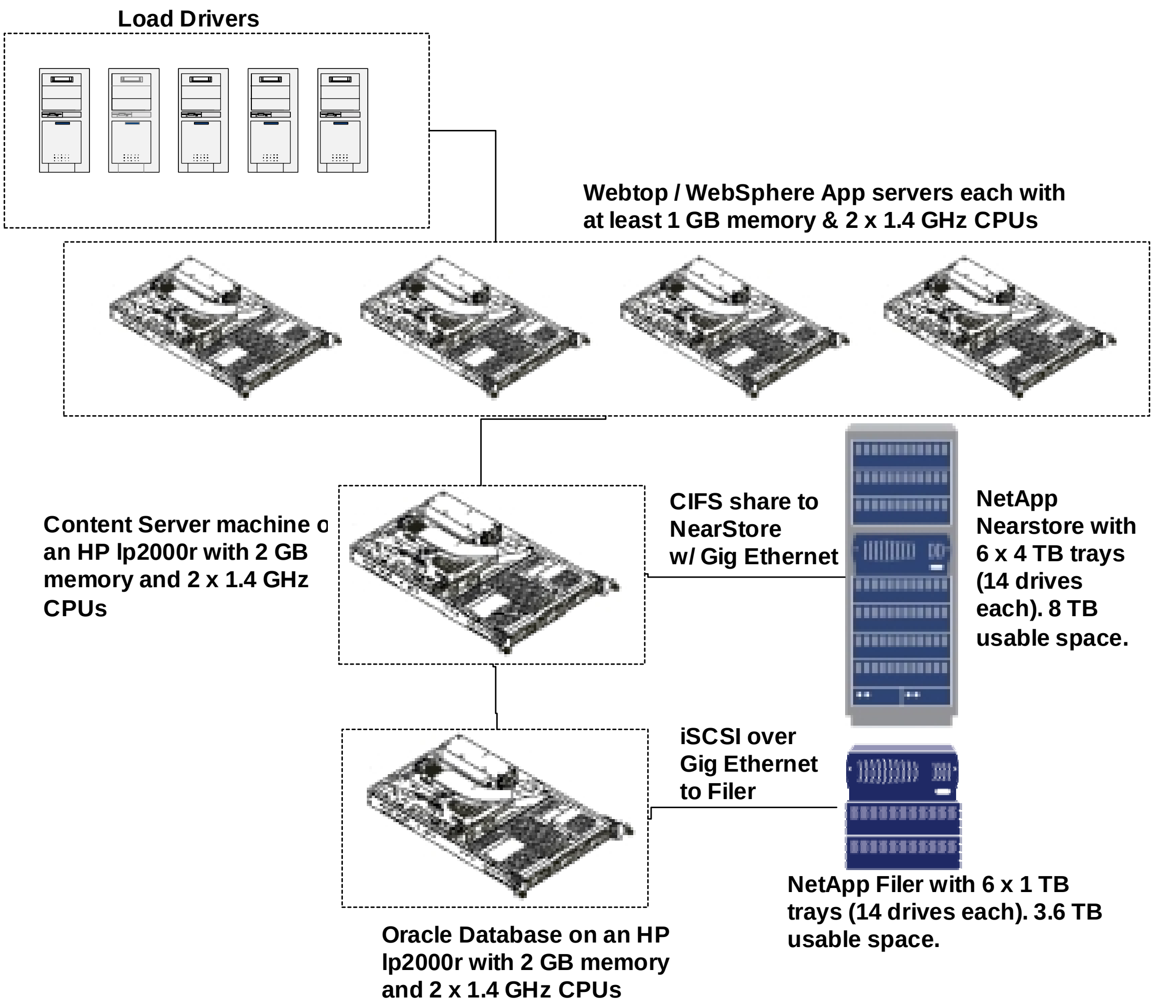


Figure 3: Hardware Configuration for Multi-User testing on Billion Objects

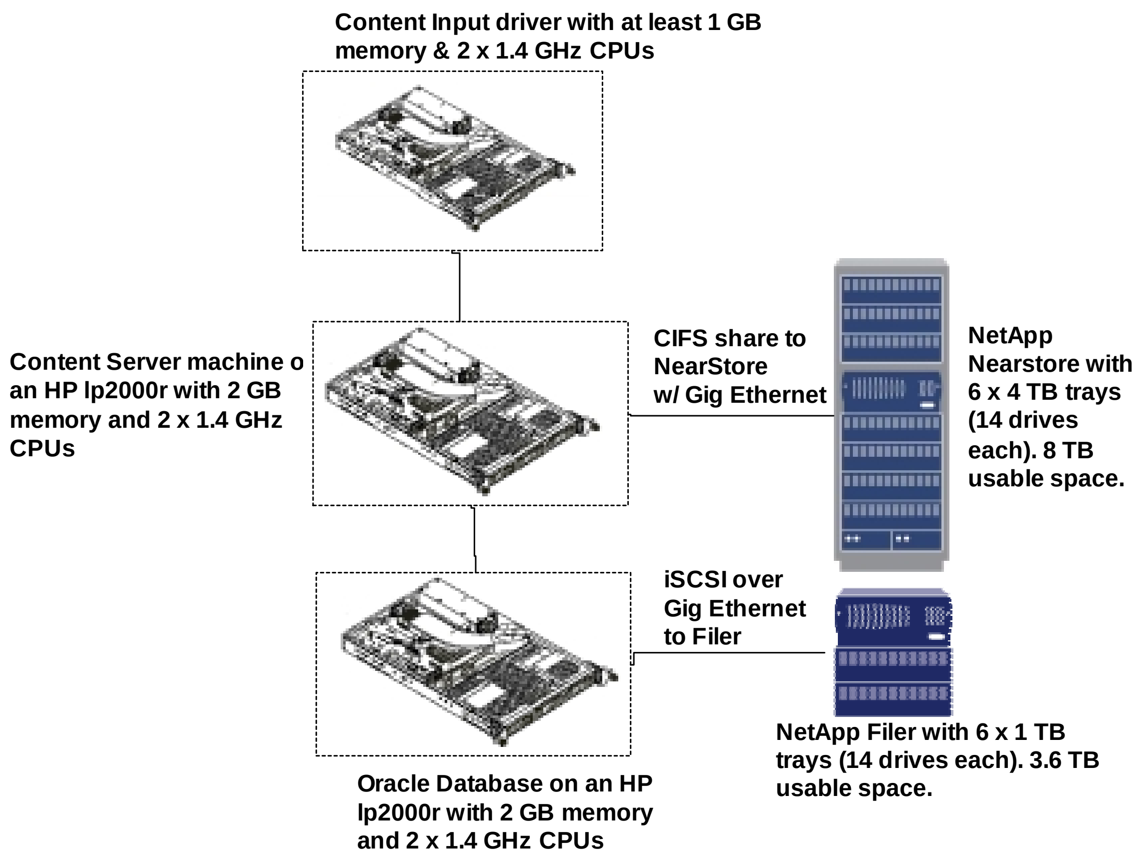
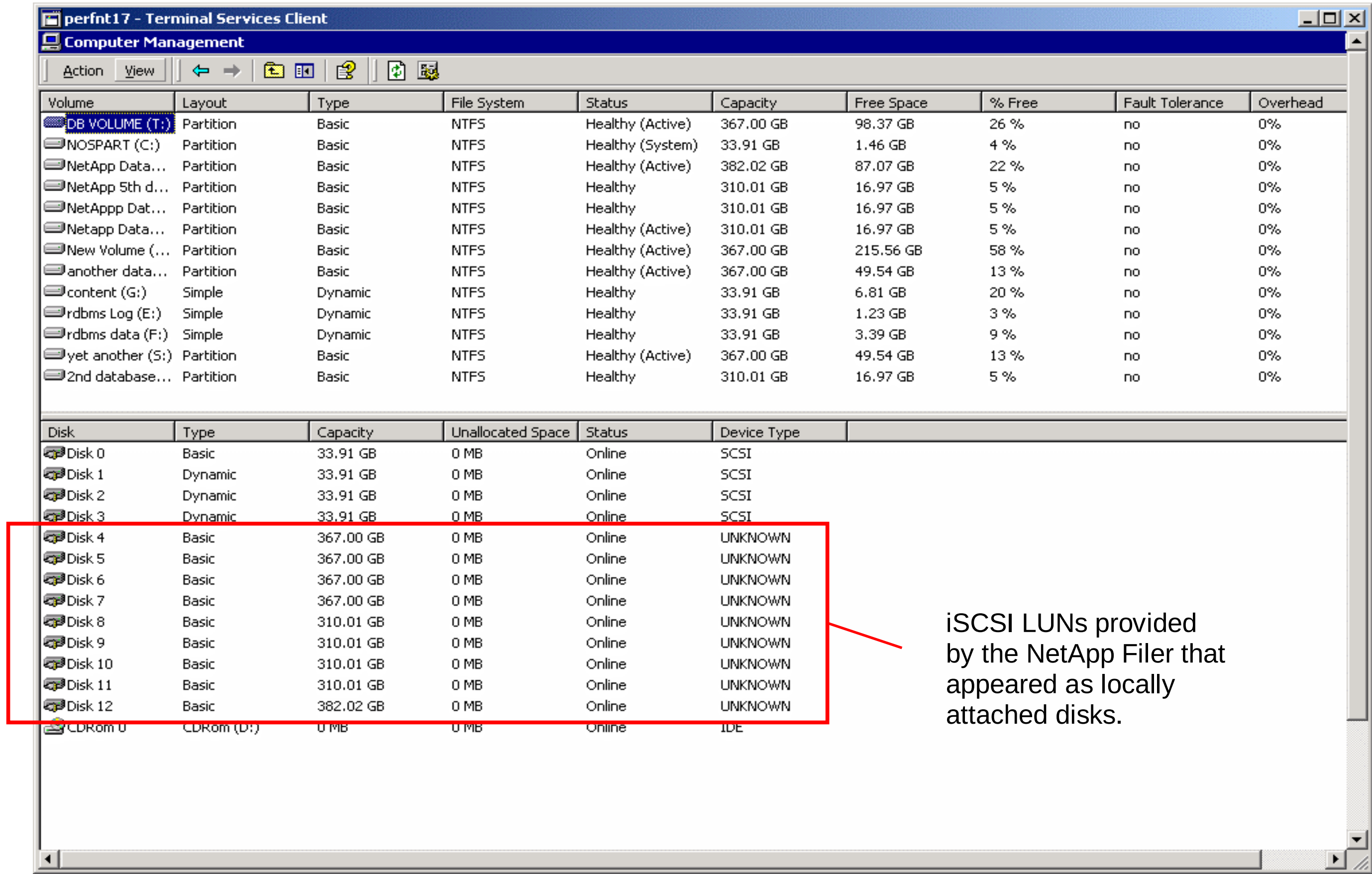


Figure 4: Hardware environment for the input load test portion of test

We used Intel/Windows servers and NetApp disk arrays for this test. All of the Webtop, Content Server and RDBMS server machines were either HP lp2000r's or lp1000r's. All are dual processor 1.4 GHz systems with 1 or 2 GB of memory.

The main database storage was a Netapp Filer with 6 drive trays at 1 TB raw space for each (about 600 GB usable). The database server machine connected to the drive array over iSCSI and a Gig bit Ethernet network. The total usable space was around 3.6 TB (of which we only used about 2.5 TB for the RDBMS). These iSCSI LUNs appeared to Windows as locally attached drives (as opposed to CIFS shares). This is illustrated in Figure 5. The windows driver for iSCSI was provided by Netapp and required Intel-based Gigbit Ethernet boards. The hardware configuration of the Netapp Filer is summarized in Appendix H.

The Filer did not appear at the start of the test with all of the disks configured. Drive storage was added over time. The process by which the database was expanded to operate with this additional drive space is described in more detail in Appendix D. One important "production" point about this process from a Windows perspective is that we had to reboot the database server machine each time additional drives were brought online.



perfnt17 - Terminal Services Client

Computer Management

Action View

Volume	Layout	Type	File System	Status	Capacity	Free Space	% Free	Fault Tolerance	Overhead
DB VOLUME (T:)	Partition	Basic	NTFS	Healthy (Active)	367.00 GB	98.37 GB	26 %	no	0%
NOSPART (C:)	Partition	Basic	NTFS	Healthy (System)	33.91 GB	1.46 GB	4 %	no	0%
NetApp Data...	Partition	Basic	NTFS	Healthy (Active)	382.02 GB	87.07 GB	22 %	no	0%
NetApp 5th d...	Partition	Basic	NTFS	Healthy	310.01 GB	16.97 GB	5 %	no	0%
NetAppp Dat...	Partition	Basic	NTFS	Healthy	310.01 GB	16.97 GB	5 %	no	0%
Netapp Data...	Partition	Basic	NTFS	Healthy (Active)	310.01 GB	16.97 GB	5 %	no	0%
New Volume (...	Partition	Basic	NTFS	Healthy (Active)	367.00 GB	215.56 GB	58 %	no	0%
another data...	Partition	Basic	NTFS	Healthy (Active)	367.00 GB	49.54 GB	13 %	no	0%
content (G:)	Simple	Dynamic	NTFS	Healthy	33.91 GB	6.81 GB	20 %	no	0%
rdbms Log (E:)	Simple	Dynamic	NTFS	Healthy	33.91 GB	1.23 GB	3 %	no	0%
rdbms data (F:)	Simple	Dynamic	NTFS	Healthy	33.91 GB	3.39 GB	9 %	no	0%
yet another (S:)	Partition	Basic	NTFS	Healthy (Active)	367.00 GB	49.54 GB	13 %	no	0%
2nd database...	Partition	Basic	NTFS	Healthy	310.01 GB	16.97 GB	5 %	no	0%

Disk	Type	Capacity	Unallocated Space	Status	Device Type
Disk 0	Basic	33.91 GB	0 MB	Online	SCSI
Disk 1	Dynamic	33.91 GB	0 MB	Online	SCSI
Disk 2	Dynamic	33.91 GB	0 MB	Online	SCSI
Disk 3	Dynamic	33.91 GB	0 MB	Online	SCSI
Disk 4	Basic	367.00 GB	0 MB	Online	UNKNOWN
Disk 5	Basic	367.00 GB	0 MB	Online	UNKNOWN
Disk 6	Basic	367.00 GB	0 MB	Online	UNKNOWN
Disk 7	Basic	367.00 GB	0 MB	Online	UNKNOWN
Disk 8	Basic	310.01 GB	0 MB	Online	UNKNOWN
Disk 9	Basic	310.01 GB	0 MB	Online	UNKNOWN
Disk 10	Basic	310.01 GB	0 MB	Online	UNKNOWN
Disk 11	Basic	310.01 GB	0 MB	Online	UNKNOWN
Disk 12	Basic	382.02 GB	0 MB	Online	UNKNOWN
CDRom 0	CDRom (D:)	0 MB	0 MB	Online	IDE

FIGURE 5: iSCSI LUNs view from the Windows system.

The main content storage system was a NetApp Nearstore. This had about 58 drives and 8 TB of usable space (of which we only used about 4 TB). Four 2TB volumes were created and shared via CIFS. Each volume had 14 drives with one of the 14's capacity reserved for parity (Netapp employs a modified RAID 4 for data integrity). Access from the Content Server machine to the Nearstore was accomplished with the CIFS volume over Gig Ethernet. The Nearstore essentially acted as a file server. Its hardware configuration is summarized in Appendix I.

Software Versions

Windows 2000 SP4

Documentum Content Server 5.2

Documentum WebTop 5.2

Documentum WDK 5.2

Documentum DFC 5.2

Documentum DMCL 5.2

Oracle 9.2 with service pack 4

IBM WebSphere 5.0.1 with APAR PQ76313.

Operating System Configuration

2000 user logins were configured for the Windows domain and each had the same prefix (user). Each login name had a number tagged on the end (e.g., user1 and user500) ranging from 1 to the total user count for that class of user. Each had the same password.

The content server and data base machines each had a separate Gig Ethernet board. Static IP addresses were configured for those interfaces as well as for the Nearstore and Filer for that network. The database server machine was configured with the Windows iSCSI driver.

DBMS Configuration

Oracle's partitioning capability was key to the success of this test. Oracle provides several different types of partitioning in 9.2, we chose Range partitioning for this test. The partitioning key was R_OBJECT_ID and eight tables were redefined as partitioned: DM_SYSOBJECT_S, DM_SYSOBJECT_R, DM_DOCUMENT_S, MAIL_DOC_S, MAIL_DOC_R, DMR_CONTENT_S, DMR_CONENT_R, and DMI_OBJECT_TYPE. Each partition was sized to hold 10,000,000 objects (an estimated 25G bytes in size including indexes) except for the initial partition and the final "maxvalue" partition (this is illustrated for DM_DOCUMENT_S below). Hence, to hold a billion objects over 103 partitions were created for these tables. The above tables were converted on an existing docbase by renaming them, recreating the tables and indexes to support partitioning, and then copying back the original data to the new table definitions. The procedure and scripts for this part of the process are summarized in Appendix C.

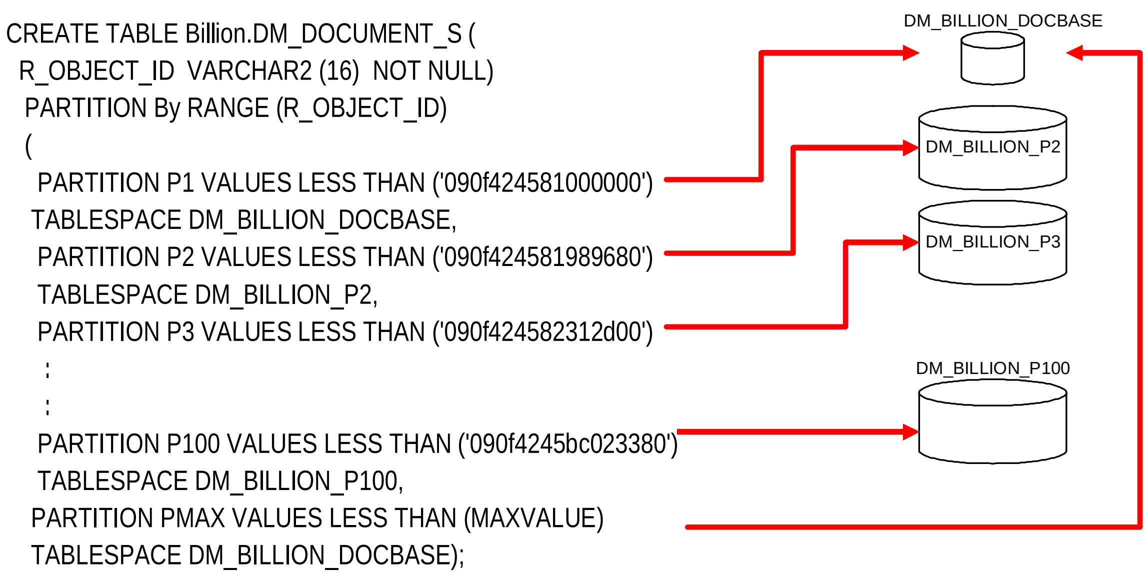


Figure 6: Conceptual Diagram of range partition based on R_OBJECT_ID

All partitions with the same basic range were mapped to the same table spaces. Hence, all of these objects' "meta-data" were located on the same table spaces. Despite this, Documentum does <u>not</u> support dropping objects en-masse by dropping a partition. This is because more complex applications cannot guarantee the locality of an object's meta-data to a particular partition.

In production environments the inserts, updates, deletes, and queries to these partitioned tables would focus on a single partition. Imaging applications, in particular, query on selective keywords (e.g., policy id or customer id) and e-mail applications filter on user name and date. This will typically focus most of the database activity to one partition (and its corresponding disks) at a time. This is illustrated in Figure 7.

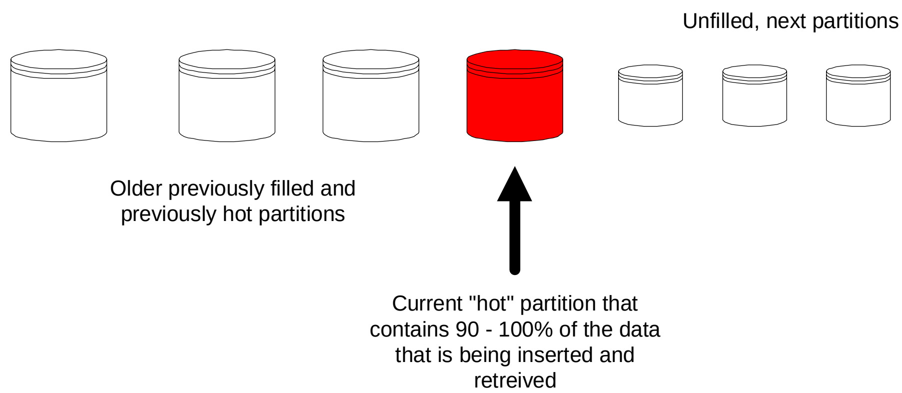


Figure 7: Range partitions

This locality of access becomes a significant advantage in that index and statistics maintenance operations on "older" partitions all but cease eventually as the users and input loaders focus on the latest data.

Range partitioned tables are less complicated to maintain if all of the partitions are created up-front. However, conditions arise that require additional partitions to be created. In this study we actually split the last partition for the large tables to add two additional partitions. The procedure for doing this and some corresponding notes are provided in Appendix E.

