

# EMC Documentum xDB 9.0 Scalability Tests

*A Detailed Review*

---

## **Abstract**

This white paper presents test results on the scalability behavior of EMC<sup>®</sup> Documentum<sup>®</sup> xDB when the amount of data, document size, or number of users increases. This should not be interpreted as a performance test. However, it does give a good indication of the expected overall performance of the system.

May 2009

---

---

Copyright © 2009 EMC Corporation. All rights reserved.

EMC believes the information in this publication is accurate as of its publication date. The information is subject to change without notice.

THE INFORMATION IN THIS PUBLICATION IS PROVIDED “AS IS.” EMC CORPORATION MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WITH RESPECT TO THE INFORMATION IN THIS PUBLICATION, AND SPECIFICALLY DISCLAIMS IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Use, copying, and distribution of any EMC software described in this publication requires an applicable software license.

For the most up-to-date listing of EMC product names, see EMC Corporation Trademarks on [EMC.com](http://EMC.com)

All other trademarks used herein are the property of their respective owners.

H4662

---

## Table of Contents

<b>Executive summary .....</b>	<b>4</b>
<b>Introduction .....</b>	<b>4</b>
Audience .....	4
<b>Database size .....</b>	<b>4</b>
<b>Document size.....</b>	<b>5</b>
<b>Number of users .....</b>	<b>5</b>
<b>Indexed queries.....</b>	<b>6</b>
<b>Multi-threaded loading .....</b>	<b>7</b>
<b>Conclusion .....</b>	<b>8</b>

## Executive summary

EMC® Documentum® xDB is a high-performance, scalable native XML database that can be used in a variety of content-oriented applications, such as publishing, dynamic websites, archiving, information mash-ups, regulatory filings, collaboration, and knowledge management. Unlike relational databases, xDB allows database structures to be easily modified to adapt to changing information requirements. It also supports complex data structures that are not easily modeled in relational rows and columns. xDB is a powerful platform for the most complex and demanding applications, with a powerful, extensible development and runtime tool set, complete support for the XQuery language, and tight integration with the leading enterprise content management system.

## Introduction

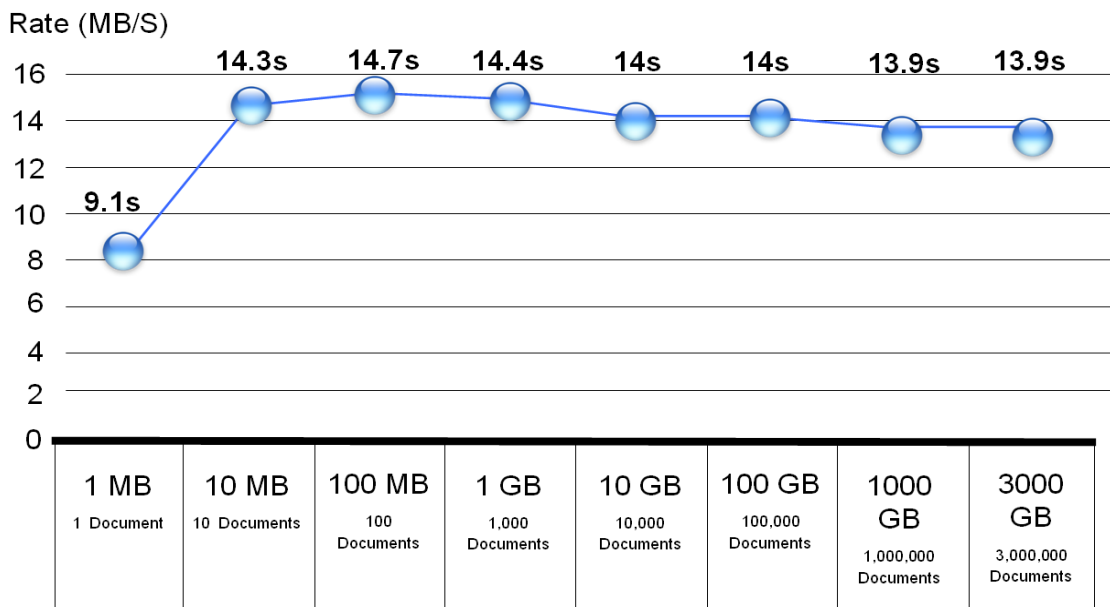
To assist customers who are evaluating xDB, the xDB engineering team has conducted a series of tests that indicate how xDB will scale for a variety of common functions, including data loading, concurrency under load, and indexed retrieval. This white paper presents the findings of these tests. The performance measures from these tests indicate excellent scalability with minimal resource requirements. xDB scales well when faced with a variety of sizing factors including database size, document size, total load size, and number of concurrent users. These tests were run on a 3.2 GHz Intel Core 2 Quad CPU (4 cores) with 8 GB RAM, using Sun Java SDK 1.6.0\_12. Although xDB supports multiple concurrent users, most tests are single threaded unless specified otherwise. The single-threaded tests do not benefit from more than one core.