We also performed a few experiments with hash partitioning. As long as the there are a power-of-two number of partitions, then the Documentum R_OBJECT_ID would be mapped equally across all partitions. In practice such an approach would spread the inserts, updates, and selects across all partitions (and hence across all of the drives). Database growth, however, would have to be handled differently. In hash partitioning the underlying partition table space is likely to consist of multiple underlying data files spread over multiple volumes. All partitions would have to grow in the same fashion. This would, over time, still localize the data access to the most recently online storage tray.

Hash partition table definition and maintenance is less complicated than range partitioning. There is no need to specify object id's or track them. Simplicity is an important aspect to a database with a long production lifetime.

It would also, theoretically, require less data sharing by an Oracle RAC implementation (however, we have not, to this date, tested this). Based on these two issues (simplicity and improved RAC support) we recommend that customers employ hash partitioning rather than range partitioning.

However, the index maintenance and statistics maintenance must span over all partitions (because the current data is spread over all partitions). This can be mitigated somewhat by additional processors and disks.

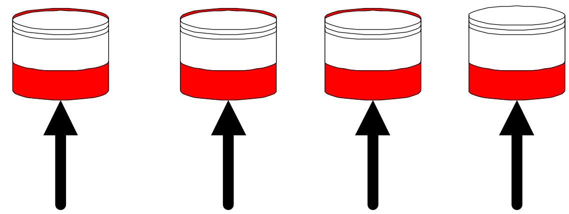


Figure 8: Even spread with Hash partitions

The Oracle configuration file (init.ora) is shown in Appendix A with the Oracle database creation scripts in Appendix B. A couple of extremely noteworthy parameters include the following:

```
cursor_sharing = FORCE
optimizer_mode = CHOOSE
optimizer_index_cost_adj = 5
optimizer_index_caching = 95
```

When running with the cost-based optimizer (optimizer_mode=CHOOSE) Oracle needs to have the subsequent two parameters (optimizer_index_cost_adj and optimizer_index_caching) set else some poor query plans will result.

Documentum Content Server Configuration

The number of concurrent sessions was set to 1000. The client session timeout was left to the default. In addition, the content server jobs were disabled for this testing effort (including the FullTextMgr). Testing with a billion full text indexed objects was beyond the scope of this effort.

There are several important points to summarize about the setup of the Documentum Content Server with partitioning:

1. the Server Docbase creation does not create partitioned tables. These tables need to be created from an existing Docbase as outlined in Appendix C.

2. the Server Index creation methods do not use the Keywords GLOBAL or LOCAL or any partitioning options. These index options must be defined using the native database commands.

3. The default index update job does not support partitioned-based statistics updates (as outlined in Appendix G). A special job should be created for this.

4. We recommend running reporting jobs like the dm_StateOfDocbase infrequently, because even the process of counting the number of objects could impact online users.

Documentum Webtop/WDK Server Configuration

The dmcl.ini enabled session pooling.

The out-of-the-box advanced search component of the WDK 5.2 will convert string search arguments to lower case so that the searches will be case insensitive. This would lead to unacceptable performance for our search-and-view users and hence the advanced search component was customized to suppress this behavior[1]. This involved creating a "customized" AdvancedSearch.class and deploy it at:

`webtop.war\WEB-INF\classes\com\documentum\custom\library\advsearch`

where webtop.war is located (for example) in:

`C:\WebSphere\AppServer\installedApps\`*`host`*`\webtop.ear\`

The code would essentially take the query formulated by the advanced search component screen and strip out the calls to the database function lower() in the WHERE clause

Also, another class `TimeExpression.class` was added to:
`webtop.war\WEB-INF\classes\com\documentum\custom\formext\docbase`

to help support the ability to search down to "seconds" resolution. The load rate for objects was so high that even in a single second hundreds of objects could be created.

In addition, some of the configuration files for the Webtop were modified so that additional attributes were searchable and that the mail_doc type was searchable. These files included:

`SearchNlsProp.properties` located in
`webtop.war\custom\strings\com\documentum\custom\library\search`

`advsearch_component.xml` located in
`webtop.war\custom\config`

`advsearch.jsp` located in
`webtop.war\custom\library\advancedsearch`

All of these changes are further detailed in Appendix N. The screens seen by the users are detailed in Appendix O.

Aside from these customizations each application server was given 775M to 1GB of memory to run the concurrent users.

[1] An alternative approach to Webtop customization would be to define a functional index in Oracle.

Multi-User Test Results

1000 concurrent LDB-Retrieval-1 Users on a multi-server configuration

Operation Name	Average Operation Response Time(secs)	Average Screen Response Time(secs)	Max Operation Response Time(secs)	Min Operation Response Time(secs)	Total Operation Count
LOGIN	2.73	2.73	13	1	1000
LOOKUP_BY_ID	6.12	1.22	21	3	5000
LOOKUP_BY_DATE_RANGE	7.94	1.59	27	2	1000
Total ops					7,000

Single-User Query Observations

The above excellent response times were made possible by the underlying data model's ability to leverage database partitioning. This section will examine how this works.

The following query is an internal one issued by the Content Server in order to locate content for a dm_sysobject by finding its corresponding dmr_content object.

```
SELECT * FROM dmr_content_rv BJ_ , dmr_content_sv AJ_ WHERE (AJ_.R_OBJECT_ID=:handle AND AJ_.R_OBJECT_ID=BJ_.R_OBJECT_ID) ORDER BY BJ_.R_OBJECT_ID,BJ_.I_POSITION
```

This is a "point query" (returns a single row) on DMR_CONTENT_S/R that performs roughly 2.5 disk IO's, 6 buffer accesses, and returns in 27 msecs. This excellent performance is made possible by the fact that R_OBJECT_ID is a partition key for these tables. The underlying database server can take advantage of this and zero in on the partition that contains the desired row. This is illustrated in Figure 9 and the query plan shown below:

```
Rows     Row Source Operation
-------  ---------------------------------------------------
      1  SORT ORDER BY
      1   NESTED LOOPS
      1    PARTITION RANGE SINGLE PARTITION: KEY KEY
      1     TABLE ACCESS BY LOCAL INDEX ROWID DMR_CONTENT_S PARTITION: KEY KEY
      1      INDEX UNIQUE SCAN D_1F0F42458000015F PARTITION: KEY KEY
      1    PARTITION RANGE SINGLE PARTITION: KEY KEY
      1     TABLE ACCESS BY LOCAL INDEX ROWID DMR_CONTENT_R PARTITION: KEY KEY
      1      INDEX RANGE SCAN D_1F0F424580000160 PARTITION: KEY KEY
```

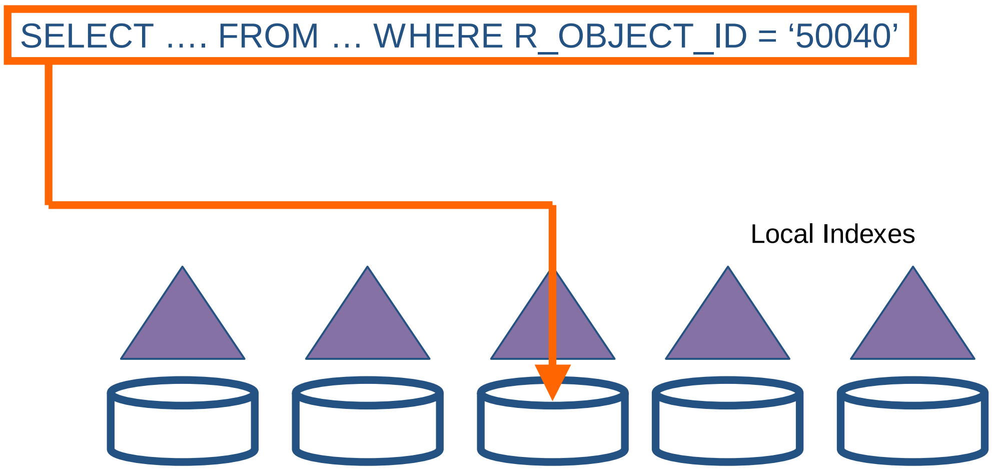


Figure 9: Query on partition key can zero-in on the desired partition for fast lookup

However, most user queries are not primarily driven by lookup’s on R_OBJECT_ID. They must utilize some other attribute (e.g., like OBJECT_NAME). All of the other indexes defined for this test were “local” (one copy for every partition to cover the rows associated with it) and “non-prefixed” (not on a partition key). A lookup on these indexes requires a search of every partition in order to locate the desired rows because the database cannot tell a head of time which partition has the desired row. This is illustrated below.

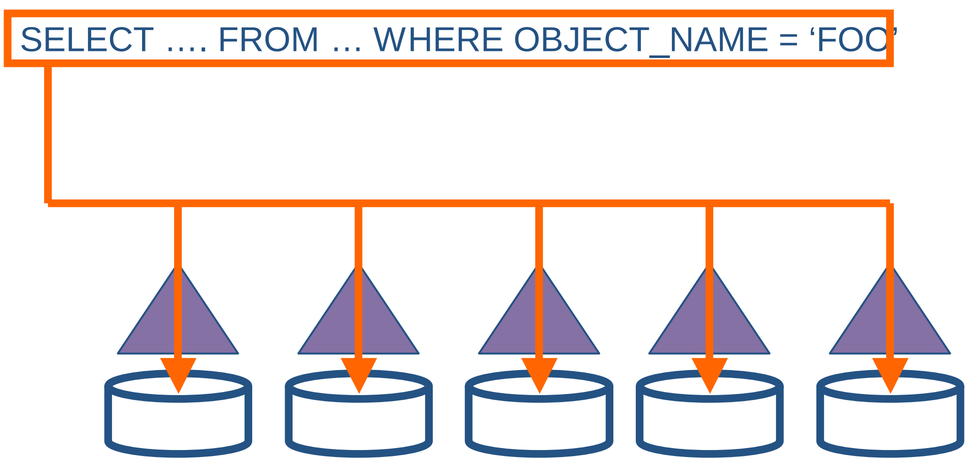


Figure 10: A query on a non-prefix local index

The multi-user benchmark had two major user-initiated queries that were like this. The first was one done on i_chronicle_id and the second on r_creation_date. The one on i_chronicle_id represented a query on a highly selective ID field (like a customer id).

```
SELECT ALL . . . . ,r_object_id,object_name . . .  FROM mail_doc
WHERE i_chronicle_id = '090f4245b24ba44b' ORDER BY  4 ASC,3 ASC
```

In this test the i_chronicle_id actually matched the r_object_id, but the database is not aware of this. Hence, the initial part of its query must search each partition. Despite this fact, the response time for this query was 105 msecs (as seen by Documentum). The underlying Oracle database did 7 disk I/Os, 334 buffer accesses, and returned the row in around 90 msecs. This excellent performance in the search of a billion records is possible because after the initial row is located (using the non-prefix index) all subsequent lookups in the join utilize the prefix R_OBJECT_ID indexes (the join column). This is illustrated below and in the subsequent query plan.

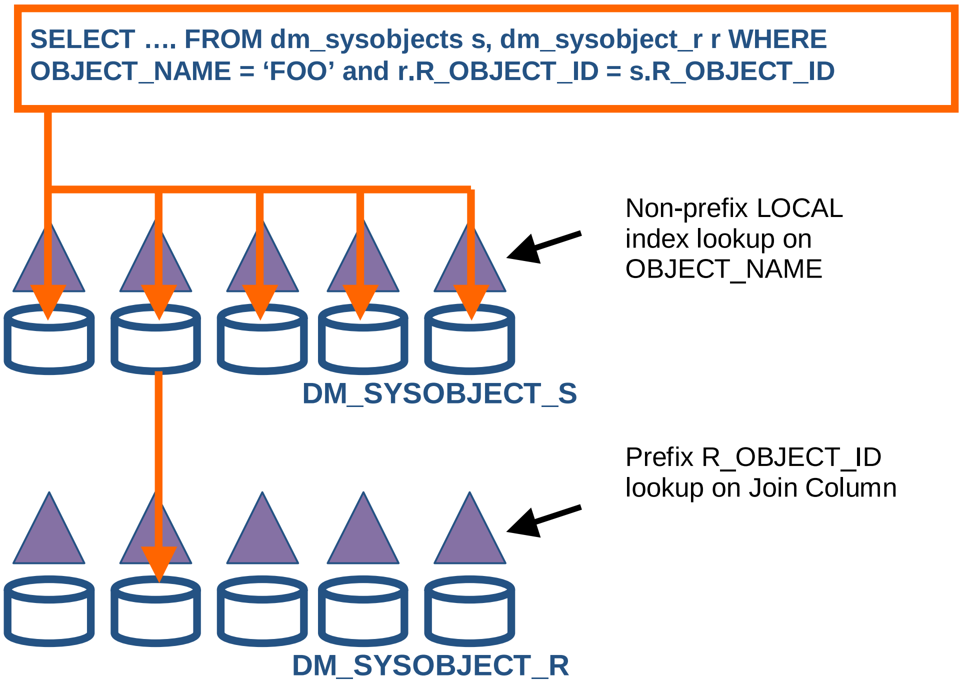


Figure 11: Query from non-prefix index and followed by prefixed lookups on join column

Note from the query plan that it is iterating over all 103 partitions to find the DM_SYSOBJECT_S record for the driving portion of the nested loop join, and then this is followed up by a prefix lookup on DM_SYSOBJECT_R.

Driving scan searches all partitions

Subsequent ones hit only single partition

```
Rows     Row Source Operation
-------  ---------------------
      2  SORT ORDER BY
      2   FILTER
      2    NESTED LOOPS
      2     NESTED LOOPS
      2      NESTED LOOPS
      1       PARTITION RANGE ALL PARTITION: 1 103
      1        TABLE ACCESS BY LOCAL INDEX ROWID DM_SYSOBJECT_S PARTITION: 1 103
      1         INDEX RANGE SCAN D_1F0F42458000000E PARTITION: 1 103
      2       PARTITION RANGE ITERATOR PARTITION: KEY KEY
      2        TABLE ACCESS BY LOCAL INDEX ROWID DM_SYSOBJECT_R PARTITION: KEY KEY
      2         INDEX RANGE SCAN D_1F0F424580000109 PARTITION: KEY KEY
      2      PARTITION RANGE ITERATOR PARTITION: KEY KEY
      2       INDEX UNIQUE SCAN D_1F0F424580000500 PARTITION: KEY KEY
      2     PARTITION RANGE ITERATOR PARTITION: KEY KEY
      2      INDEX UNIQUE SCAN D_1F0F424580000501 PARTITION: KEY KEY
      1    NESTED LOOPS
      1     TABLE ACCESS BY INDEX ROWID DM_ACL_S
      1      INDEX UNIQUE SCAN D_1F0F424580000103
      1     TABLE ACCESS BY INDEX ROWID DM_ACL_R
      3      INDEX RANGE SCAN D_1F0F424580000102
```

The final query (the range one on r_creation_date) returned on average 260 rows. This cost many more buffer accesses and disk I/Os and an ultimate response time of 1.5 seconds.

Not all of the queries were optimized correctly during this test by Oracle. The resulting response time for those queries could ultimately be many minutes. However, Documentum allows developers to leverage query hints as a mechanism to improve performance.

For example, the following query:

```
 SELECT r_is_virtual_doc, r_link_cnt,, …… r_modify_date FROM mail_doc where
object_name = 'HHHH915784' order by 4 asc, 3 asc
```

Had its response time substantially improved when changed to:

```
SELECT r_is_virtual_doc, r_link_cnt,, …… r_modify_date FROM mail_doc where
object_name = 'HHHH915784' order by 4 asc, 3 asc ENABLE (FORCE_ORDER)
```

This DQL hint FORCE_ORDER gets passed down to the underlying Oracle query as /*+ ORDERED */ and had a substantial impact on query performance.

Using Query Hint ENABLE(FORCE_ORDER)	Query Response in seconds	Disk I/Os	CPU seconds consumed
no	130	503,933	54
yes	3	285	0.14

TABLE 2: Impact of ORDERED query hint on DQL query by Object Name.

In addition, due to how we updated statistics (see Appendix G) some initial optimization problems occurred because the "global" statistics were out of sync with the more accurate "local statistics". This was resolved by purging the global statistics for that table, enabling Oracle to only use the more accurate local ones.[2]

Also, we setup a smaller set of documents (60,000+ of type-SOP) and compared query performance against those documents on the Billion object docbase vs. a smaller Docbase that only had those Documents. This simulates a consolidated environment in which a smaller (less performance scrutinized) system is merged with a larger, high performance online repository. The most important observation about that experiment was that some of the "small system" query plans changed so that they took a little more time to execute (in one case 100 msecs to 800 msecs) due to a query plan change that happened on the larger Docbase. This implies that extra care must be taken for the smaller application if it shares the repository space of the larger one to ensure that its queries are properly optimized. Given its origins, that level of scrutiny might not have occurred previously.

All of these anomalies re-enforce the need for developers to pre-test their production queries on a database with a significant number of objects. It is unlikely that a billion objects are required for this test environment; 10 to 20 million objects would suffice. Documentum provides a free benchmark loader on the developer website that can be useful for this purpose.

Finally, during the testing it became apparent that the Database I/O and CPU quota features are quite essential for production environments. It may not be easy to stop users from issuing poorly optimized queries. These queries could take hours to run and could impact online response. The CPU and I/O quota's would allow such run-away queries to be terminated automatically. Oracle

[2] Special thanks to Kevin Kincaid of USAA for resolving this one quickly during a review of the Test.

9.2 provides for a rich set of quota options including CPU_PER_CALL and LOGICAL_READS_PER_CALL.

Multi-User Test Observations

Hardware Summary and Operating System Summary for Multi-User Tests

The mean and normalized CPU metrics for the test are shown below. The largest consumer of CPU was the Webtop / Websphere interface. This was expected and is why that tier was allocated a total of eight CPUs.

Tier	CPU (secs) per reference WCM Op measured	Measured CPU (Min)	% of CPU measured per server (mean)
Oracle	0.16	19	40%
Content Server	0.04	5	8%
Webtop / WebSphere	0.49	58	35%
Total:	0.69	82	

TABLE 3: CPU profile for the 1000 users per 30 min test

The Content Server machine was about 8 - 12% busy during the test.. On each Webtop / WebSphere machine the CPU utilization ranged from 30 to 50%. The database server was around 40% (but peaked at almost 60% see below). Supporting more than 1,000 users with this workload would have required likely an additional Webtier system and more processors on the Database server side.

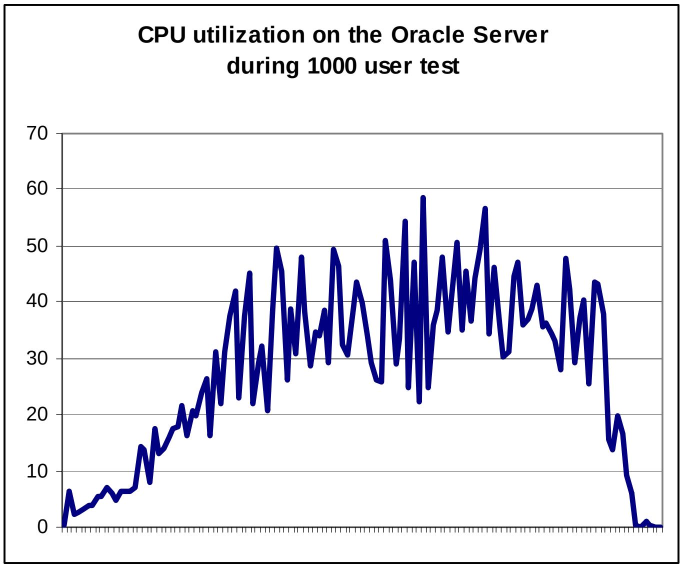


FIGURE 12: Peak CPU usage vs. Total available CPU capacity

The disk transfers for the database server are shown below. Although the average number of disk I/Os was about 200 per second the peak exceeded 500 per second. This I/O was concentrated over the drive volumes that had the partitions being hit during the test. The users queried a range of object id's that spanned about 30 million.

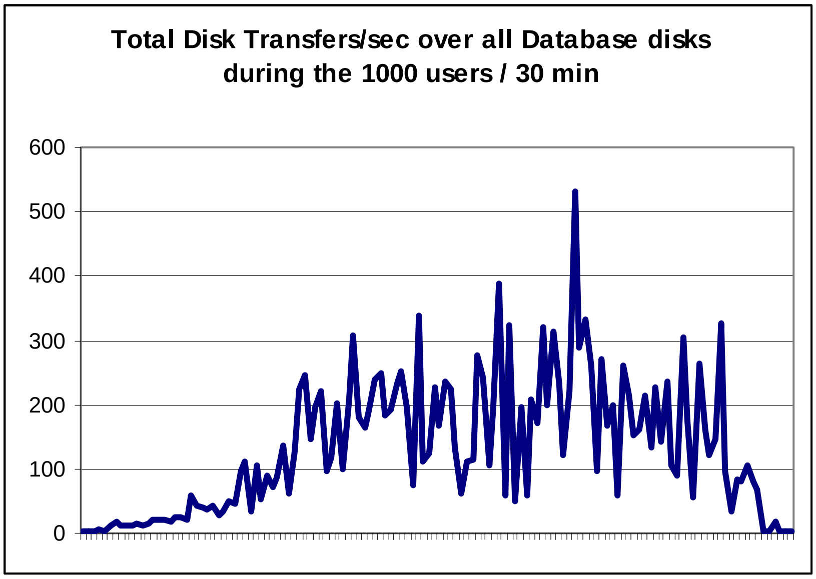


FIGURE 13: Disk writes/sec for on Database server during 1000 users/ 30 min test

DBMS Summary for Multi-User Tests

Most of the important observations about the database relative to querying have been already made. The only additional point is that during the multi-user run the Documentum connection pooling allowed for there to only be slightly more than 100 Oracle sessions to support the 1000 users. The Oracle user connections during the 1000 user run are shown below.

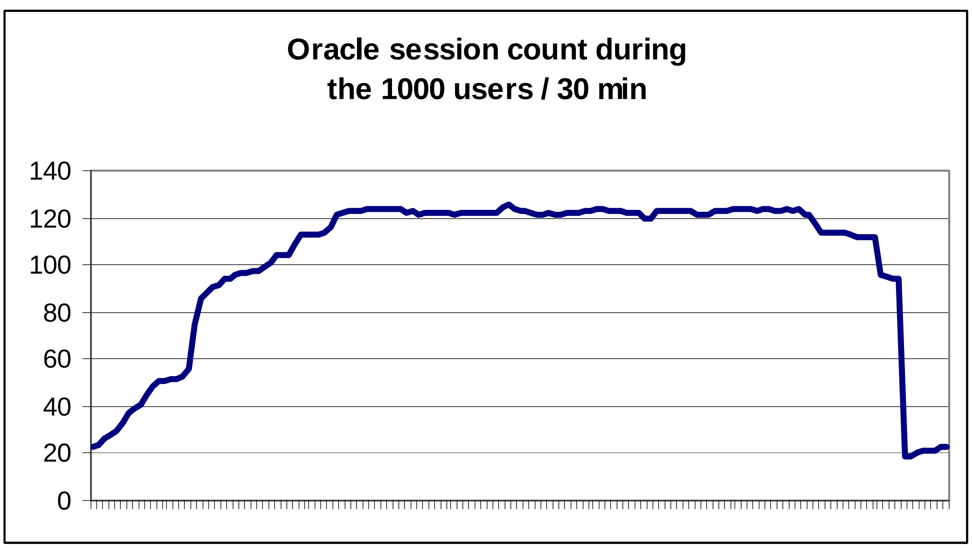


FIGURE 14: Oracle user connections during the 1000 user run (less than 30 min)

Content Input Test Results

The two tables below show the performance metrics for loading 1 to 10 parallel loaders into both the Billion object docbase and the Billion_NOT one. In each test every loader stored the same number of objects (1,000).

Content Input Statistics for Billion Docbase

Number of parallel loaders	Load Throughput (objs per hour)	Number total of documents loaded	Average transaction time (msec)
1	31,920	1000	112
2	67,380	2000	106
5	125,880	5000	142
10	160,000	10000	224

Content Input Statistics for Billion_NOT Docbase

Number of parallel loaders	Load Throughput (objs per hour)	Number total of documents loaded	Average transaction time
1	36,600	1000	98
2	73,080	2000	98
5	131,460	5000	136
10	165,360	10000	216

In all cases the loading into Billion_NOT was faster than into the much larger docbase, however, the difference was slight. This is illustrated in Figure 15 below as the difference ranges from 15% (14 msec difference) to 4% (8 msecs). This demonstrates the point that the performance of the system degrades only slightly as even though the total number of objects grew to a billion.

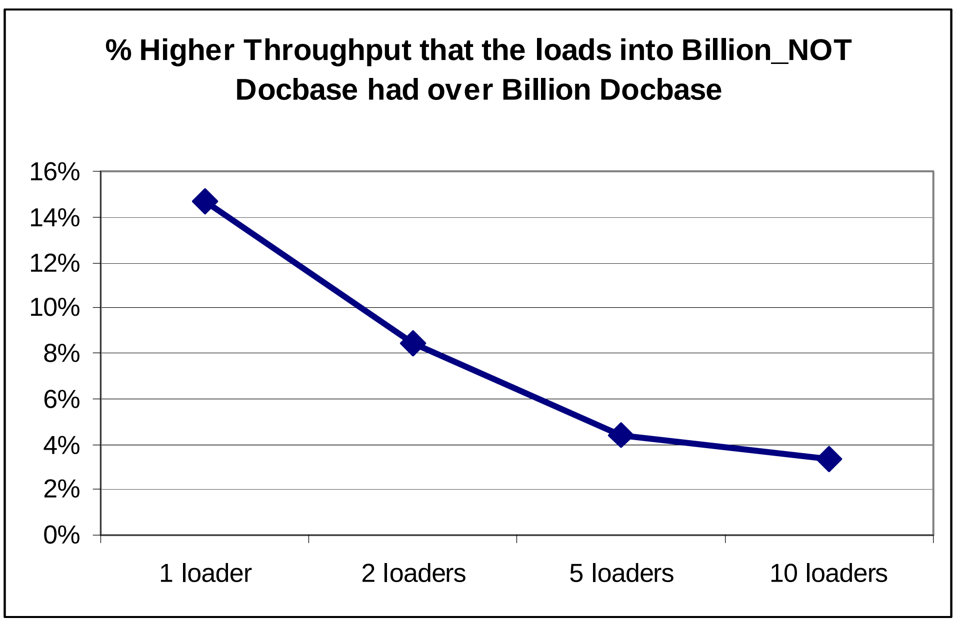


Figure 15: Illustration of slight degradation caused by the storage of a billion objects

Hardware Summary & Operating System Summary for Content Input Tests

Tier	CPU (secs) per reference WCM Op measured	Measured CPU (Min)	% of CPU measured per server (mean)
Oracle	0.031	330	70
Content Server	0.027	269	60
Total:	0.058	599	

TABLE 9: CPU profile for the 10 loader test

. The database server machine was the chief bottleneck in these load tests. The average CPU utilization peaked at 70% for 10 parallel loaders. This is shown in Figure 16 below.

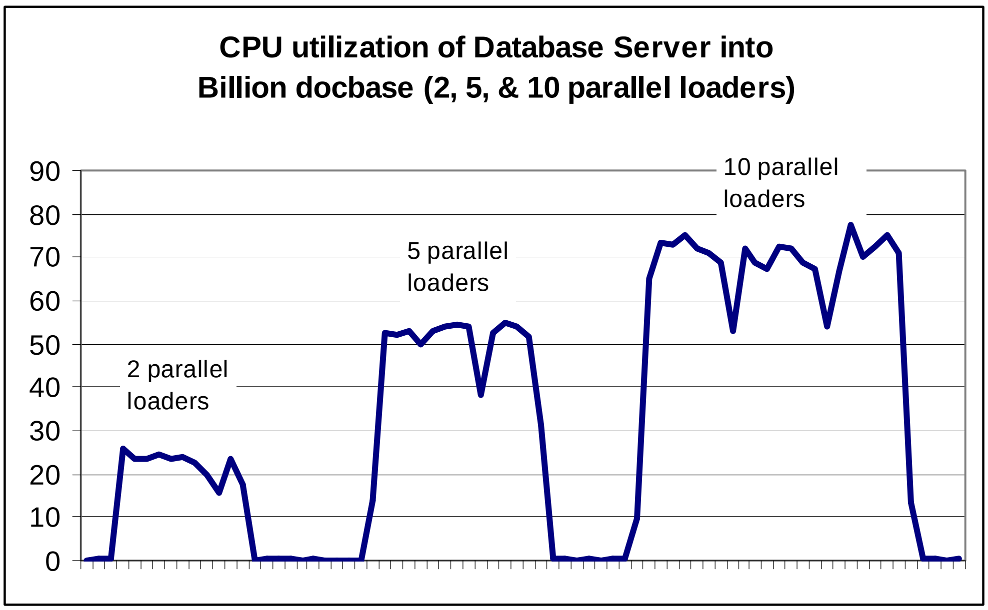


Figure 16: CPU utilization of database server on parallel Loader tests

However, the Content Server consumption followed closely behind this (see Figure 17). The Content Server CPU consumption, unlike that of the database server, in this type of workload is influenced by the size of the content. The documents were 80K in size. Smaller documents would have consumed less CPU on the Content Server side.

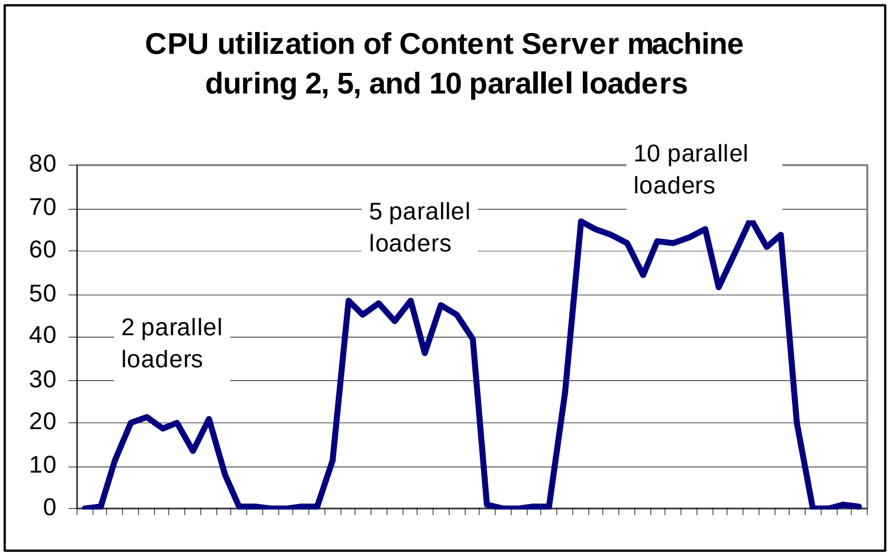


Figure 17: Content Server CPU utilization on Parallel loader test

As far as we could tell the disk arrays were not the bottleneck in the loading. The network between the hosts and the disk arrays was at 1000M bits. The activity of the Filer was small relative to that of the Nearstore. The CPU utilization of the Nearstore is shown below. The disk byte throughput for the Nearstore is shown in Figure 19. We could not tell, however, how many separate I/Os were occurring during the tests. However, this load rate was so much smaller than the rate achieved during the initial bulk loading phase, it is unlikely that the drives were the bottleneck for the test.

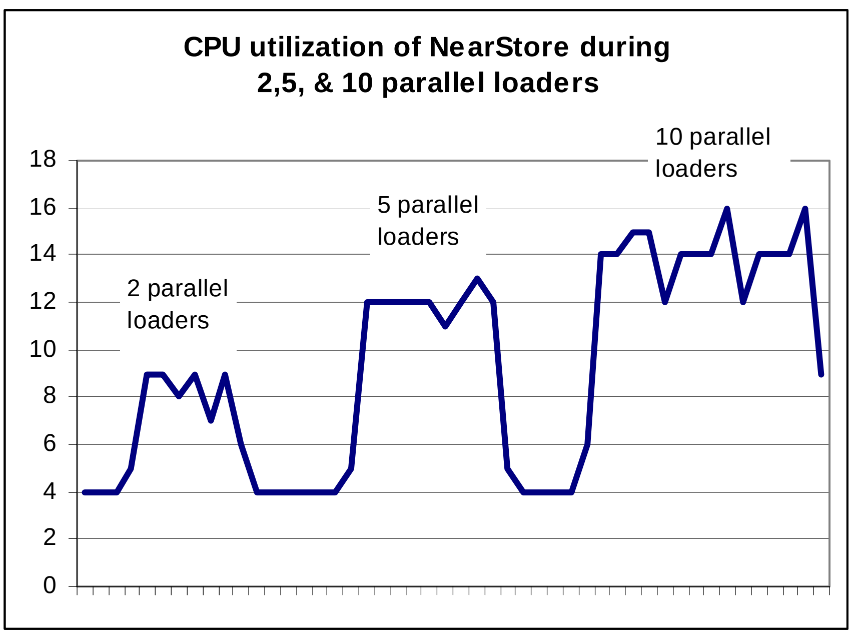


Figure 18: Nearstore CPU utilization for some of the parallel tests

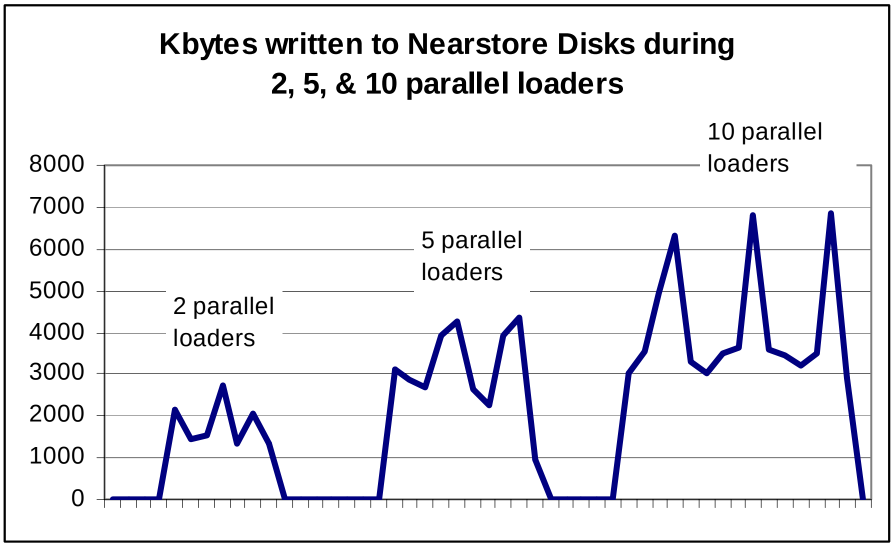


Figure 19: Disk activity for Nearstore on parallel loader test

Some Database Maintenance Metrics

Operation	Time	Notes
Rebuild 22 indexes for 10 million objects	60 min	Single Partition Parallel degree = 4 NO LOGGING See Appendix P for sample output
Compute statistics for 10 million objects	10 min	Single Partition of all 8 tables Parallel degree = 4 1% of dataset sampled per partition

The final key area of testing was the database maintenance operations. The areas we focused on were index rebuilding and statistics estimation. Measured values for these are shown above. These times illustrate the main "quandary" in the struggle to decide if hash or range partitioning should be employed. Since large repositories would have a certain amount of data locality the task of rebuilding indexes and computing statistics could be typically limited to 70 minutes. However, for a hash partitioned database all partitions must be operated on implying (in our case) 100+ hours to do the same task. It would be imperative to configure the database with many more CPUs than we currently did in an effort to drive down the time to rebuild indexes.

The table and index sizes for the partitioned objects are shown in the next two tables. The total size of the database (including some of the free space left in each table space) was about 2.6 TB.

	Total Blocks Allocated
DMI_OBJECT_TYPE	4,177,781
DMR_CONTENT_R	21,207,615
DMR_CONTENT_S	21,886,381
DM_DOCUMENT_S	3,060,358
DM_SYSOBJECT_R	16,480,289
DM_SYSOBJECT_S	55,929,982
MAIL_DOC_R	11,491,637
MAIL_DOC_S	12,310,297
	146,544,340
	x 8192 bytes per block
	1,200,491,233,280 bytes

INDEX_NAME	SUM(LEAF_BLOCKS)
ADDED1	3,799,672
ADDED2	5,409,892
ADDED3	2,440,942
ADDED4	7,519,800
DMI_OBJECT_TYPE_UNIQUE	4,898,460
D_1F0F424580000005	4,177,636
D_1F0F42458000000E	3,926,909
D_1F0F42458000000F	3,018,184
D_1F0F424580000010	3,935,602
D_1F0F42458000002A	1,823,962
D_1F0F42458000002F	5,479,694
D_1F0F424580000032	2,667,097
D_1F0F424580000034	4,171,388
D_1F0F424580000038	4,996,928
D_1F0F42458000003C	3,929,700
D_1F0F424580000108	3,785,049
D_1F0F424580000109	8,746,289
D_1F0F424580000145	3,784,147
D_1F0F42458000015F	3,791,747
D_1F0F424580000160	4,315,551
D_1F0F424580000500	3,776,012
D_1F0F424580000501	8,648,792
	99,043,453
	x 8192 bytes per block
	811,363,966,976

Appendices

Appendix A: Init.ora

```
background_dump_dest = F:\oracle\admin\perfn17\bdump
compatible = 9.2.0.0.0
control_files = ('F:\oracle\oradata\perfn17\control01.ctl',
'F:\oracle\oradata\perfn17\control02.ctl',
'F:\oracle\oradata\perfn17\control03.ctl')
core_dump_dest = F:\oracle\admin\perfn17\cdump
db_block_size = 8192
db_cache_size = 763363328
db_domain = ''
db_file_multiblock_read_count = 16
db_name = perfn17
dispatchers = '(PROTOCOL=TCP) (SERVICE=perfn17XDB)'
fast_start_mttr_target = 300
hash_join_enabled = TRUE
instance_name = perfn17
java_pool_size = 33554432
large_pool_size = 41943040
log_archive_dest_1 = 'LOCATION=F:\oracle\ora92\RDBMS'
open_cursors = 300
optimizer_index_caching = 95
optimizer_index_cost_adj = 5
parallel_max_servers = 10
pga_aggregate_target = 25165824
processes = 150
query_rewrite_enabled = FALSE
remote_login_passwordfile = EXCLUSIVE
sga_max_size = 1066477240
shared_pool_size = 209715200
sort_area_size = 524288
star_transformation_enabled = FALSE
timed_statistics = TRUE
undo_management = AUTO
undo_retention = 10800
undo_tablespace = UNDOTBS1
user_dump_dest = F:\oracle\admin\perfn17\udump
```

Appendix B: Billion Object Bulk Loading Notes

In this section we detail how the billion objects were loaded. As mentioned earlier our test hardware could support only about 4 million objects per day through the normal loading process, hence, it would have taken 8 months to load a billion objects. Timeline constraints on the availability of the hardware would not permit this. Hence an alternate strategy of using the Oracle Direct Path Loader was followed instead. In this, the rows associated with an object are fabricated into ascii text files with values populated in the same manner that the Content Server would. These ascii files were used as input to the Direct Path Loader (DPL). Since the DPL by-passes even normal transaction processing of the Oracle server (it writes the pages directly to disk) we were able to dramatically improve the load rate.[3]

The content, on the other hand, was loaded by a separate custom loader that mimicked the algorithm followed by the Content Server. These two programs however, had to coordinate their activities so that the metadata in the dmr_content records matched file, format, and location to the ones stored to disk.

On the hardware side Windows-based HP lp1000r's were used to gen the ascii files for the DPL. The files themselves resided in a windows CIFS share on the Netapp Filer. The lp2000r which ran the Oracle database and DPL also had access to this share (to read the files). Most of the machines interconnected to the Filer via Gigabit Ethernet. Three dual-processor Lp1000r's were used for the content loading. Again, Gigabit Ethernet connections were employed when possible. The Content file systems on the NearStore were connected to these servers via Windows CIFS. The hardware for this load phase is outlined in Figure 20.

Overall this strategy for bulk loading the content worked very well. The table below summarizes some of the metrics:

	metric	notes
Max Content Load Rate (per day)	250 million 4K files per day	Using all three content loader machines.
Single Row Generator	25 million in 8 hours	
3 Row Generators	75 million in 13.5 hours	Three generating in parallel
Direct Path Load time	80 min for 25 million objects	See Appendix R for sample DP loader output for largest and smallest tables

Once a partition was filled it could be indexed and statistics computed. In order to ensure reliable and speedy loading we chose to SKIP_INDEX_MAINTENANCE for the Direct Path Loader. The

[3] The downside of this approach is that each "instance" of this row generator is intimately linked with the Content Server release. Many internal fields within the database are subject to change without notice. Contact Documentum consulting for more information if interested.

unfortunate side affect of this was that the indexes for that partition were put in an invalid state. This would cause a query against all of the partitions to fail (until the indexes were rebuilt).