## Audience

This white paper is intended for EMC technical sales staff to assist customers who are evaluating xDB performance.

## Database size

This test loads multiple documents of 1 MB each into the database, committing each one individually. This test shows that the load rate is independent of the number of documents loaded.

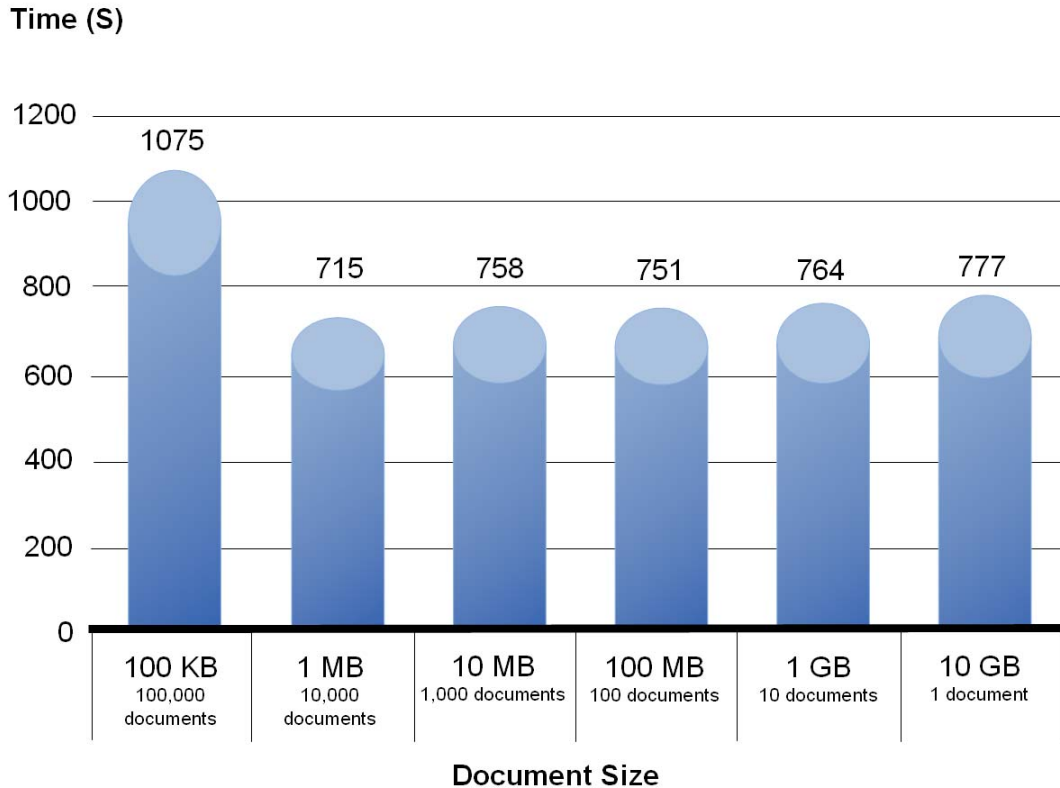


**Figure 1. Load rate is independent of the number of documents loaded**

---

## Document size

This test loads 10 GB of data into the database, using documents of sizes ranging from 100 KB to 10 GB. This test shows that the load time is independent of the document size for a given overall database size.