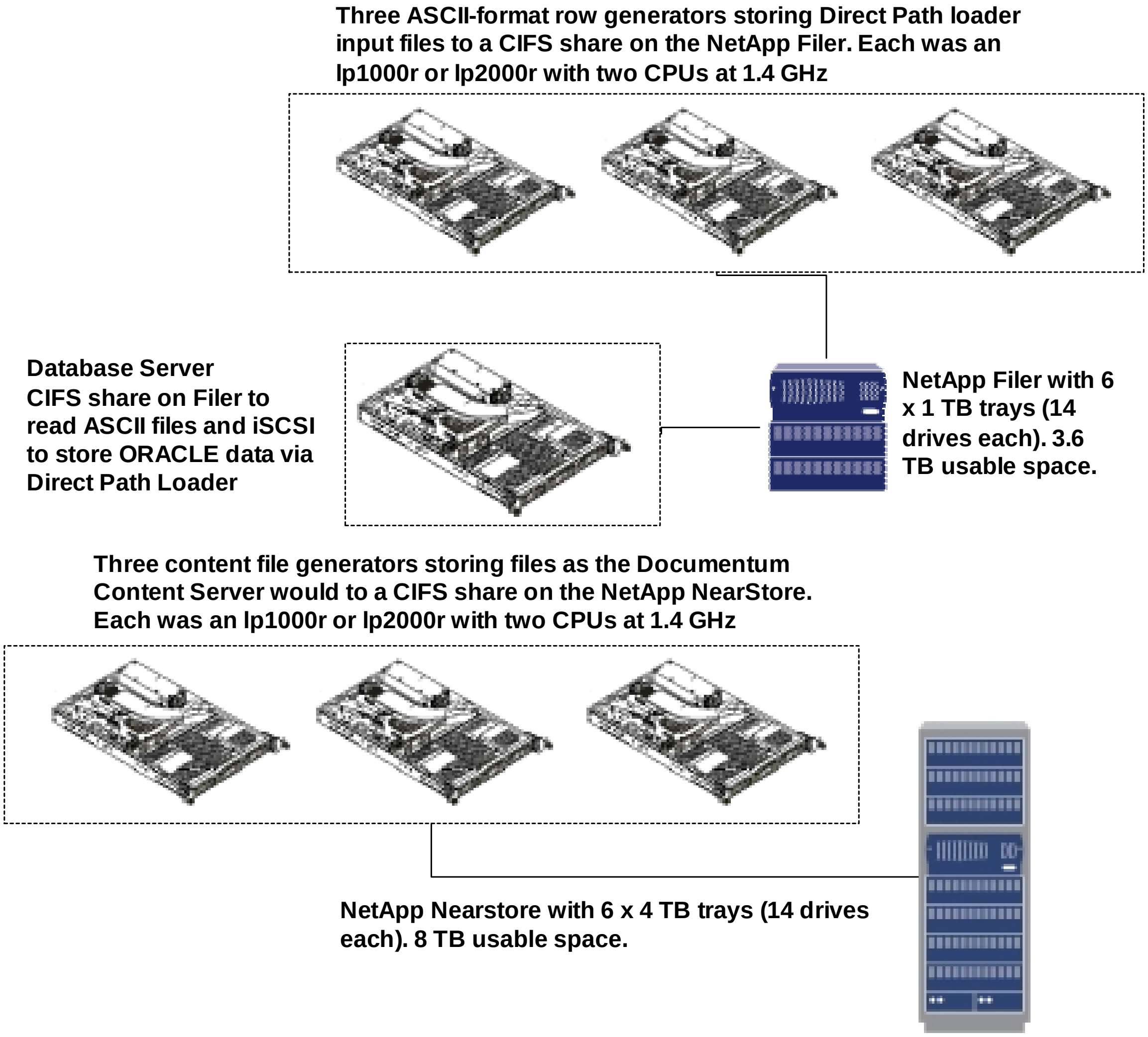


Figure 20: Loader hardware

Appendix C: Database Table Creation & Partitioning Related Scripts

The following section describes how eight tables of the initial Docbase were converted into partitioned tables.

Step 1: Shutdown the Content Server.

Step 2: Backup the current database

Step 3: Rename the tables (DM_SYSOBJECT_S, DM_SYSOBJECT_R, DM_DOCUMENT_S, MAIL_DOC_S, MAIL_DOC_R, DMI_OBJECT_TYPE, DMR_CONTENT_S, DMR_CONTENT_R) to some other name:

```
ALTER TABLE DM_BILLION.DM_SYSOBJECT_S RENAME TO DM_SYSOBJECT_S_1 ;
ALTER TABLE DM_BILLION.DM_SYSOBJECT_R RENAME TO DM_SYSOBJECT_R_1 ;
ALTER TABLE DM_BILLION.DMI_OBJECT_TYPE_S RENAME TO DMI_OBJECT_TYPE_S_1 ;
ALTER TABLE DM_BILLION.DM_MAIL_DOC_S RENAME TO DM_MAIL_DOC_S_1 ;
ALTER TABLE DM_BILLION.DM_MAIL_DOC_R RENAME TO DM_MAIL_DOC_R_1 ;
ALTER TABLE DM_BILLION.DM_DOCUMENT_S RENAME TO DM_DOCUMENT_S_1 ;
ALTER TABLE DM_BILLION.DMR_CONTENT_S RENAME TO DMR_CONTENT_S_1 ;
ALTER TABLE DM_BILLION.DMR_CONTENT_R RENAME TO DMR_CONTENT_R_1 ;
```

Step 4: Define the new table spaces

```
CREATE TABLESPACE DM_BILLION_P2 DATAFILE 'M:\oracle\oradata\DM_BILLION_P2.DBF' SIZE 25G
EXTENT MANAGEMNET LOCAL UNIFORM SIZE 10M;
CREATE TABLESPACE DM_BILLION_P3 DATAFILE 'M:\oracle\oradata\DM_BILLION_P3.DBF' SIZE 25G
EXTENT MANAGEMNET LOCAL UNIFORM SIZE 10M;
CREATE TABLESPACE DM_BILLION_P4 DATAFILE 'M:\oracle\oradata\DM_BILLION_P4.DBF' SIZE 25G
EXTENT MANAGEMNET LOCAL UNIFORM SIZE 10M;
CREATE TABLESPACE DM_BILLION_P5 DATAFILE 'M:\oracle\oradata\DM_BILLION_P5.DBF' SIZE 25G
EXTENT MANAGEMNET LOCAL UNIFORM SIZE 10M;
CREATE TABLESPACE DM_BILLION_P6 DATAFILE 'M:\oracle\oradata\DM_BILLION_P6.DBF' SIZE 25G
EXTENT MANAGEMNET LOCAL UNIFORM SIZE 10M;
CREATE TABLESPACE DM_BILLION_P7 DATAFILE 'M:\oracle\oradata\DM_BILLION_P7.DBF' SIZE 25G
EXTENT MANAGEMNET LOCAL UNIFORM SIZE 10M;
CREATE TABLESPACE DM_BILLION_P8 DATAFILE 'M:\oracle\oradata\DM_BILLION_P8.DBF' SIZE 25G
EXTENT MANAGEMNET LOCAL UNIFORM SIZE 10M;
CREATE TABLESPACE DM_BILLION_P9 DATAFILE 'M:\oracle\oradata\DM_BILLION_P9.DBF' SIZE 25G
EXTENT MANAGEMNET LOCAL UNIFORM SIZE 10M;
CREATE TABLESPACE DM_BILLION_P10 DATAFILE 'M:\oracle\oradata\DM_BILLION_P10.DBF' SIZE 25G
EXTENT MANAGEMNET LOCAL UNIFORM SIZE 10M;
CREATE TABLESPACE DM_BILLION_P11 DATAFILE 'M:\oracle\oradata\DM_BILLION_P11.DBF' SIZE 25G
EXTENT MANAGEMNET LOCAL UNIFORM SIZE 10M;
```

```
CREATE TABLESPACE DM_BILLION_P12 DATAFILE 'M:\oracle\oradata\DM_BILLION_P12.DBF' SIZE 25G
EXTENT MANAGEMNET LOCAL UNIFORM SIZE 10M;
CREATE TABLESPACE DM_BILLION_P13 DATAFILE 'M:\oracle\oradata\DM_BILLION_P13.DBF' SIZE 25G
EXTENT MANAGEMNET LOCAL UNIFORM SIZE 10M;
CREATE TABLESPACE DM_BILLION_P14 DATAFILE 'M:\oracle\oradata\DM_BILLION_P14.DBF' SIZE 25G
EXTENT MANAGEMNET LOCAL UNIFORM SIZE 10M;


-- Creation of temporary tablespace

CREATE TEMPORARY TABLESPACE temp TEMPFILE 'M:\oracle\oradata\DM_temp.tmp' SIZE 25G
EXTENT MANAGEMNET LOCAL UNIFORM SIZE 10M;
```

Step 5: Create the smaller (to be expanded later) table spaces

```
CREATE TABLESPACE DM_BILLION_P15 DATAFILE 'M:\oracle\oradata\DM_BILLION_P15.DBF' SIZE 10M
EXTENT MANAGEMNET LOCAL UNIFORM SIZE 100K;
CREATE TABLESPACE DM_BILLION_P16 DATAFILE 'M:\oracle\oradata\DM_BILLION_P16.DBF' SIZE 10M
EXTENT MANAGEMNET LOCAL UNIFORM SIZE 100K;
       :
       :
       :
CREATE TABLESPACE DM_BILLION_P98 DATAFILE 'M:\oracle\oradata\DM_BILLION_P98.DBF' SIZE 10M
EXTENT MANAGEMNET LOCAL UNIFORM SIZE 100K;
CREATE TABLESPACE DM_BILLION_P99 DATAFILE 'M:\oracle\oradata\DM_BILLION_P99.DBF' SIZE 10M
EXTENT MANAGEMNET LOCAL UNIFORM SIZE 100K;
CREATE TABLESPACE DM_BILLION_P100 DATAFILE 'M:\oracle\oradata\DM_BILLION_P100.DBF' SIZE 10M
EXTENT MANAGEMNET LOCAL UNIFORM SIZE 100K;

Spool off
```

Step 6: Redefine the tables and indexes

Once preceding step is complete then these tables to be need redefined to support partitioning. A small example of this would be the DM_DOCUMENT_S table and its corresponding
indexes would be as shown below. First, assume that:

```
SELECT MAX(R_OBJECT_ID) FROM DM_DOCUMENT_S;
```

Returns the value: 0900f0128100001C. We will place all current rows and this one in the default DM_BILLION_DOCBASE tablespace. All subsequently created ones will show up in the newly created table space. This will set new storage statistics to cover the larger expected allocations.

```
DROP TABLE Billion.DM_DOCUMENT_S CASCADE CONSTRAINTS ;

CREATE TABLE Billion.DM_DOCUMENT_S (
  R_OBJECT_ID  VARCHAR2 (16)  NOT NULL)
   PARTITION By RANGE (R_OBJECT_ID)
   (
       PARTITION P1 VALUES LESS THAN ('090f424581000000')
       TABLESPACE DM_BILLION_DOCBASE,
       PARTITION P2 VALUES LESS THAN ('090f424581989680')
       TABLESPACE DM_BILLION_P2,
       PARTITION P3 VALUES LESS THAN ('090f424582312d00')
       TABLESPACE DM_BILLION_P3,
              :
              :
       PARTITION P98 VALUES LESS THAN ('090f4245bad10680')
       TABLESPACE DM_BILLION_P98,
       PARTITION P99 VALUES LESS THAN ('090f4245bb699d00')
       TABLESPACE DM_BILLION_P99,
       PARTITION P100 VALUES LESS THAN ('090f4245bc023380')
       TABLESPACE DM_BILLION_P100,
    PARTITION PMAX VALUES LESS THAN (MAXVALUE)
    TABLESPACE DM_BILLION_DOCBASE);

DROP INDEX BILLION.D_1F0F424580000145;

CREATE UNIQUE INDEX BILLION.D_1F0F424580000145 ON
  BILLION.DM_DOCUMENT_S(R_OBJECT_ID)
  LOCAL
;
```

Initially we proposed to split the last partition as it filled. Hence one would start with a few partitions and then eventually have many. The above strategy of allocating the partitions up-front (initially proposed by Joy Zhou of Documentum IT during a review of the database design) was favored in this project because it minimized maintenance during the load. It was felt that the query penalty for non-prefixed queries would be minimal for small, empty partitions.

Appendix D: Growing the database with additional Drive hardware

Despite the “offline” loading nature of the Direct Path Loader, database growth was handled in a fashion that would likely mirror production systems. This appendix describes how this was accomplished. First, not all of the storage was online or even available at the start of the test. In fact, at the start only about 350 GB were online. Despite this, it was decided that for as much as possible all partitions should be created initially (hopefully avoiding any splits): some of the table spaces full sized (25 GB) and others not (1 M). The smaller partitions/table spaces could then be resized later when additional storage was available. This procedure for resizing storage is outlined below.

1. Once the new Storage was online (could be seen by Windows) the soon-to-be-resized table spaces were placed in an offline state.
2. These tablespaces were copied to the new storage area (this could take seconds for each file).
3. The newly copied tablespaces were renamed via Oracle to the new path and brought online.
4. Finally, they were expanded inplace. The expansion would take about 10 minutes for 25 GBytes.

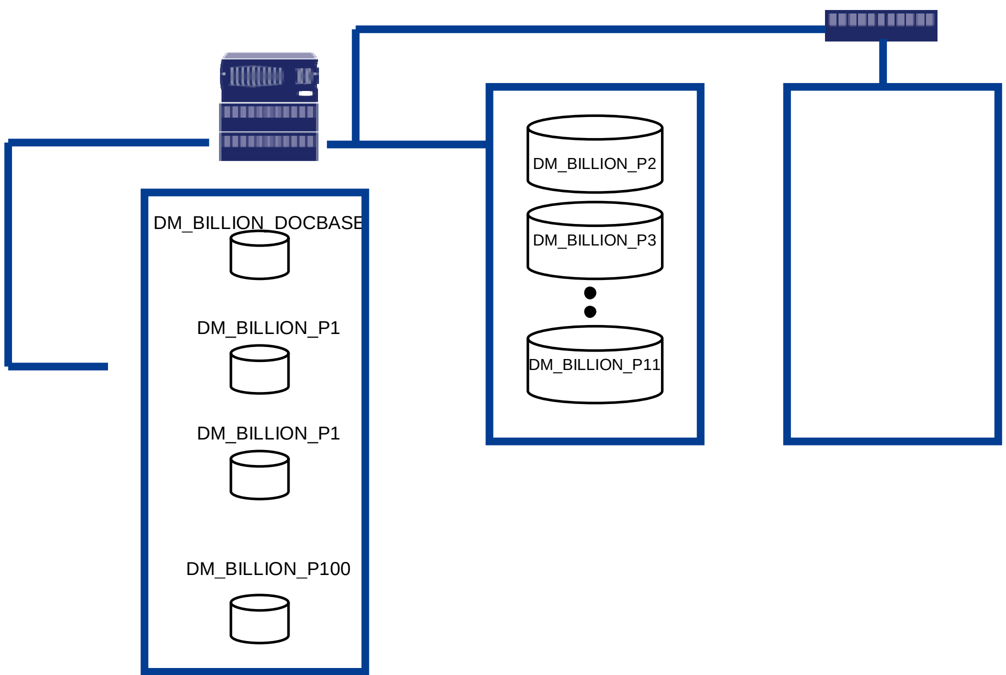


Figure 21: New storage added to Filer during database growth phase

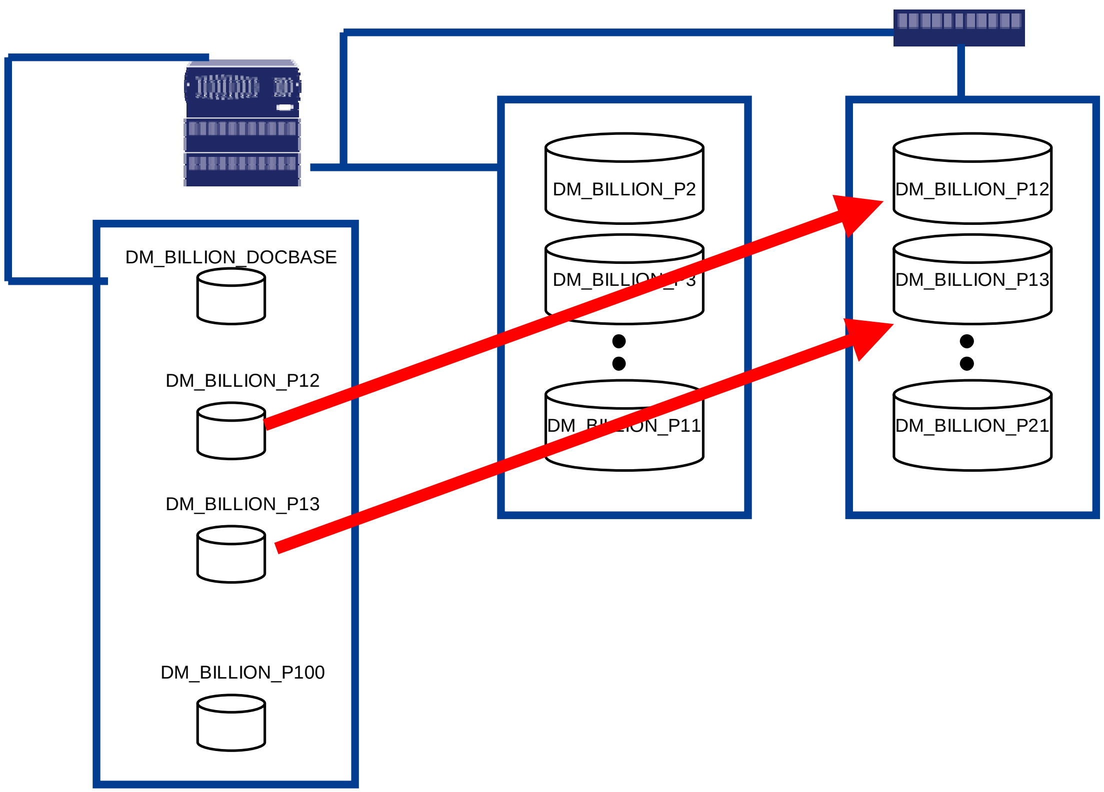


Figure 22: Growing disk capacity with new volumes

An example outline of the scripts used for this procedure are shown below:

1. Alter "soon-to-be-relocated" table spaces offline:

Alter Tablespace DM_BILLION_P60 Offline;
Alter Tablespace DM_BILLION_P61 Offline;
Alter Tablespace DM_BILLION_P62 Offline;
Alter Tablespace DM_BILLION_P63 Offline;

Alter Tablespace DM_BILLION_P64 Offline;
:
:
Alter Tablespace DM_BILLION_P96 Offline;
Alter Tablespace DM_BILLION_P97 Offline;
Alter Tablespace DM_BILLION_P98 Offline;
Alter Tablespace DM_BILLION_P99 Offline;
Alter Tablespace DM_BILLION_P100 Offline;

2. Copy data files associated with those tablespaces to the new drives:

```
Copy M:\ORACLE\ORADATA\DM_BILLION_P60.DBF R:\ORACLE\ORADATA\DM_BILLION_P60.DBF
Copy M:\ORACLE\ORADATA\DM_BILLION_P61.DBF R:\ORACLE\ORADATA\DM_BILLION_P61.DBF
Copy M:\ORACLE\ORADATA\DM_BILLION_P62.DBF R:\ORACLE\ORADATA\DM_BILLION_P62.DBF
Copy M:\ORACLE\ORADATA\DM_BILLION_P63.DBF R:\ORACLE\ORADATA\DM_BILLION_P63.DBF
Copy M:\ORACLE\ORADATA\DM_BILLION_P64.DBF R:\ORACLE\ORADATA\DM_BILLION_P64.DBF
       :
       :
Copy M:\ORACLE\ORADATA\DM_BILLION_P96.DBF T:\ORACLE\ORADATA\DM_BILLION_P96.DBF
Copy M:\ORACLE\ORADATA\DM_BILLION_P97.DBF U:\ORACLE\ORADATA\DM_BILLION_P97.DBF
Copy M:\ORACLE\ORADATA\DM_BILLION_P98.DBF U:\ORACLE\ORADATA\DM_BILLION_P98.DBF
Copy M:\ORACLE\ORADATA\DM_BILLION_P99.DBF U:\ORACLE\ORADATA\DM_BILLION_P99.DBF
Copy M:\ORACLE\ORADATA\DM_BILLION_P100.DBF U:\ORACLE\ORADATA\DM_BILLION_P100.DBF
```

3. Rename tablespace locations within Oracle:

```
Alter Database Rename File 'M:\ORACLE\ORADATA\DM_BILLION_P60.DBF' To 'R:\ORACLE\ORADATA\DM_BILLION_P60.DBF';
Alter Database Rename File 'M:\ORACLE\ORADATA\DM_BILLION_P61.DBF' To 'R:\ORACLE\ORADATA\DM_BILLION_P61.DBF';
Alter Database Rename File 'M:\ORACLE\ORADATA\DM_BILLION_P62.DBF' To 'R:\ORACLE\ORADATA\DM_BILLION_P62.DBF';
Alter Database Rename File 'M:\ORACLE\ORADATA\DM_BILLION_P63.DBF' To 'R:\ORACLE\ORADATA\DM_BILLION_P63.DBF';
Alter Database Rename File 'M:\ORACLE\ORADATA\DM_BILLION_P64.DBF' To 'R:\ORACLE\ORADATA\DM_BILLION_P64.DBF';
Alter Database Rename File 'M:\ORACLE\ORADATA\DM_BILLION_P65.DBF' To 'R:\ORACLE\ORADATA\DM_BILLION_P65.DBF';
              :
              :
Alter Database Rename File 'M:\ORACLE\ORADATA\DM_BILLION_P97.DBF' To 'U:\ORACLE\ORADATA\DM_BILLION_P97.DBF';
Alter Database Rename File 'M:\ORACLE\ORADATA\DM_BILLION_P98.DBF' To 'U:\ORACLE\ORADATA\DM_BILLION_P98.DBF';
Alter Database Rename File 'M:\ORACLE\ORADATA\DM_BILLION_P99.DBF' To 'U:\ORACLE\ORADATA\DM_BILLION_P99.DBF';
```

```
Alter Database Rename File 'M:\ORACLE\ORADATA\DM_BILLION_P100.DBF' To 'U:\ORACLE\ORADATA\DM_BILLION_P100.DBF';
```

4. Online the database

```
Alter Tablespace DM_BILLION_P60 Online;
Alter Tablespace DM_BILLION_P61 Online;
Alter Tablespace DM_BILLION_P62 Online;
Alter Tablespace DM_BILLION_P63 Online;
Alter Tablespace DM_BILLION_P64 Online;
Alter Tablespace DM_BILLION_P65 Online;
       :
       :
Alter Tablespace DM_BILLION_P96 Online;
Alter Tablespace DM_BILLION_P97 Online;
Alter Tablespace DM_BILLION_P98 Online;
Alter Tablespace DM_BILLION_P99 Online;
Alter Tablespace DM_BILLION_P100 Online;
```

5. Resize tablespace

```
Alter database datafile  'R:\ORACLE\ORADATA\DM_BILLION_P60.DBF' Resize 25000 M;
Alter database datafile  'R:\ORACLE\ORADATA\DM_BILLION_P61.DBF' Resize 25000 M;
Alter database datafile  'R:\ORACLE\ORADATA\DM_BILLION_P62.DBF' Resize 25000 M;
Alter database datafile  'R:\ORACLE\ORADATA\DM_BILLION_P63.DBF' Resize 25000 M;
Alter database datafile  'R:\ORACLE\ORADATA\DM_BILLION_P64.DBF' Resize 25000 M;
             :
             :
Alter database datafile  'T:\ORACLE\ORADATA\DM_BILLION_P96.DBF' Resize 25000 M;
Alter database datafile  'U:\ORACLE\ORADATA\DM_BILLION_P97.DBF' Resize 25000 M;
Alter database datafile  'U:\ORACLE\ORADATA\DM_BILLION_P98.DBF' Resize 25000 M;
Alter database datafile  'U:\ORACLE\ORADATA\DM_BILLION_P99.DBF' Resize 25000 M;
Alter database datafile  'U:\ORACLE\ORADATA\DM_BILLION_P100.DBF' Resize 25000 M;
```

Appendix E: Adding Additional Partitions

The following scripts were used to add additional partitions.

Step 1: Add new table spaces

Step 2: split the last partition PMAX into two partitions (one of which is a new PMAX partition).

```
Alter Table dm_sysobject_s Split Partition PMAX At (090f4245bc9aca00) Into (Partition P101 Tablespace DM_BILLION_P101,
Partition PMAX);
Alter Table dm_sysobject_r Split Partition PMAX At (090f4245bc9aca00) Into (Partition P101 Tablespace DM_BILLION_P101,
Partition PMAX);
Alter Table dmi_object_type Split Partition PMAX At (090f4245bc9aca00) Into (Partition P101 Tablespace DM_BILLION_P101,
Partition PMAX);
Alter Table dm_document_s Split Partition PMAX At (090f4245bc9aca00) Into (Partition P101 Tablespace DM_BILLION_P101,
Partition PMAX);
Alter Table mail_doc_s Split Partition PMAX At (090f4245bc9aca00) Into (Partition P101 Tablespace DM_BILLION_P101,
Partition PMAX);
Alter Table mail_doc_r Split Partition PMAX At (090f4245bc9aca00) Into (Partition P101 Tablespace DM_BILLION_P101,
Partition PMAX);
Alter Table dmr_content_s Split Partition PMAX At (060f4245bc9aca00) Into (Partition P101 Tablespace DM_BILLION_P101,
Partition PMAX);
Alter Table dmr_content_r Split Partition PMAX At (060f4245bc9aca00) Into (Partition P101 Tablespace DM_BILLION_P101,
Partition PMAX);
spool off
```

Step 3: Rebuild the Oracle views on the various tables

Splitting partitions caused a side effect that any query interacting over all partitions to behave very badly (perform poorly) because some views became “invalid”. Once this happened Oracle’s PGA area might grow during query execution to beyond what was possible to support on the hardware (making the machine thrash). The remedy was to recompile all of the views driving down the response times to their expected values.

Appendix F: Database Index re-Creation scripts

The following is an example of an index re-build script for a single partition:

```
ALTER INDEX  BILLION.D_1F0F424580000109 REBUILD PARTITION P25 NOLOGGING PARALLEL (DEGREE 4);
ALTER INDEX  BILLION.D_1F0F424580000010 REBUILD PARTITION P25 NOLOGGING PARALLEL (DEGREE 4);
ALTER INDEX  BILLION.D_1F0F424580000108  REBUILD PARTITION P25 NOLOGGING PARALLEL (DEGREE 4) ;
ALTER INDEX  BILLION.D_1F0F42458000000E  REBUILD PARTITION P25 NOLOGGING PARALLEL (DEGREE 4);
ALTER INDEX  BILLION.D_1F0F42458000002A  REBUILD PARTITION P25 NOLOGGING PARALLEL (DEGREE 4);
ALTER INDEX  BILLION.D_1F0F42458000000F  REBUILD PARTITION P25 NOLOGGING PARALLEL (DEGREE 4);
ALTER INDEX  BILLION.D_1F0F42458000002F  REBUILD PARTITION P25 NOLOGGING PARALLEL (DEGREE 4);
ALTER INDEX  BILLION.D_1F0F424580000032  REBUILD PARTITION P25 NOLOGGING PARALLEL (DEGREE 4);
ALTER INDEX  BILLION.D_1F0F42458000003C  REBUILD PARTITION P25 NOLOGGING PARALLEL (DEGREE 4);
ALTER INDEX  BILLION.ADDED1  REBUILD PARTITION P25 NOLOGGING PARALLEL (DEGREE 4);
ALTER INDEX  BILLION.D_1F0F424580000034 REBUILD PARTITION P25_6 NOLOGGING PARALLEL (DEGREE 4);
ALTER INDEX  BILLION.D_1F0F424580000038 REBUILD PARTITION P25_6 NOLOGGING PARALLEL (DEGREE 4);
ALTER INDEX  BILLION.D_1F0F424580000160 REBUILD PARTITION P25_6 NOLOGGING PARALLEL (DEGREE 4);
ALTER INDEX  BILLION.D_1F0F424580000005 REBUILD PARTITION P25_6 NOLOGGING PARALLEL (DEGREE 4);
ALTER INDEX  BILLION.D_1F0F42458000015F REBUILD PARTITION P25_6 NOLOGGING PARALLEL (DEGREE 4);
ALTER INDEX  BILLION.D_1F0F424580000501 REBUILD PARTITION P25 NOLOGGING PARALLEL (DEGREE 4);
ALTER INDEX  BILLION.ADDED4 REBUILD PARTITION P25 NOLOGGING PARALLEL (DEGREE 4);
ALTER INDEX  BILLION.D_1F0F424580000500 REBUILD PARTITION P25 NOLOGGING PARALLEL (DEGREE 4);
ALTER INDEX  BILLION.ADDED2 REBUILD PARTITION P25 NOLOGGING PARALLEL (DEGREE 4);
ALTER INDEX  BILLION.ADDED3  REBUILD PARTITION P25 NOLOGGING PARALLEL (DEGREE 4);
ALTER INDEX  BILLION.D_1F0F424580000145 REBUILD PARTITION P25 NOLOGGING PARALLEL (DEGREE 4);
ALTER INDEX  BILLION.DMI_OBJECT_TYPE_UNIQUE REBUILD PARTITION P25 NOLOGGING PARALLEL (DEGREE 4);
```

Appendix G: Statistics Computation scripts

```
set feed off
set line 200
set head off
set time on
set timing on
set echo on
spool get_billion_table_stats.lst
Execute DBMS_STATS.GATHER_TABLE_STATS(ownname=>'billion',tabname=>'MAIL_DOC_R',partname
=>'P1',estimate percent=>1,degree=>4,granularity=>'partition',cascade=> True);
Execute DBMS_STATS.GATHER_TABLE_STATS(ownname=>'billion',tabname=>'MAIL_DOC_S',partname
=>'P1',estimate percent=>1,degree=>4,granularity=>'partition',cascade=> True);
Execute DBMS_STATS.GATHER_TABLE_STATS(ownname=>'billion',tabname=>'DM_DOCUMENT_S',partname
=>'P1',estimate percent=>1,degree=>4,granularity=>'partition',cascade=> True);
Execute DBMS_STATS.GATHER_TABLE_STATS(ownname=>'billion',tabname=>'DMI_OBJECT_TYPE',partname
=>'P1',estimate percent=>1,degree=>4,granularity=>'partition',cascade=> True);
Execute DBMS_STATS.GATHER_TABLE_STATS(ownname=>'billion',tabname=>'DM_SYSOBJECT_R',partname
=>'P1',estimate percent=>1,degree=>4,granularity=>'partition',cascade=> True);
Execute DBMS_STATS.GATHER_TABLE_STATS(ownname=>'billion',tabname=>'DM_SYSOBJECT_S',partname
=>'P1',estimate percent=>1,degree=>4,granularity=>'partition',cascade=> True);
Execute DBMS_STATS.GATHER_TABLE_STATS(ownname=>'billion',tabname=>'DMR_CONTENT_R',partname
=>'P1 6',estimate percent=>1,degree=>4,granularity=>'partition',cascade=> True);
Execute DBMS_STATS.GATHER_TABLE_STATS(ownname=>'billion',tabname=>'DMR_CONTENT_S',partname
=>'P1 6',estimate percent=>1,degree=>4,granularity=>'partition',cascade=> True);
Execute DBMS_STATS.GATHER_TABLE_STATS(ownname=>'billion',tabname=>'MAIL_DOC_R',partname
=>'P2',estimate percent=>1,degree=>4,granularity=>'partition',cascade=> True);
Execute DBMS_STATS.GATHER_TABLE_STATS(ownname=>'billion',tabname=>'MAIL_DOC_S',partname
=>'P2',estimate percent=>1,degree=>4,granularity=>'partition',cascade=> True);
Execute DBMS_STATS.GATHER_TABLE_STATS(ownname=>'billion',tabname=>'DMI_OBJECT_TYPE',partname
=>'P2',estimate percent=>1,degree=>4,granularity=>'partition',cascade=> True);
Execute DBMS_STATS.GATHER_TABLE_STATS(ownname=>'billion',tabname=>'DM_DOCUMENT_S',partname
=>'P2',estimate percent=>1,degree=>4,granularity=>'partition',cascade=> True);
Execute DBMS_STATS.GATHER_TABLE_STATS(ownname=>'billion',tabname=>'DM_SYSOBJECT_R',partname
=>'P2',estimate percent=>1,degree=>4,granularity=>'partition',cascade=> True);
Execute DBMS_STATS.GATHER_TABLE_STATS(ownname=>'billion',tabname=>'DM_SYSOBJECT_S',partname
=>'P2',estimate percent=>1,degree=>4,granularity=>'partition',cascade=> True);
Execute DBMS_STATS.GATHER_TABLE_STATS(ownname=>'billion',tabname=>'DMR_CONTENT_R',partname
=>'P2_6',estimate percent=>1,degree=>4,granularity=>'partition',cascade=> True);
Execute DBMS_STATS.GATHER_TABLE_STATS(ownname=>'billion',tabname=>'DMR_CONTENT_S',partname
=>'P2_6',estimate_percent=>1,degree=>4,granularity=>'partition',cascade=> True);
:
:
:
```

```
:
:
Execute DBMS_STATS.GATHER_TABLE_STATS(ownname=>'billion',tabname=>'MAIL_DOC_R',partname
=>'P102',estimate percent=>1,degree=>4,granularity=>'partition',cascade=> True);
Execute DBMS_STATS.GATHER_TABLE_STATS(ownname=>'billion',tabname=>'DMI_OBJECT_TYPE',partname
=>'P102',estimate percent=>1,degree=>4,granularity=>'partition',cascade=> True);
Execute DBMS_STATS.GATHER_TABLE_STATS(ownname=>'billion',tabname=>'DM_SYSOBJECT_S',partname
=>'P102',estimate percent=>1,degree=>4,granularity=>'partition',cascade=> True);
Execute DBMS_STATS.GATHER_TABLE_STATS(ownname=>'billion',tabname=>'DM_SYSOBJECT_R',partname
=>'P102',estimate percent=>1,degree=>4,granularity=>'partition',cascade=> True);
Execute DBMS_STATS.GATHER_TABLE_STATS(ownname=>'billion',tabname=>'DM_DOCUMENT_S',partname
=>'P102',estimate_percent=>1,degree=>4,granularity=>'partition',cascade=> True);
Execute DBMS_STATS.GATHER_TABLE_STATS(ownname=>'billion',tabname=>'MAIL_DOC_S',partname
=>'P102',estimate_percent=>1,degree=>4,granularity=>'partition',cascade=> True);
Execute DBMS_STATS.GATHER_TABLE_STATS(ownname=>'billion',tabname=>'DMR_CONTENT_S',partname
=>'P102_6',estimate_percent=>1,degree=>4,granularity=>'partition',cascade=> True);
Execute DBMS_STATS.GATHER_TABLE_STATS(ownname=>'billion',tabname=>'DMR_CONTENT_R',partname
=>'P102_6',estimate_percent=>1,degree=>4,granularity=>'partition',cascade=> True);

Execute DBMS_STATS.GATHER_TABLE_STATS(ownname=>'billion',tabname=>'DMR_CONTENT_S',partname
=>'PMAX',estimate_percent=>1,degree=>4,granularity=>'partition',cascade=> True);
Execute DBMS_STATS.GATHER_TABLE_STATS(ownname=>'billion',tabname=>'MAIL_DOC_R',partname
=>'PMAX',estimate_percent=>1,degree=>4,granularity=>'partition',cascade=> True);
Execute DBMS_STATS.GATHER_TABLE_STATS(ownname=>'billion',tabname=>'MAIL_DOC_S',partname
=>'PMAX',estimate_percent=>1,degree=>4,granularity=>'partition',cascade=> True);
Execute DBMS_STATS.GATHER_TABLE_STATS(ownname=>'billion',tabname=>'DM_SYSOBJECT_R',partname
=>'PMAX',estimate_percent=>1,degree=>4,granularity=>'partition',cascade=> True);
Execute DBMS_STATS.GATHER_TABLE_STATS(ownname=>'billion',tabname=>'DMR_CONTENT_R',partname
=>'PMAX',estimate_percent=>1,degree=>4,granularity=>'partition',cascade=> True);
Execute DBMS_STATS.GATHER_TABLE_STATS(ownname=>'billion',tabname=>'DM_SYSOBJECT_S',partname
=>'PMAX',estimate_percent=>1,degree=>4,granularity=>'partition',cascade=> True);
Execute DBMS_STATS.GATHER_TABLE_STATS(ownname=>'billion',tabname=>'DM_DOCUMENT_S',partname
=>'PMAX',estimate_percent=>1,degree=>4,granularity=>'partition',cascade=> True);
Execute DBMS_STATS.GATHER_TABLE_STATS(ownname=>'billion',tabname=>'DMI_OBJECT_TYPE',partname
=>'PMAX',estimate_percent=>1,degree=>4,granularity=>'partition',cascade=> True);
spool off
```

Appendix H: NetApp Filer Configuration

The following items, shown from the Filer OS command shell, show the system information.

The "Vol status" command show the 5 database volumes (with a number of drives that ranges from 14 to 17). The ASCII load files for the direct path loader were stored on a volume called "ascii_files".

```
netapp> vol status
         Volume State      Status            Options
           vol0 online     normal            root
    ascii_files online     normal            nosnap=on, raidsize=7
      database1 online     normal            nosnap=on, raidsize=17
      database2 online     normal            nosnap=on, raidsize=14
      database3 online     normal            nosnap=on, raidsize=14
      database4 online     normal            nosnap=on, raidsize=14
      database5 online     normal            nosnap=on, raidsize=14
```

Each database volume was around 660G bytes. There was about 116G reserved for Netapp "snapshots". These provide instantaneous backups by saving future changes to the blocks. We did not have sufficient space to reserve these for the entire database.

```
netapp> df
Filesystem              kbytes       used      avail capacity  Mounted on
/vol/vol0/            50119928     190088   49929840       0%  /vol/vol0/
/vol/vol0/.snapshot   12529980       6284   12523696       0%  /vol/vol0/.snapshot
/vol/ascii_files/    357104472  277849460   79255012      78%  /vol/ascii_files/
/vol/ascii_files/.snapshot   18794972        272   18794700       0%  /vol/ascii_files/.snapshot
/vol/database1/      852038740  339049980  112016396      40%  /vol/database1/
/vol/database1/.snapshot  150359776  176691816          0     118%  /vol/database1/.snapshot
/vol/database2/      692281480  466777696  225503784      67%  /vol/database2/
/vol/database2/.snapshot  122167316          0  122167316       0%  /vol/database2/.snapshot
/vol/database3/      692281480  589794960  102486520      85%  /vol/database3/
/vol/database3/.snapshot  122167316          0  122167316       0%  /vol/database3/.snapshot
/vol/database4/      692281480  641039192   51242288      93%  /vol/database4/
/vol/database4/.snapshot  122167316          0  122167316       0%  /vol/database4/.snapshot
/vol/database5/      692281480  641039192   51242288      93%  /vol/database5/
/vol/database5/.snapshot  122167316          0  122167316       0%  /vol/database5/.snapshot
netapp>
```

As mentioned earlier the database file systems were connected to the database server via iSCSI. Each volume was setup as two iSCSI LUNs. Initially, we had chosen "space reservation" for the initial LUN's. This has the side affect of reserving the same amount of space for backup. Once it was determined that we needed additional space the space reservation was broken and new LUNs created from that newly freed up space. This is why the initial LUNs are 310G and the latter ones 367G.

```
netapp> lun show
        /vol/database1/database    382.0g (410194713600)  (r/w, online, mapped)
        /vol/database2/database    310.0g (332877081600)  (r/w, online, mapped)
        /vol/database2/database2a  367.0g (394073164800)  (r/w, online, mapped)
        /vol/database3/database    310.0g (332877081600)  (r/w, online, mapped)
        /vol/database3/database3a  367.0g (394073164800)  (r/w, online, mapped)
        /vol/database4/database    310.0g (332877081600)  (r/w, online, mapped)
        /vol/database4/database4a  367.0g (394073164800)  (r/w, online, mapped)
        /vol/database5/database    310.0g (332877081600)  (r/w, online, mapped)
        /vol/database5/database5a  367.0g (394073164800)  (r/w, online, mapped)

netapp> iscsi show initiator
Initiators connected on adapter iswta:
  Tgt_PG  iSCSI Initiator Name / ISID
     4    iqn.1991-05.com.microsoft:perfnt17 / 40:00:01:37:00:01
     3    iqn.1991-05.com.microsoft:perfnt15.dctmperf.com / 40:00:01:37:00:01

netapp> iscsi show adapter
Adapter:         iswta
Slot:            N/A
Description:     NetApp Software Implementation
Status:          Online
Target Portal Groups:
        portal group 1: inet 172.20.41.198 port 3260
        portal group 3: inet 100.0.0.1 port 3260
        portal group 4: inet 100.0.0.3 port 3260
```

Appendix I: NetApp NearStore Configuration

The following is the configuration output for the Netapp Nearstore array.

```
netapp5> sysconfig
        NetApp Release 6.4.2: Mon Sep 15 12:35:28 PDT 2003
        System ID: 0050394469 (netapp5)
        System Serial Number: 1037288 (netapp5)
        System Rev: G1
        Backplane Part Number: 104-00011
        Backplane Rev: G1
        Backplane Serial Number: 1037288
        slot 0: System Board
                Processors:         2
                Memory Size:        6144 MB
        slot 0: 10/100 Ethernet Controller IV
                e0 MAC Address:     00:a0:98:01:1e:62 (auto-100tx-up)
        slot 0: NetApp ATA/IDE Adapter 0a (0x000001f0)
                0a.0                 30MB
        slot 2: SCSI Host Adapter 2a
        slot 2: SCSI Host Adapter 2b
        slot 4: SCSI Host Adapter 4a
                12 Disks:            2544.0GB
                1 shelf with EMU
        slot 4: SCSI Host Adapter 4b
        slot 5: SCSI Host Adapter 5a
                12 Disks:            2544.0GB
                1 shelf with EMU
        slot 5: SCSI Host Adapter 5b
                12 Disks:            2544.0GB
                1 shelf with EMU
        slot 6: SCSI Host Adapter 6a
                12 Disks:            2544.0GB
                1 shelf with EMU
        slot 6: SCSI Host Adapter 6b
                12 Disks:            2544.0GB
                1 shelf with EMU
        slot 7: NVRAM
                Memory Size:        256 MB
        slot 7: NetApp ATA/IDE Adapter 7a (0x0000afe0)
        slot 8: SCSI Host Adapter 8a
                12 Disks:            2544.0GB
```

```
                1 shelf with EMU
        slot 8: SCSI Host Adapter 8b
                12 Disks:            2544.0GB
                1 shelf with EMU
        slot 9: SCSI Host Adapter 9a
                12 Disks:            2544.0GB
                1 shelf with EMU
        slot 9: SCSI Host Adapter 9b
                12 Disks:            2544.0GB
                1 shelf with EMU
        slot 10: Dual 10/100/1000 Ethernet Controller V
                e10a MAC Address:   00:07:e9:3e:f8:02 (auto-unknown-cfg_down)
                e10b MAC Address:   00:07:e9:3e:f8:03 (auto-1000t-fd-up)
```

```
netapp5> vol status
         Volume State      Status            Options
           vol0 online     normal            root
           vol1 online     normal            raidsize=14, fs_size_fixed=on
           vol2 online     normal            raidsize=14, fs_size_fixed=on
           vol4 online     normal            raidsize=14, fs_size_fixed=on
           vol3 online     normal            raidsize=14, fs_size_fixed=on
```

Initially we had created a single large 8TB file system for all of the objects. However, a bug in the netapp OS limited the maximum number of files in that file system to 200M. We found that by created smaller 2TB file systems we were able to configure around 400M files for each. Netapp has since fixed this bug in their latest software release.

```
netapp5> maxfiles
Volume vol0: maximum number of files is currently 6444162 (4637 used).
Volume vol1: maximum number of files is currently 389999989 (67445556 used).
Volume vol3: maximum number of files is currently 199999989 (182099810 used).
Volume vol4: maximum number of files is currently 389999989 (387519807 used).
Volume vol2: maximum number of files is currently 389999989 (387394099 used).
```