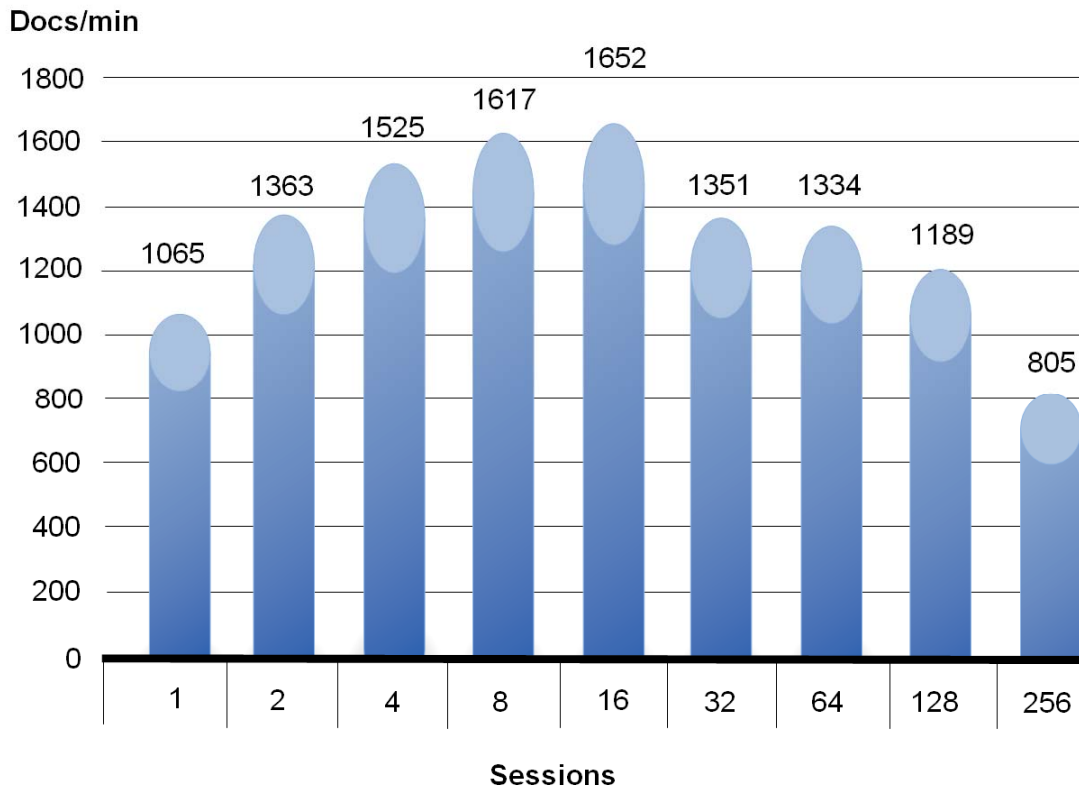


**Figure 2. Load time is independent of the document size**

The longer time for small documents is due to each document being committed separately. This has to flush the database log to the disk. However, using a single large transaction instead of 100,000 transactions, the first test of loading 100,000 documents of 100 KB takes only 781 seconds, which is comparable to the other results.

## Number of users

For this test, multiple concurrent sessions retrieve a random document and serialize it. The database contains 3 million documents of 1 MB each for a total of 3 TB of data. Using random documents ensures that the document retrieved is likely not to be in any cache.



**Figure 3. Document retrieval rates vs. number of concurrent sessions**

The number of documents per minute goes up and down due to two effects that work in opposite directions:

- With more sessions, there is more concurrency: Disks can serve requests concurrently and CPU usage can be done concurrently with disk requests. This increases the throughput.
- With only a few concurrent sessions, the blocks of a document are read sequentially from disk. With many concurrent sessions, there is usually another session that has read a block from a disk before the next block in a particular document is read. This causes the disk access pattern to become random, which is slower than sequential, because more seeks have to be done.

Note that for typical applications, users will only submit a request every so many seconds. So 256 concurrent sessions represents many thousands of simultaneous users.