The connection from the Documentum Content Server and the Nearstore was CIFS over Gigabit Ethernet.

```
netapp5> cifs shares
Name         Mount Point                       Description
----         -----------                       -----------
ETC$         /etc                              Remote Administration BUILTIN\Administrators / Full Control
HOME         /vol/vol0/home                    Default Share                    everyone / Full Control
C$           /                                 Remote Administration BUILTIN\Administrators / Full Control
```

```
share3c      /vol/vol3/q31                                          everyone / Full Control
share2       /vol/vol2/q2                                           everyone / Full Control
share1       /vol/vol1/q1                                           everyone / Full Control
share4       /vol/vol4/q4                                           everyone / Full Control
```

```
netapp5> df
Filesystem              kbytes       used      avail capacity  Mounted on
/vol/vol0/           156288248     124696  156163552       0%  /vol/vol0/
/vol/vol0/.snapshot   39072060       4120   39067940       0%  /vol/vol0/.snapsh
ot
/vol/vol1/          2031747200  289277536 1742469664      14%  /vol/vol1/
/vol/vol1/.snapshot  507936796      88676  507848120       0%  /vol/vol1/.snapsh
ot
/vol/vol3/           781441232  715317692   66123540      92%  /vol/vol3/
/vol/vol3/.snapshot  195360304    7283076  188077228       4%  /vol/vol3/.snapsh
ot
/vol/vol4/          2031747200 1628144460  403602740      80%  /vol/vol4/
/vol/vol4/.snapshot  507936796     118668  507818128       0%  /vol/vol4/.snapsh
ot
/vol/vol2/          2031747200 1625454264  406292932      80%  /vol/vol2/
/vol/vol2/.snapshot  507936796     275332  507661464       0%  /vol/vol2/.snapshot
```

Appendix J: State of Docbase

```
StateOfDocbase Report For DocBase Billion As Of 11/16/2003 20:45:56

Docbase Configuration:
  Federation Name:                <Billion is not in a Federation>
  Docbase ID:                     1000005
  Security Mode:                  acl
  Folder Security:                On
  Authorization Protocol:         <Not defined>
  Database:                       Oracle
  Database Index Store:           DM_Billion_index
  Mac Access Protocol:            none

Server Configuration:
  Server Name:                    Billion
  Server Version:                 5.2.0.185 Win32.Oracle
  Default ACL:                    Default ACL of User
  Host Name:                      perfnt23
  Install Owner:                  dmadmin
  Install Domain:                 qamaster2
  Operator Name:                  Billion
  Agent Launcher:                 agent_exec_method
  Checkpoint Interval:            300 seconds
  Compound Integrity:             On - Server enforces integrity for virtual documents
  Turbo Backing Store:            filestore_01
  Rendition Backing Store:        <Not defined>
  Web Server Location:            PERFNT23
  Web Server Port:                80
  Rightsite Image:                <Not defined>
  Secure Connect Mode:            native
  Trusted Mode:                   0

Server Locations:
  events_location                 C:\Documentum\share\data\events
  common_location                 C:\Documentum\share\data\common
  temp_location                   C:\Documentum\share\temp
  log_location                    C:\Documentum\dba\log
  system_converter_location       C:\Documentum\product\5.2\convert
  user_converter_location         <Not defined>
  verity_location                 C:\Documentum\fulltext\verity271
  user_validation_location        <Not defined>
  assume_user_location            C:\Documentum\dba\dm_assume_user.exe
  change_password_location        C:\Documentum\dba\dm_change_password.exe
  signature_chk_loc               C:\Documentum\dba\dm_check_password.exe
  stage_destroyer_location        <Not defined>
```

Set Information:

```
ALLUSERSPROFILE=C:\Documents and Settings\All Users
APPDATA=C:\Documents and Settings\dmadmin.QAMASTER2\Application Data
APP_JAR=C:\Documentum\RightSite\applications\wcm\app_jarfiles
ClassPath=C:\Program
Files\Documentum\dctm.jar;C:\Documentum\config;C:\Documentum\product\5.2\install\java;C:\Documentum\RightSite\applicatio
ns\centraladmin\app;C:\Documentum\RightSite\applications\centraladmin\app\com\documentum\ldap\ldapfilt.jar;C:\Documentum
\RightSite\applications\centraladmin\app\com\documentum\ldap\ldapjdk.jar;C:\Documentum\RightSite\applications\wcm\app_ja
rfiles\support.jar;C:\Documentum\RightSite\applications\wcm\app_jarfiles\cps.jar;C:\Documentum\RightSite\applications\wc
m\app_jarfiles\wcm.jar;C:\Documentum\RightSite\applications\wcm\app_jarfiles\wcmclient.jar;C:\Documentum\RightSite\appli
cations\wcm\app_jarfiles\xmlTemplateServer.jar;C:\Documentum\RightSite\applications\wcm\app_jarfiles\htmlParser.jar;C:\P
ROGRA~1\DOCUME~1\Shared\Dds.jar
CLIENTNAME=PERFNT22
CommonProgramFiles=C:\Program Files\Common Files
COMPUTERNAME=PERFNT23
ComSpec=C:\WINNT\system32\cmd.exe
DCTMDA=C:\Documentum\RightSite\applications\centraladmin
DFC_DATA=C:\Documentum
DM_HOME=C:\Documentum\product\5.2
DOCUMENTUM=C:\Documentum
IC_ZIP=C:\WINNT\java\trustlib
JMQ_HOME=C:\Program Files\iPlanetMessageQueue2.0
LOGONSERVER=\\QANT5
NLS_LANG=AMERICAN_AMERICA.UTF8
NUMBER_OF_PROCESSORS=2
OS=Windows_NT
Os2LibPath=C:\WINNT\system32\os2\dll;
Path=C:\Program Files\Documentum\Shared;C:\Documentum\product\5.2\bin;C:\oracle\ora92\bin;C:\Program
Files\Oracle\jre\1.3.1\bin;C:\Program
Files\Oracle\jre\1.1.8\bin;C:\WINNT\system32;C:\WINNT;C:\WINNT\System32\Wbem;C:\Program Files\Common Files\Network
Associates\VirusScan Engine\4.0.xx\;C:\Program Files\iPlanetMessageQueue2.0\bin;C:\Program Files\Debugging Tools for
Windows;c:\ntreskit\;c:\shell\bin;C:\DOCUME~2\RIGHTS~1\product\bin;C:\Program Files\Rational\common
PATHEXT=.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.VBS
PROCESSOR_ARCHITECTURE=x86
PROCESSOR_IDENTIFIER=x86 Family 6 Model 11 Stepping 1, GenuineIntel
PROCESSOR_LEVEL=6
PROCESSOR_REVISION=0b01
ProgramFiles=C:\Program Files
PROMPT=$P$G
SESSIONNAME=RDP-Tcp#46
SystemDrive=C:
SystemRoot=C:\WINNT
TEMP=C:\DOCUME~1\DMADMI~1.QAM\LOCALS~1\Temp
TMP=C:\DOCUME~1\DMADMI~1.QAM\LOCALS~1\Temp
```

USERDOMAIN=QAMASTER2
USERNAME=dmadmin
USERPROFILE=C:\Documents and Settings\dmadmin.QAMASTER2
WF_RESOURCES=C:\oracle\ora92\WF\RES\WFus.RES
windir=C:\WINNT
_NT_SYMBOL_PATH=c:\winnt\symbols

Registered tables in the docbase:

Table Name	Table Owner	Owner:Group:World Permits
adm_turbo_size	Billion	1:0:0
dm_display_config_r	Billion	1:1:1
dm_display_config_s	Billion	1:1:1
dm_extents	Billion	1:0:0
dm_federation_log	Billion	15:7:3
dm_free_space	Billion	1:0:0
dm_indexes	Billion	1:0:0
dm_portinfo	Billion	1:0:1
dm_queue	Billion	1:0:1
dm_replica_catalog	Billion	15:0:0
dm_replica_delete	Billion	15:0:0
dm_replica_delete_info	Billion	15:0:0
dm_replication_events	Billion	15:0:0
dm_scope_config_r	Billion	1:1:1
dm_scope_config_s	Billion	1:1:1
dm_store_s	Billion	1:0:0
dm_tasks_all	Billion	1:0:1
dm_tasks_dequeued	Billion	1:0:1
dm_tasks_queued	Billion	1:0:1
dmi_dd_type_info_rv	Billion	1:1:1
dmi_dd_type_info_sv	Billion	1:1:1
dmi_index_s	Billion	1:0:0
dmi_replica_record_r	Billion	1:0:0
lookup_holder	Billion	15:0:0

Number of Documents by Type:

Document Type	Count
mail_doc	1,006,632,963
sop	59,950
dm_document	36,915
dm_folder	4,045
dm_cabinet	3,560
dmi_expr_code	118
dm_activity	98
dm_method	79

dm_job	34
dm_location	25
dm_registered	24
dm_process	12
dm_procedure	11
dm_smart_list	4
dm_script	3
dm_menu_system	2
dm_business_pro	1
dm_mount_point	1
dm_xml_application	1
dm_xml_config	1
dm_server_config	1
dm_query	1
dm_outputdevice	1
dm_docbase_config	1
dm_format_preferences	1
Total:	-------------
	1,006,737,852

Number of Documents by Format:

Document Format	Count
msw8	251,723,830
crtext	251,658,897
pdf	251,658,274
tiff	251,658,241
text	30,484
jpeg	40
gif	27
mdoc55	9
maker55	5
<NO CONTENT>	3
dtd	3
vrf	2
amipro	1
xml	1
wp8	1
wp7	1
wp6	1
ustn	1
ppt8_template	1
ppt8	1
powerpoint	1
msw8template	1
msw6	1
excel5book	1

ms_access7	1
ms_access8	1
ms_access8_mde	1
excel8book	1
excel8template	1
Total:	-------------
	1,006,729,832

Number of Documents by Storage Area:

Storage Area	Count
filestore02	385,875,968
filestore_03	385,875,969
filestore_01	61,765
filestore_05	161,139,456
filestore_06	73,776,671
<NO STORE>	3
Total:	-------------
	1,006,729,832

Content Size(KB) by Format:

Format	Largest	Average	Total
pdf	102	4	967,292,401
msw8	79	4	887,729,975
tiff	4	3	770,256,527
crtext	177	2	616,569,167
text	50	24	717,889
gif	44	44	1,196
mdoc55	159	85	762
html	8	8	320
vrf	155	107	213
maker55	38	23	115
dtd	40	32	97
ms_access7	82	82	82
jpeg	2	2	67
ms_access8_mde	60	60	60
ms_access8	58	58	58
msw8template	27	27	27
excel5book	15	15	15
powerpoint	15	15	15
excel8book	14	14	14
excel8template	14	14	14
msw6	12	12	12
ustn	10	10	10
ppt8_template	10	10	10

ppt8	8	8	8
amipro	4	4	4
wp7	1	1	1
wp8	1	1	1
dm_internal	1	1	1
wp6	1	1	1
xml	1	1	1
dm_fulltext_copy	0	0	0

Content Size(KB) by Renditions:

Format	Largest	Average	Total
html	8	8	320
dm_fulltext_copy	0	0	0

```
Content Size(KB) Summary:
filestore_01
  Largest Content:                 177
  Average Content:                  31
  Total Content:             1,931,151
filestore_02
  Largest Content:                   4
  Average Content:                   3
  Total Content:         1,543,503,872
filestore_03
  Largest Content:                  79
  Average Content:                  79
  Total Content:         1,543,503,876
filestore_05
  Largest Content:                   4
  Average Content:                   3
  Total Content:           644,557,824
filestore_06
  Largest Content:                   4
  Average Content:                   3
  Total Content:           295,106,684
dm_turbo_store
  Largest Content:                   0
  Average Content:                   0
  Total Content:                     0

------------------------

  GTotal Content:        3,242,569,048
  GTotal Rendition:                320 (0.00% of total content)
```

```
Number of Users and Groups:
  Named Users                                  3,547
  Groups                                           8

ACL Summary:
  Number of ACLs:                                6,246
  Number of Internal ACLs:                       6,235
  Number of External System ACLs:                   11
  Number of External Private ACLs:                   0
```

Appendix K: Sample benchmark configuration file

```
DOCBASE_NAME = Billion  # docbase name
HOSTNAME = perfnt24           # name of web server Host
SERVER_OS = windows                 # unix or windows
PORT = 9080                  # Tomcat
PASSWD = bench1              # password for all users
APP_NAME = webtop             # web application name (used in the urls)
TRACE_LEVEL = 5              # trace level, (10 is most verbose - use only for debugging)
USER_TYPE   = Contributor    # don't change this


BENCHMARK_TYPE = TEST


TEST_OPERATIONS =
LOGIN,SEARCH_BY_CHRONICLE_ID,SEARCH_BY_CHRONICLE_ID,SEARCH_BY_CHRONICLE_ID,SEARCH_BY_CHRONICLE_ID,SEARCH_BY_
CHRONICLE_ID,SEARCH_BY_CREATION_DATE

USER_XACT_CNT   = 7
USER_PREFIX = user         # prefix for actual user name
TOTAL_USERS     = 125       # total number of users that will log on
USER_SUFX_START =   701       # start of suffix index
USER_NUMBER     =  125       # number of users
INITIAL_LOGIN_WAIT = 10      # wait time(seconds) for each successive user to login
USER_SLEEP_TIME = 60         # sleep time (seconds) between operations
SLEEP_TIME_RANDOM = TRUE    # time between operations can be exact(false) or slightly random
USER_RESTART_INTERVAL= 100  # sleep time in seconds to allow for timeout - 35 mins is 2100
TEST_DURATION   = 6000        # Duration of test (seconds)


# These values are docbase specific
CABINET_PREFIX = CABINET     # prefix of the test cabinets for WEBTOP_SIMPLE tests
TARGET_FOLDER = 50_folders    # folder width for WEBTOP_NAVIGATE tests
OBJECTS_DISPLAYED = 20

# Next 2 parameters are used for SEARCH_BY_OBJECT_ID and SEARCH_BY_OBJECT_ID_FO operations
```

```
#LOW_OBJECT_ID  =090f4245b1e03500  # last two digits are ignored,
LOW_OBJECT_ID  =090f42459ffe9200  # last two digits are ignored,
                          # assumes =090F424580027A00
HIGH_OBJECT_ID =090f4245a0972880  # last two digits are ignored,
                          # assumes =090F42458002AF00

# Next 5 parameters are used for SEARCH_BY_CDATE_RANGE and SEARCH_BY_CDATE_RANGE_FO operations
DAY_SEARCH_TOKEN =9/22/2003
HOUR_SEARCH_TOKEN =18
MINUTE_SEARCH_TOKEN =41
SECOND_SEARCH_TOKEN =00
RANGE_LIMIT =60

# Next 4 parameters are used for the search operations
SEARCH_STRING1 =subject
SEARCH_RANGE1 =1000
SEARCH_STRING2 =unused
SEARCH_RANGE2 =0

# Maintain a minimum number of subscriptions for a user so the subscriptions will not dwindle
SUBSCRIPTION_LIMIT = 3

# These should not require editing
CONFIG_TYPE = USER           # defines the user profile
ALL_DYNAMIC_HTML = true      # used for discarding images, css and js files
TOTAL_WORKER_THREADS = 1     # total worker threads
CONSECUTIVE_WP_ACCESS = 4    # consecutive web page accesses
CONTENT_LOC_UNIX = /export/perfsun1-2/bueche/contentXfer # unix content location
DELETE_CTXFR_FILES=true
```

documentum

Appendix L: Custom Index Definitions

```
CREATE INDEX BILLION.ADDED1 ON
  BILLION.DM_SYSOBJECT_S(R_CREATION_DATE,TITLE)
  LOCAL
;
REATE INDEX BILLION.ADDED2 ON
  BILLION.MAIL_DOC_S(DATE_RECIEVED,MESSAGE_IDENTIFIER)
  LOCAL
;
CREATE INDEX BILLION.ADDED3 ON
  BILLION.MAIL_DOC_S(MESSAGE_IDENTIFIER)
  LOCAL
;
CREATE INDEX BILLION.ADDED4 ON
  BILLION.MAIL_DOC_R(RECIPIENTS, OTHER_RECIPIENTS)
  LOCAL;
```

Appendix M: Type definition for mail_doc

This type mail_doc is a subtype of dm_document.

Attribute name	Docbase Type
message_identifier DD Label: “Message Identifier”	char(255)
media_type DD Label: “Media Type”	char(32)
publication_date DD Label: “Publication Date”	char(64)
date_received DD Label: “Date Received”	char(64)
recipients DD Label: “Recipient”	char(48) repeating
other_recipients DD Label: “Other Recipient”	char(48) repeating
originating_org DD Label: “Originating Organization”	char(128)

Appendix N: Customization Changes Made to Webtop Advanced Search

SearchNlsProp.properties (items in bold indicated "customized" area)

```
#
#***************************************************************************
#
# Confidential Property of Documentum, Inc.
# (c) Copyright Documentum, Inc. 2001.
# All Rights reserved.
# May not be used without prior written agreement
# signed by a Documentum corporate officer.
#
# *************************************************************************
#
#***************************************************************************


# An ordered and comma seperated list of other properties files
# Lookup starts with this file, and then processes down to the included files sequentially.
# Note: all the included properties should introduce no problems in packaging component.
#       - properties in the parent directory or current directory may be included.
#       - properties in dependent-base applications may be included.
#       - properties in dependent components may be included.
# Note: There is little penality in including a property file more than twice or circularly, either directly or
indirectly.
NLS_INCLUDES=com.documentum.web.form.query.PredicateNlsProp,\
             com.documentum.webcomponent.GenericObjectNlsProp,\
             com.documentum.webcomponent.GenericActionNlsProp,\
             com.documentum.webcomponent.GenericNavigationNlsProp

MSG_DOCUMENT_BYFULLTEXT=Show Documents
MSG_FOLDERS_FOUND=Show Folders
MSG_SYSOBJECT_BYPROPERTY=Show All Objects

MSG_DOCUMENT=Document
MSG_FOLDER=Folder
MSG_CATEGORY=Category
MSG_SYSOBJECT=Object
```

MSG_MAIL_DOC=Mail Doc

```
MSG_NAME=Name
MSG_OBJECT_ID=Object ID
```

MSG_CHRONICLE_ID=Chronicle ID
MSG_RECIPIENTS=Recipients
MSG_OTHER_RECIPIENTS=Other Recipients
MSG_DATE_RECIEVED=Date Recieved
MSG_MESSAGE_IDENTIFIER=Message Identifier

```
MSG_TYPE=Type
MSG_TITLE=General
MSG_PROP_TITLE=Title
MSG_SUBJECT=Subject
MSG_AUTHORS=Authors
MSG_OWNERS=Owners
MSG_KEYWORDS=Keywords
MSG_EVIDENCE=Evidence
MSG_CREATED=Created
MSG_MODIFIED=Modified
MSG_DATE_MODIFIED=Date Modified
MSG_WEEK=week
MSG_DAY=day
MSG_MONTH=month
MSG_PLEASE_SELECT=Please select...
MSG_YEAR=year
MSG_ADV_SEARCH=Advanced Search
MSG_CLOSE=Close
MSG_SEARCH_FOR=Search for
MSG_TEXT=Full Text
MSG_FIND_RESULTS=Find Results:
MSG_CONTAINING=Containing all the words
MSG_EXACT_PHRASE=With the exact phrase
MSG_ANY_WORDS=With any of the words
MSG_WITHOUT_WORDS=Without the words
MSG_CASE=Match case
MSG_PROPERTIES_LABEL=Properties
MSG_LOOK_IN=Look in
MSG_BROWSE=Browse
MSG_AND=and
MSG_REMOVE=Remove
MSG_ADD_PROPERTY=Add Property
MSG_DATE=Date
MSG_IN_THE_LAST=In the last
MSG_SIZE=Size
MSG_ADV_OPTIONS=Advanced options
MSG_FIND_HIDDEN=Find hidden objects
MSG_FIND_VERSIONS=Find all versions
```

MSG_FORCE_ORDER=ENABLE(FORCE_ORDER)

```
MSG_SEARCH=Search
MSG_CLOSE=Close
MSG_CLEAR=Clear
```

```
MSG_CANCEL=Cancel
MSG_INVALID_DATE=Invalid Date
MSG_QUOTE=Quotes not allowed
MSG_LOCK_OWNER=Checked Out By
MSG_PATH=Path
MSG_RANK=Ranking
MSG_WAIT=Please wait, processing search...

MSG_COULDNT_RESET=Could not reset control
MSG_KB=KB
MSG_DATE_CREATED=Date Created
MSG_HELP=Help
MSG_FROM_DATE=From
MSG_TO_DATE=To

MSG_FOLDER_NOTEXIST=Folder {0} doesn''t exist.
MSG_INVALID_SIZE=Size value ({0}) is invalid in advanced search.
MSG_INVALID_INTEGER=Integer value ({0}) is invalid in advanced search.
MSG_INVALID_NUMBER=Number value ({0}) is invalid in advanced search.

MSG_IS_TRUE=is true
MSG_IS_FALSE=is false

# Accessibility strings
MSG_HELP_TIP=Help Button
MSG_SEARCH_TIP=Search Button
MSG_CANCEL_TIP=Cancel Button
MSG_CLEAR_TIP=Clear Button
MSG_CLOSE_TIP=Close Button
MSG_BROWSE_TIP=Browse Button
MSG_REMOVE_TIP=Remove Button
MSG_ADD_PROPERTY_TIP=Add Property Button
MSG_SAVE_SEARCH_TIP=Save Search Button
MSG_SEARCH_FILTER=Search Filter
```

Advsearch_component.xml (abridged with changes in **bold**)

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>

<!-- Confidential Property of Documentum, Inc.                            -->
<!-- (c) Copyright Documentum, Inc. 2001.                                 -->
<!-- All Rights reserved.                                                 -->
<!-- May not be used without prior written agreement                      -->
<!-- signed by a Documentum corporate officer.                            -->
<!--                                                                      -->
<!-- Component: advancedSearch                                            -->
<!-- Scope: None                                                          -->
<config version='1.0'>

   <!-- this component is the generic version so has unqualified scope -->
   <scope>

      <!-- the browser tree component definition -->
      <component id="advsearch" extends="advsearch:webcomponent\config\library\search\advsearch_component.xml">

         <!-- Description (not NLS'd) -->
        <desc>
            You can use the advanced_search (advanced search) component to do the following:
            1. Provide advanced search functionality in all views at all times, based on
            sysobject or custom attributes, to locate specific system objects in specific
            docbases.
            2. Define exactly which attributes to provide as (optional) search criteria for
            users, depending on the current context. For example, documents and folders need
            different sets of attributes.
            3. Optimize searches by allowing parameters, object types, and constraints on text,
            date, and file size to be included in search queries.
            4. Allow users to specify the number of records to return per page of search results.
            5. Allow users to perform specific actions on located objects.
         </desc>

         <!-- Description (not NLS'd) -->
         <desc>
            Advanced search component.
         </desc>

         <!-- Component Layout -->
         <pages>
            <start>/custom/library/advancedsearch/advsearch.jsp</start>
         </pages>

         <!-- Component Behavior -->
         <class>com.documentum.custom.library.advsearch.AdvSearch</class>
```

```
<nlsbundle>com.documentum.custom.library.search.SearchNlsProp</nlsbundle>

<params>
   <param name="folderpath" required="false"></param>
</params>

<!-- Component specific Configuration -->

<!-- All the types to display in advanced search -->
<search_types>

   <type id='dm_document'>
      <name><nlsid>MSG_DOCUMENT</nlsid></name>
      <attributes>
         <attribute>
           <name><nlsid>MSG_NAME</nlsid></name>
           <docbase_attribute>object_name</docbase_attribute>
           <attribute_type>string</attribute_type>
         </attribute>

         <attribute>
           <name><nlsid>MSG_OBJECT_ID</nlsid></name>
           <docbase_attribute>r_object_id</docbase_attribute>
           <attribute_type>string</attribute_type>
         </attribute>

         <attribute>
             <name><nlsid>MSG_CHRONICLE_ID</nlsid></name>
             <docbase_attribute>i_chronicle_id</docbase_attribute>
             <attribute_type>string</attribute_type>
           </attribute>

         <attribute>
           <name><nlsid>MSG_TYPE</nlsid></name>
           <docbase_attribute>r_object_type</docbase_attribute>
           <attribute_type>string</attribute_type>
         </attribute>

         <attribute>
           <name><nlsid>MSG_PROP_TITLE</nlsid></name>
           <docbase_attribute>title</docbase_attribute>
           <attribute_type>string</attribute_type>
         </attribute>

         <attribute>
           <name><nlsid>MSG_SUBJECT</nlsid></name>
           <docbase_attribute>subject</docbase_attribute>
```

```
        <attribute_type>string</attribute_type>
      </attribute>

        :
        :
        :
        :

<type id='mail_doc'>
   <name><nlsid>MSG_MAIL_DOC</nlsid></name>
   <attributes>
      <attribute>
        <name><nlsid>MSG_NAME</nlsid></name>
        <docbase_attribute>object_name</docbase_attribute>
        <attribute_type>string</attribute_type>
      </attribute>

      <attribute>
        <name><nlsid>MSG_OBJECT_ID</nlsid></name>
        <docbase_attribute>r_object_id</docbase_attribute>
        <attribute_type>string</attribute_type>
      </attribute>

        <attribute>
          <name><nlsid>MSG_CHRONICLE_ID</nlsid></name>
          <docbase_attribute>i_chronicle_id</docbase_attribute>
          <attribute_type>string</attribute_type>
        </attribute>

      <attribute>
        <name><nlsid>MSG_TYPE</nlsid></name>
        <docbase_attribute>r_object_type</docbase_attribute>
        <attribute_type>string</attribute_type>
      </attribute>

      <attribute>
        <name><nlsid>MSG_PROP_TITLE</nlsid></name>
        <docbase_attribute>title</docbase_attribute>
        <attribute_type>string</attribute_type>
      </attribute>

      <attribute>
        <name><nlsid>MSG_SUBJECT</nlsid></name>
        <docbase_attribute>subject</docbase_attribute>
```

```
  <attribute_type>string</attribute_type>
</attribute>

<attribute>
  <name><nlsid>MSG_AUTHORS</nlsid></name>
  <docbase_attribute>authors</docbase_attribute>
  <attribute_type>string</attribute_type>
</attribute>

<attribute>
  <name><nlsid>MSG_KEYWORDS</nlsid></name>
  <docbase_attribute>keywords</docbase_attribute>
  <attribute_type>string</attribute_type>
</attribute>

<attribute>
  <name><nlsid>MSG_CREATED</nlsid></name>
  <docbase_attribute>r_creation_date</docbase_attribute>
  <attribute_type>time</attribute_type>
</attribute>

<attribute>
  <name><nlsid>MSG_MODIFIED</nlsid></name>
  <docbase_attribute>r_modify_date</docbase_attribute>
  <attribute_type>time</attribute_type>
</attribute>

<attribute>
    <name><nlsid>MSG_DATE_RECIEVED</nlsid></name>
    <docbase_attribute>date_recieved</docbase_attribute>
    <attribute_type>time</attribute_type>
  </attribute>

  <attribute>
    <name><nlsid>MSG_RECIPIENTS</nlsid></name>
    <docbase_attribute>recipients</docbase_attribute>
    <attribute_type>string</attribute_type>
  </attribute>

  <attribute>
    <name><nlsid>MSG_OTHER_RECIPIENTS</nlsid></name>
    <docbase_attribute>other_recipients</docbase_attribute>
    <attribute_type>string</attribute_type>
  </attribute>
```

```
              <attribute>
                <name><nlsid>MSG_MESSAGE_IDENTIFIER</nlsid></name>
                <docbase_attribute>message_identifier</docbase_attribute>
                <attribute_type>string</attribute_type>
              </attribute>
           </attributes>
        </type>

    </search_types>

      :
      :
      :

   </scope>
</config>
```

advsearch.jsp (abridged and annotated in bold):

```
<%
/**
 *******************************************************************************
 *
 * Confidential Property of Documentum, Inc.
 * (c) Copyright Documentum, Inc. 2001.
 * All Rights reserved.
 * May not be used without prior written agreement
 * signed by a Documentum corporate officer.
 *
 *******************************************************************************
 *
             :
             :
             :

                     <dmf:panel name='advoptions'>
                        <table>
                           <tr>
                              <td width='7'>
                               
                              </td>
                              <td>
                                 <table>
                                    <tr>
                                       <td scope="row">
                                          <dmf:checkbox name='findhiddencheck' nlsid='MSG_FIND_HIDDEN'
tooltipnlsid="MSG_FIND_HIDDEN" />
                                       </td>
                                    </tr>
                                    <tr>
                                       <td scope="row">
                                          <dmf:checkbox name='findversioncheck' nlsid='MSG_FIND_VERSIONS'
tooltipnlsid="MSG_FIND_VERSIONS" />
                                       </td>
                                    </tr>
                                    <tr>
                                       <td scope="row">
                                              <dmf:checkbox name='forceordercheck' nlsid='MSG_FORCE_ORDER'
tooltipnlsid="MSG_FORCE_ORDER" />
                                           </td>
                                    </tr>
                                 </table>
                              </td>
```

:
:

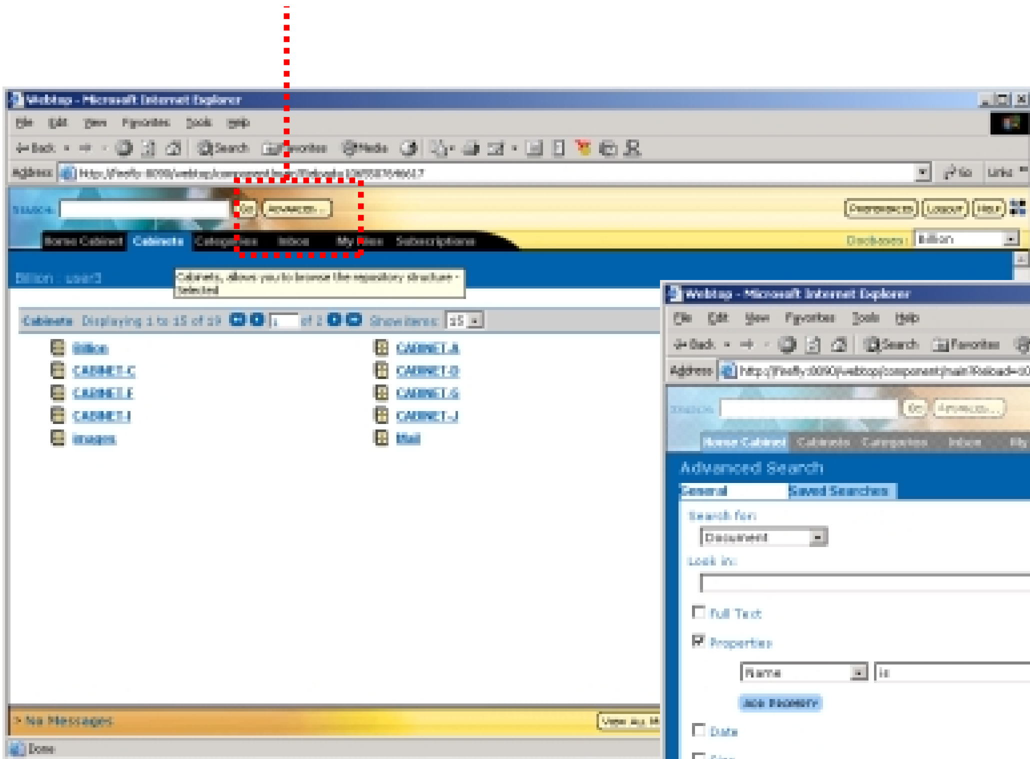
AdvSearch.java (abridged , only “modified” method shown):

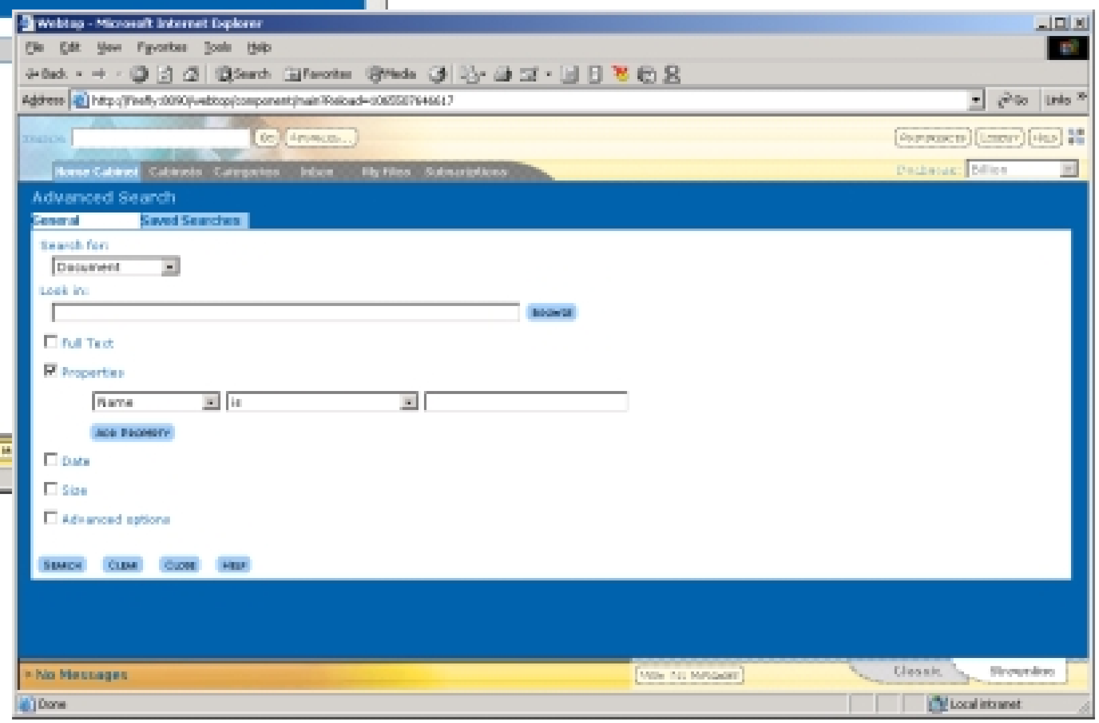
```
/**
 * Really do the search. The implementation nests the basic search component to do the search.
 *
 * @param args       The parameters collected from the UI controls
 * @param context    The current context
 */

protected void doSearch(ArgumentList args, Context context)
{

     System.out.println(args.get("query"));

     String s1 = args.get("query");
     String s9 = "";

     String find = "lower(";
     String find2 = ")";

     int first = 0;
     int last = 0;
     while(true){
         last = s1.indexOf(find,first);
         if( last == -1 )
             break;
         s9 = s9 + s1.substring(first,last);
         first = last + find.length();
         last = s1.indexOf(find2,first);
         s9 = s9 + s1.substring(first,last);
         first = last + 1;
     }
     if (first > 0)
         s9 = s9 + s1.substring(first);
     else
         s9 = s1;
```

```
        Panel advPanel = (Panel)getControl(ADV_PANEL);
        if (advPanel.isVisible())
        {

            Checkbox forceOrderCheck = (Checkbox)getControl(FORCE_ORDER_CHECK);
            if (forceOrderCheck.isEnabled() == true && forceOrderCheck.getValue() == true)
            {
                s9 = s9 + " ENABLE(FORCE_ORDER)";
            }
        }

        System.out.println("Stripping 'lower()'...");
        System.out.println(s9);
        args.replace("query", s9);


        setComponentReturnJump("search", args, context);

    }
```

Appendix O: User Interface Screens for non-prefix ID search

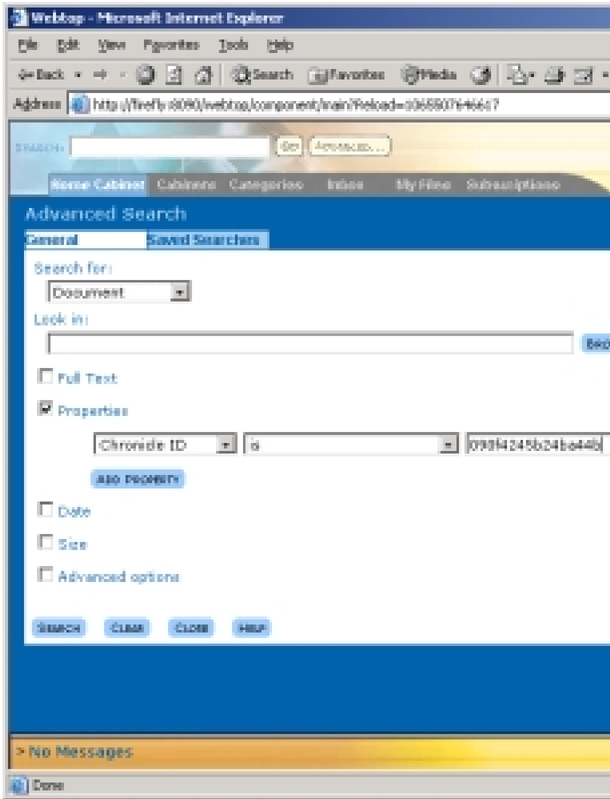
Step 1: Select “advanced search” from the “cabinets” screen

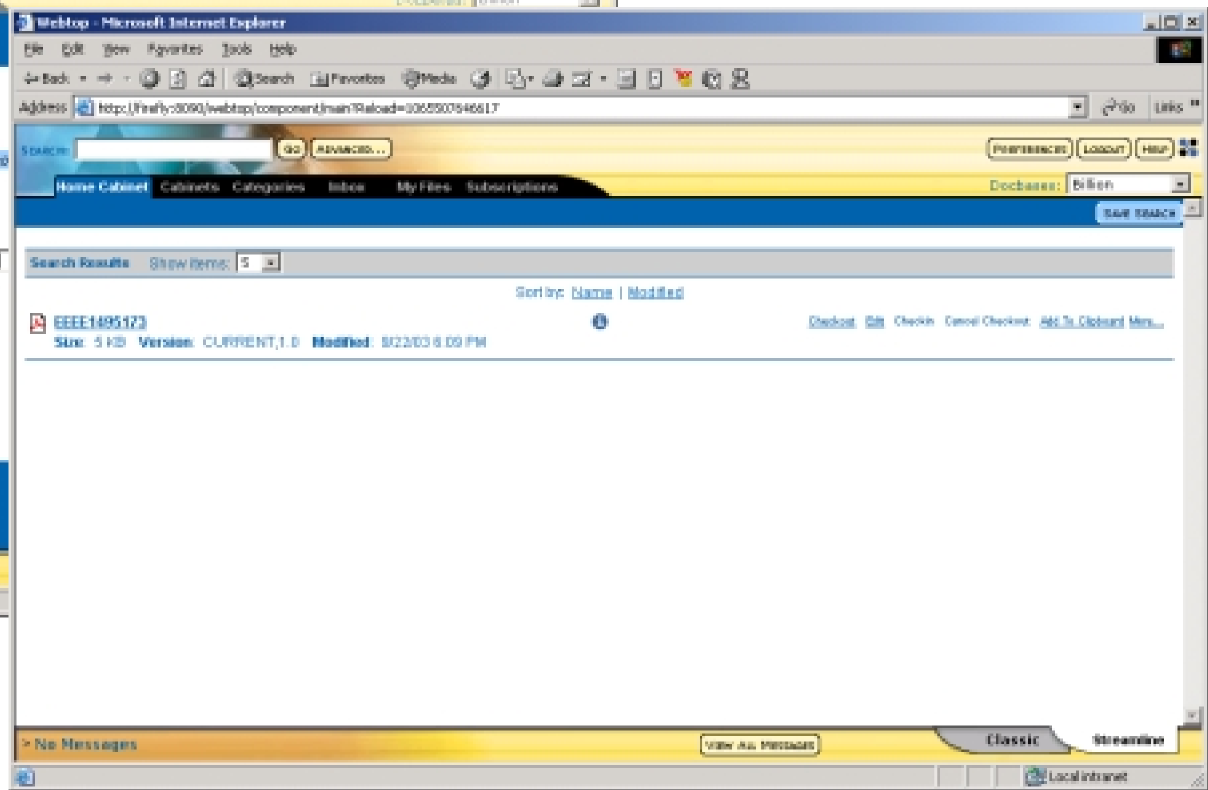




This will display the initial advanced search screen.

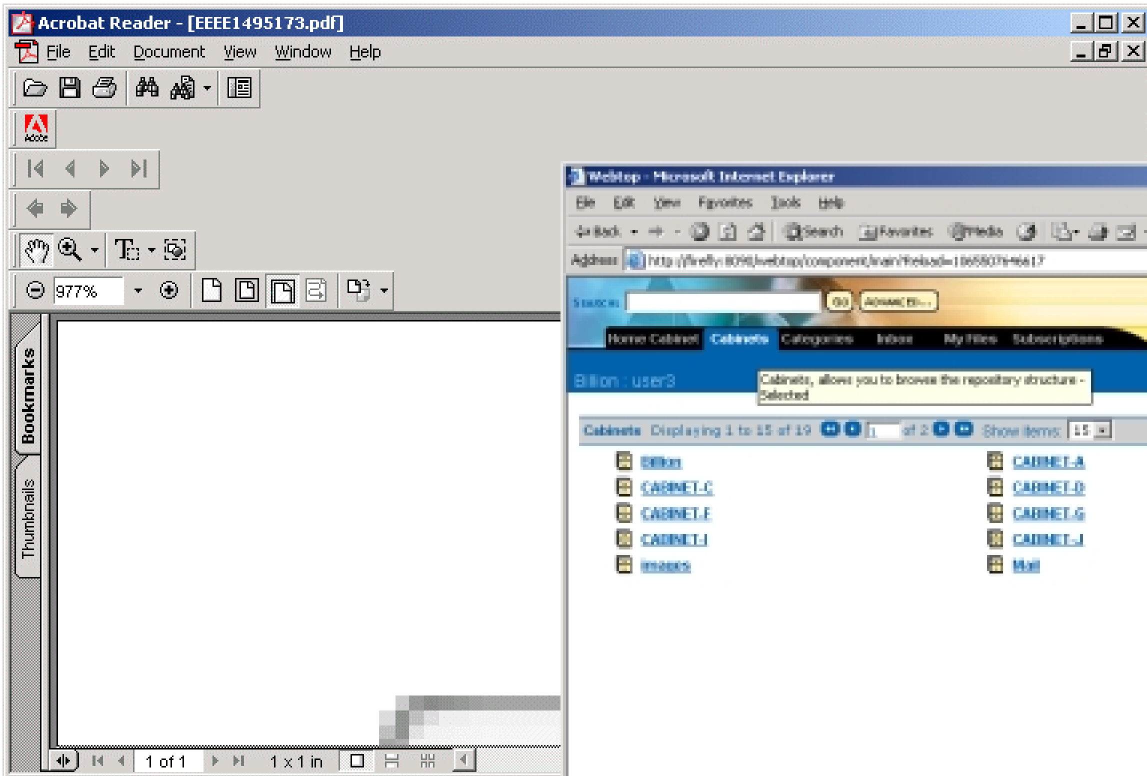
Step 2: The user fills in the various drop-downs and search parameters:



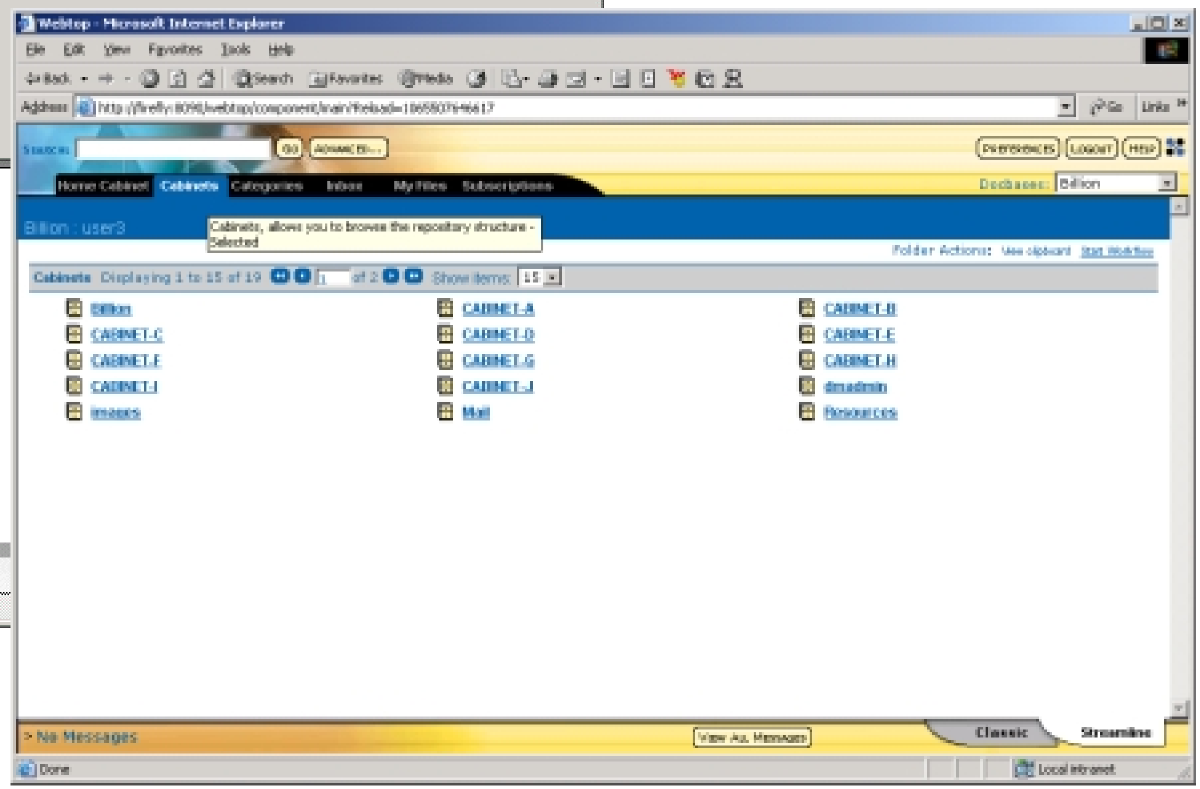


Step 3: Query is started and results are displayed.

Step 4: Document is viewed



Step 5: User moves back to “cabinet” display



Appendix P: Output from Index Rebuild of single partition