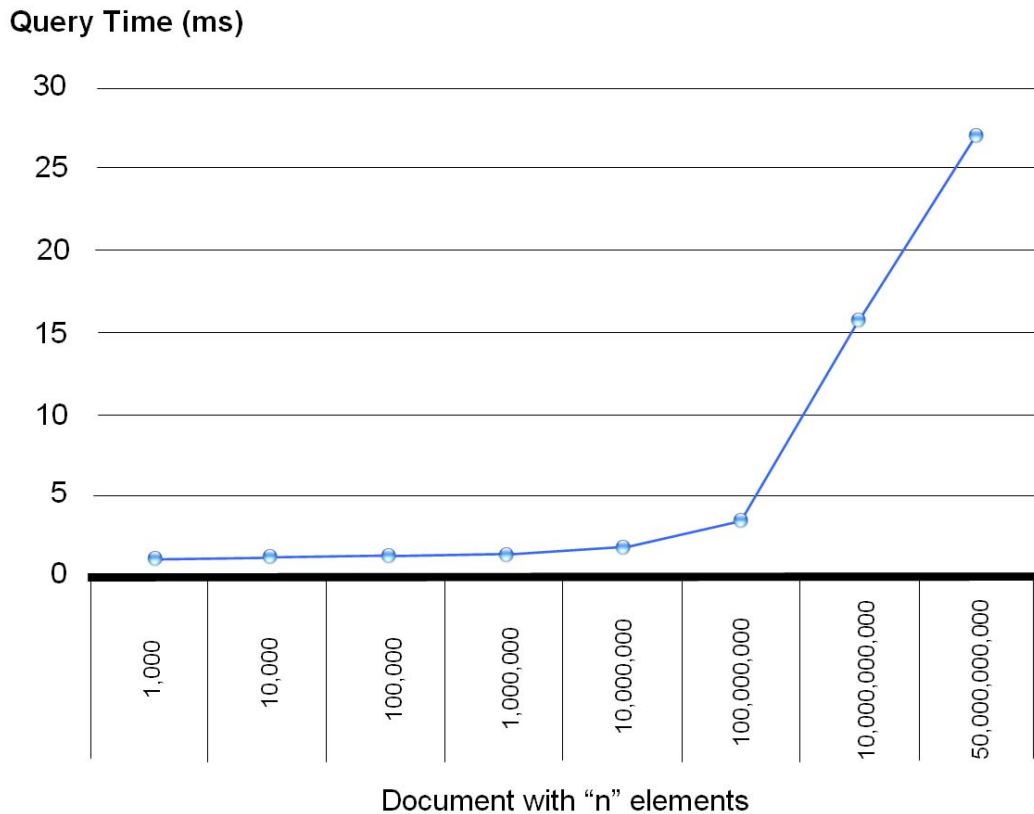
## Indexed queries

In this test we create a document like the following with a variable number of elements:

```
<root>
  <element id="0"/>
  <element id="1"/>
  <element id="2"/>
  ...
</root>
```

---

Using an index of type long on the id attribute, we execute the XQuery query `//element[@id=N]`, using a random value for *N* to avoid caching effects. We serialize the result of the query to make sure we access the data itself, not only the index.



**Figure 4. Indexed retrieval vs. number of elements in a document**

Up to 10 million elements, the document is contained completely in the server cache, and retrieval is very fast. With large documents, the result is almost never in the cache and has to be retrieved from disk, which takes several milliseconds. The document with 100 million elements does not fit in the cache completely, but the randomly queried element has a large chance of being in the cache. When the document becomes very large, the index will be deeper, requiring more disk accesses. Also, as the disk becomes more full, average seek times will increase.

## Multi-threaded loading

In this test we loaded documents of four different sizes (for example, one can think of four types of high volume transaction logs from a bank). The test shows the expected load time of xDB for a diverse dataset.

---

**Table 1. Multi-threaded loading tests**

Size (bytes)	Number	Total data (GB)
5,008	10,000,000	46.6
48,889	1,000,000	45.5
198,331	100,000	18.5
5,631,417	10,000	52.4
<b>Total</b>	<b>11,110,000</b>	<b>163</b>

The data was loaded in 500 threads, where each thread loaded 1/500th of the data in random order. Test results are as follows

Total clock time (hh:mm:ss)	2:47:20
Average load rate	13.6 MB/s
Size of datafile	133.7 GB
Average CPU usage	2.47 (out of 4)

The average load rate of 13.6 MB/s is consistent with the single-threaded test shown earlier, indicating that xDB scales well even with a concurrent load. Note, too, that the total storage required is only 81 percent of the original data size (133.7 GB vs. 163 GB), demonstrating the efficiency of xDB's internal storage format.

## Conclusion

The tests presented in this paper are a measure of the excellent scalability of xDB for a variety of common functions. Although actual performance will vary depending on a number of implementation-specific factors, these tests can be a useful starting point for sizing xDB for your application.

For further information about EMC Documentum xDB please contact Jerry Silver, Product Marketing Manager, at [silver\\_jerry@emc.com](mailto:silver_jerry@emc.com).