```
SQL> select to_char(sysdate,'dd.mm.yyyy hh24:mi:ss') date1 from dual;

DATE1
-------------------
25.09.2003 02:34:06

Elapsed: 00:00:00.00
SQL> ALTER INDEX  BILLION.D_1F0F424580000109 REBUILD PARTITION P96 NOLOGGING PARALLEL (DEGREE 4);

Index altered.

Elapsed: 00:03:25.08

SQL> ALTER INDEX  BILLION.D_1F0F424580000010 REBUILD PARTITION P96 NOLOGGING PARALLEL (DEGREE 4);

Index altered.

Elapsed: 00:02:33.07

SQL> ALTER INDEX  BILLION.D_1F0F424580000108  REBUILD PARTITION P96 NOLOGGING PARALLEL (DEGREE 4) ;

Index altered.

Elapsed: 00:02:35.02

SQL> ALTER INDEX  BILLION.D_1F0F42458000000E  REBUILD PARTITION P96 NOLOGGING PARALLEL (DEGREE 4);

Index altered.

Elapsed: 00:02:41.03

SQL> ALTER INDEX  BILLION.D_1F0F42458000002A  REBUILD PARTITION P96 NOLOGGING PARALLEL (DEGREE 4);

Index altered.

Elapsed: 00:02:00.08

SQL> ALTER INDEX  BILLION.D_1F0F42458000000F  REBUILD PARTITION P96 NOLOGGING PARALLEL (DEGREE 4);

Index altered.

Elapsed: 00:02:40.02
```

```
SQL> ALTER INDEX  BILLION.D_1F0F42458000002F  REBUILD PARTITION P96 NOLOGGING PARALLEL (DEGREE 4);

Index altered.

Elapsed: 00:03:15.08

SQL> ALTER INDEX  BILLION.D_1F0F424580000032  REBUILD PARTITION P96 NOLOGGING PARALLEL (DEGREE 4);

Index altered.

Elapsed: 00:02:18.04

SQL> ALTER INDEX  BILLION.D_1F0F42458000003C  REBUILD PARTITION P96 NOLOGGING PARALLEL (DEGREE 4);

Index altered.

Elapsed: 00:02:35.06

SQL> ALTER INDEX  BILLION.ADDED1  REBUILD PARTITION P96 NOLOGGING PARALLEL (DEGREE 4);

Index altered.

Elapsed: 00:02:59.07

SQL> ALTER INDEX  BILLION.D_1F0F424580000034 REBUILD PARTITION P96_6 NOLOGGING PARALLEL (DEGREE 4);

Index altered.

Elapsed: 00:02:16.02

SQL> ALTER INDEX  BILLION.D_1F0F424580000038 REBUILD PARTITION P96_6 NOLOGGING PARALLEL (DEGREE 4);

Index altered.

Elapsed: 00:02:40.00

SQL> ALTER INDEX  BILLION.D_1F0F424580000160 REBUILD PARTITION P96_6 NOLOGGING PARALLEL (DEGREE 4);

Index altered.

Elapsed: 00:02:13.08

SQL> ALTER INDEX  BILLION.D_1F0F424580000005 REBUILD PARTITION P96_6 NOLOGGING PARALLEL (DEGREE 4);

Index altered.

Elapsed: 00:02:24.09
```

```
SQL>
SQL> ALTER INDEX  BILLION.D_1F0F42458000015F REBUILD PARTITION P96_6 NOLOGGING PARALLEL (DEGREE 4);

Index altered.

Elapsed: 00:02:00.04
SQL>

SQL> ALTER INDEX  BILLION.D_1F0F424580000501 REBUILD PARTITION P96 NOLOGGING PARALLEL (DEGREE 4);

Index altered.

Elapsed: 00:03:44.02

SQL> ALTER INDEX  BILLION.ADDED4 REBUILD PARTITION P96 NOLOGGING PARALLEL (DEGREE 4);

Index altered.

Elapsed: 00:04:16.02

SQL> ALTER INDEX  BILLION.D_1F0F424580000500 REBUILD PARTITION P96 NOLOGGING PARALLEL (DEGREE 4);

Index altered.

Elapsed: 00:01:42.07

SQL> ALTER INDEX  BILLION.ADDED2 REBUILD PARTITION P96 NOLOGGING PARALLEL (DEGREE 4);

Index altered.

Elapsed: 00:02:34.08

SQL> ALTER INDEX  BILLION.ADDED3  REBUILD PARTITION P96 NOLOGGING PARALLEL (DEGREE 4);

Index altered.

Elapsed: 00:01:50.01

SQL> ALTER INDEX  BILLION.D_1F0F424580000145 REBUILD PARTITION P96 NOLOGGING PARALLEL (DEGREE 4);

Index altered.

Elapsed: 00:01:14.02

SQL> ALTER INDEX  BILLION.DMI_OBJECT_TYPE_UNIQUE REBUILD PARTITION P96 NOLOGGING PARALLEL (DEGREE 4);

Index altered.
```

```
Elapsed: 00:01:37.03
SQL> select to_char(sysdate,'dd.mm.yyyy hh24:mi:ss') date1 from dual;

DATE1
-------------------
25.09.2003 03:29:48

Elapsed: 00:00:00.00
```

Appendix R: Sample Bulk loader output for largest and smallest tables (25 million objects)

******* largest table *******

```
SQL*Loader: Release 9.2.0.1.0 - Production on Tue Sep 23 05:03:27 2003

Copyright (c) 1982, 2002, Oracle Corporation.  All rights reserved.

Control File:   dm_sysobject_s_c33.ctl
Data File:      I:\bulk load data files2\dm_sysobject_s_d33.dat
  Bad File:     dm_sysobject_s_d33.bad
  Discard File:  none specified

 (Allow all discards)

Number to load: ALL
Number to skip: 0
Errors allowed: 50
Continuation:    none specified
Path used:      Direct

Table DM_SYSOBJECT_S, loaded from every logical record.
Insert option in effect for this table: APPEND
TRAILING NULLCOLS option in effect

   Column Name                  Position   Len  Term Encl Datatype
------------------------------ ---------- ----- ---- ---- ---------------------
R_OBJECT_ID                         FIRST     *   ,       CHARACTER
OBJECT_NAME                          NEXT     *   ,       CHARACTER
R_OBJECT_TYPE                        NEXT     *   ,       CHARACTER
TITLE                                NEXT     *   ,       CHARACTER
SUBJECT                              NEXT     *   ,       CHARACTER
A_APPLICATION_TYPE                   NEXT     *   ,       CHARACTER
A_STATUS                             NEXT     *   ,       CHARACTER
R_CREATION_DATE                      NEXT     *   ,       DATE MM/DD/YYYY HH:MI:SS AM
R_MODIFY_DATE                        NEXT     *   ,       DATE MM/DD/YYYY HH:MI:SS AM
R_MODIFIER                           NEXT     *   ,       CHARACTER
R_ACCESS_DATE                        NEXT     *   ,       DATE MM/DD/YYYY HH:MI:SS AM
A_IS_HIDDEN                          NEXT     *   ,       CHARACTER
I_IS_DELETED                         NEXT     *   ,       CHARACTER
A_RETENTION_DATE                     NEXT     *   ,       DATE MM/DD/YYYY HH:MI:SS AM
A_ARCHIVE                            NEXT     *   ,       CHARACTER
A_COMPOUND_ARCHITECTURE              NEXT     *   ,       CHARACTER
A_LINK_RESOLVED                      NEXT     *   ,       CHARACTER
```

```
I_REFERENCE_CNT                     NEXT    *    ,      CHARACTER
I_HAS_FOLDER                        NEXT    *    ,      CHARACTER
R_LINK_CNT                          NEXT    *    ,      CHARACTER
R_LINK_HIGH_CNT                     NEXT    *    ,      CHARACTER
R_ASSEMBLED_FROM_ID                 NEXT    *    ,      CHARACTER
R_FRZN_ASSEMBLY_CNT                 NEXT    *    ,      CHARACTER
R_HAS_FRZN_ASSEMBLY                 NEXT    *    ,      CHARACTER
RESOLUTION_LABEL                    NEXT    *    ,      CHARACTER
R_IS_VIRTUAL_DOC                    NEXT    *    ,      CHARACTER
I_CONTENTS_ID                       NEXT    *    ,      CHARACTER
A_CONTENT_TYPE                      NEXT    *    ,      CHARACTER
R_PAGE_CNT                          NEXT    *    ,      CHARACTER
R_CONTENT_SIZE                      NEXT    *    ,      CHARACTER
A_FULL_TEXT                         NEXT    *    ,      CHARACTER
A_STORAGE_TYPE                      NEXT    *    ,      CHARACTER
I_CABINET_ID                        NEXT    *    ,      CHARACTER
OWNER_NAME                          NEXT    *    ,      CHARACTER
OWNER_PERMIT                        NEXT    *    ,      CHARACTER
GROUP_NAME                          NEXT    *    ,      CHARACTER
GROUP_PERMIT                        NEXT    *    ,      CHARACTER
WORLD_PERMIT                        NEXT    *    ,      CHARACTER
I_ANTECEDENT_ID                     NEXT    *    ,      CHARACTER
I_CHRONICLE_ID                      NEXT    *    ,      CHARACTER
I_LATEST_FLAG                       NEXT    *    ,      CHARACTER
R_LOCK_OWNER                        NEXT    *    ,      CHARACTER
R_LOCK_DATE                         NEXT    *    ,      DATE MM/DD/YYYY HH:MI:SS AM
R_LOCK_MACHINE                      NEXT    *    ,      CHARACTER
LOG_ENTRY                           NEXT    *    ,      CHARACTER
I_BRANCH_CNT                        NEXT    *    ,      CHARACTER
I_DIRECT_DSC                        NEXT    *    ,      CHARACTER
R_IMMUTABLE_FLAG                    NEXT    *    ,      CHARACTER
R_FROZEN_FLAG                       NEXT    *    ,      CHARACTER
R_HAS_EVENTS                        NEXT    *    ,      CHARACTER
ACL_DOMAIN                          NEXT    *    ,      CHARACTER
ACL_NAME                            NEXT    *    ,      CHARACTER
A_SPECIAL_APP                       NEXT    *    ,      CHARACTER
I_IS_REFERENCE                      NEXT    *    ,      CHARACTER
R_CREATOR_NAME                      NEXT    *    ,      CHARACTER
R_IS_PUBLIC                         NEXT    *    ,      CHARACTER
R_POLICY_ID                         NEXT    *    ,      CHARACTER
R_RESUME_STATE                      NEXT    *    ,      CHARACTER
R_CURRENT_STATE                     NEXT    *    ,      CHARACTER
R_ALIAS_SET_ID                      NEXT    *    ,      CHARACTER
A_CATEGORY                          NEXT    *    ,      CHARACTER
LANGUAGE_CODE                       NEXT    *    ,      CHARACTER
A_IS_TEMPLATE                       NEXT    *    ,      CHARACTER
A_CONTROLLING_APP                   NEXT    *    ,      CHARACTER
R_FULL_CONTENT_SIZE                 NEXT    *    ,      CHARACTER
```

```
A_IS_SIGNED                        NEXT      *   ,      CHARACTER
I_IS_REPLICA                       NEXT      *   ,      CHARACTER
I_VSTAMP                           NEXT      *   ,      CHARACTER

The following index(es) on table DM_SYSOBJECT_S were processed:
index BILLION.D_1F0F424580000108 partition P80 was made unusable due to:
SKIP_INDEX_MAINTENANCE option requested
index BILLION.D_1F0F424580000108 partition P81 was made unusable due to:
SKIP_INDEX_MAINTENANCE option requested
index BILLION.D_1F0F424580000108 partition P82 was made unusable due to:
SKIP_INDEX_MAINTENANCE option requested
index BILLION.D_1F0F42458000000E partition P80 was made unusable due to:
SKIP_INDEX_MAINTENANCE option requested
index BILLION.D_1F0F42458000000E partition P81 was made unusable due to:
SKIP_INDEX_MAINTENANCE option requested
index BILLION.D_1F0F42458000000E partition P82 was made unusable due to:
SKIP_INDEX_MAINTENANCE option requested
index BILLION.D_1F0F42458000002A partition P80 was made unusable due to:
SKIP_INDEX_MAINTENANCE option requested
index BILLION.D_1F0F42458000002A partition P81 was made unusable due to:
SKIP_INDEX_MAINTENANCE option requested
index BILLION.D_1F0F42458000002A partition P82 was made unusable due to:
SKIP_INDEX_MAINTENANCE option requested
index BILLION.D_1F0F42458000000F partition P80 was made unusable due to:
SKIP_INDEX_MAINTENANCE option requested
index BILLION.D_1F0F42458000000F partition P81 was made unusable due to:
SKIP_INDEX_MAINTENANCE option requested
index BILLION.D_1F0F42458000000F partition P82 was made unusable due to:
SKIP_INDEX_MAINTENANCE option requested
index BILLION.D_1F0F42458000002F partition P80 was made unusable due to:
SKIP_INDEX_MAINTENANCE option requested
index BILLION.D_1F0F42458000002F partition P81 was made unusable due to:
SKIP_INDEX_MAINTENANCE option requested
index BILLION.D_1F0F42458000002F partition P82 was made unusable due to:
SKIP_INDEX_MAINTENANCE option requested
index BILLION.D_1F0F424580000032 partition P80 was made unusable due to:
SKIP_INDEX_MAINTENANCE option requested
index BILLION.D_1F0F424580000032 partition P81 was made unusable due to:
SKIP_INDEX_MAINTENANCE option requested
index BILLION.D_1F0F424580000032 partition P82 was made unusable due to:
SKIP_INDEX_MAINTENANCE option requested
index BILLION.D_1F0F42458000003C partition P80 was made unusable due to:
SKIP_INDEX_MAINTENANCE option requested
index BILLION.D_1F0F42458000003C partition P81 was made unusable due to:
SKIP_INDEX_MAINTENANCE option requested
index BILLION.D_1F0F42458000003C partition P82 was made unusable due to:
SKIP_INDEX_MAINTENANCE option requested
index BILLION.ADDED1 partition P80 was made unusable due to:
```

```
SKIP_INDEX_MAINTENANCE option requested
index BILLION.ADDED1 partition P81 was made unusable due to:
SKIP_INDEX_MAINTENANCE option requested
index BILLION.ADDED1 partition P82 was made unusable due to:
SKIP_INDEX_MAINTENANCE option requested

Table DM_SYSOBJECT_S:
  25165824 Rows successfully loaded.
  0 Rows not loaded due to data errors.
  0 Rows not loaded because all WHEN clauses were failed.
  0 Rows not loaded because all fields were null.

  Date conversion cache disabled due to overflow (default size: 1000)

  Partition P80: 9859456 Rows loaded.
  Partition P81: 10000000 Rows loaded.
  Partition P82: 5306368 Rows loaded.

Bind array size not used in direct path.
Column array  rows :    5000
Stream buffer bytes:  256000
Read   buffer bytes: 1048576

Total logical records skipped:          0
Total logical records read:      25165824
Total logical records rejected:         0
Total logical records discarded:        0
Total stream buffers loaded by SQL*Loader main thread:    10506
Total stream buffers loaded by SQL*Loader load thread:    42021

Run began on Tue Sep 23 05:03:27 2003
Run ended on Tue Sep 23 05:27:40 2003

Elapsed time was:     00:24:12.67
CPU time was:         00:14:42.64
```

***** smallest table *****

```
SQL*Loader: Release 9.2.0.1.0 - Production on Tue Sep 23 05:36:37 2003

Copyright (c) 1982, 2002, Oracle Corporation.  All rights reserved.

Control File:   dm_document_s_c33.ctl
Data File:      I:\bulk load data files2\dm_document_s_d33.dat
  Bad File:     dm_document_s_d33.bad
  Discard File:  none specified
  (Allow all discards)
```

```
Number to load: ALL
Number to skip: 0
Errors allowed: 50
Continuation:    none specified
Path used:      Direct

Table DM_DOCUMENT_S, loaded from every logical record.
Insert option in effect for this table: APPEND
TRAILING NULLCOLS option in effect

   Column Name                  Position   Len  Term Encl Datatype
------------------------------ ---------- ----- ---- ---- ---------------------
R_OBJECT_ID                         FIRST     *   ,       CHARACTER

The following index(es) on table DM_DOCUMENT_S were processed:
index BILLION.D_1F0F424580000145 partition P80 was made unusable due to:
SKIP_INDEX_MAINTENANCE option requested
index BILLION.D_1F0F424580000145 partition P81 was made unusable due to:
SKIP_INDEX_MAINTENANCE option requested
index BILLION.D_1F0F424580000145 partition P82 was made unusable due to:
SKIP_INDEX_MAINTENANCE option requested

Table DM_DOCUMENT_S:
  25165824 Rows successfully loaded.
  0 Rows not loaded due to data errors.
  0 Rows not loaded because all WHEN clauses were failed.
  0 Rows not loaded because all fields were null.

  Partition P80: 9859456 Rows loaded.
  Partition P81: 10000000 Rows loaded.
  Partition P82: 5306368 Rows loaded.

Bind array size not used in direct path.
Column array  rows :    5000
Stream buffer bytes:  256000
Read   buffer bytes: 1048576

Total logical records skipped:          0
Total logical records read:      25165824
Total logical records rejected:         0
Total logical records discarded:        0
Total stream buffers loaded by SQL*Loader main thread:     5184
Total stream buffers loaded by SQL*Loader load thread:        0

Run began on Tue Sep 23 05:36:37 2003
Run ended on Tue Sep 23 05:38:27 2003
Elapsed time was:     00:01:50.49
CPU time was:         00:00:29.64
```