

# ***NSTL***

*A CMP Company*

**NSTL Final Report**

**Defragmentation  
Performance Testing**

November, 1999

## TABLE OF CONTENTS

<b>EXECUTIVE SUMMARY</b> .....	<b>3</b>
<b>TESTING ENVIRONMENT</b> .....	<b>5</b>
PREPARING THE SYSTEMS.....	5
HOW NSTL MEASURED FRAGMENTATION .....	5
APPLICATIONS TESTING.....	6
<i>Excel 2000(Client Side Test)</i> .....	6
<i>Microsoft Exchange 5.5 and Outlook 2000 (Client and Server Side tests)</i> .....	8
<i>SQL Server 7.0 (Server Side Test)</i> .....	10
<b>ABOUT THE TESTS</b> .....	<b>13</b>
EXCEL 2000 TEST.....	13
SQL SERVER 7.0 TEST.....	13
OUTLOOK 2000/EXCHANGE 5.5 TEST.....	13
<b>RESULTS</b> .....	<b>14</b>
EXCEL 2000 BENCHMARKS .....	14
OUTLOOK 2000/EXCHANGE 5.5 BENCHMARKS.....	15
<i>Configuration 1</i> .....	15
<i>Configuration 2</i> .....	16
SQL SERVER 7.0 BENCHMARKS .....	17
<i>Configuration1</i> .....	17
<i>Configuration1 (Continued)</i> .....	18
<i>Configuration1 (Continued)</i> .....	19
<i>Configuration2</i> .....	20
<i>Configuration2 (Continued)</i> .....	21
<i>Configuration2 (Continued)</i> .....	22
<b>PERFORMANCE TEST CONCLUSIONS</b> .....	<b>23</b>
EXCEL 2000.....	23
SQL SERVER 7.0.....	23
OUTLOOK 2000/EXCHANGE 5.5.....	24
<b>SYSTEM COMPONENTS</b> .....	<b>26</b>
CONFIGURATION #1.....	26
CONFIGURATION #2.....	27

This report was prepared by NSTL under contract for Executive Software. NSTL does not guarantee the accuracy, adequacy or completeness of the services provided. NSTL MAKES NO WARRANTIES, EXPRESSED OR IMPLIED, AS TO RESULTS TO BE OBTAINED BY ANY PERSON OR ENTITY FROM USE OF THE CONTENTS OF THIS REPORT. NSTL MAKES NO EXPRESS OR IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE OF ANY PRODUCT MENTIONED IN THIS REPORT.

---

## EXECUTIVE SUMMARY

---

NSTL is the leading independent hardware and software testing organization in the microcomputer industry, dedicated to providing high quality services and test tools to the PC community. NSTL has extensive experience developing and conducting objective tests to assess new and existing products for compatibility, performance, usability, acceptance (bug) testing, and BIOS evaluation. NSTL's proficiency and thoroughness provide clients with a high quality, cost-effective means to conduct testing. More information on NSTL is available through the World Wide Web at <http://www.nstl.com>.

NSTL, under an agreement with Executive Software, conducted defragmentation performance testing using Diskeeper® 5.0, the most commonly used disk defragmentation program for Windows NT<sup>1</sup>. The main goal of testing was to document the effects of defragmentation on system performance and show the increase in performance when a system has been defragmented. Diskeeper's defragmentation tool was used to defragment and analyze all systems before and during testing. More information on Diskeeper is available through the World Wide Web at <http://www.diskeeper.com>.

Benchmark testing was conducted on four computer systems, running Microsoft® Windows® 2000 RC2 build 2128. Two systems were running Microsoft Window 2000 Professional RC2 build 2128 and two were running Microsoft Windows Windows® 2000 Server RC2 build 2128. Two of the most common real-world system configurations were tested; these configurations were based on independent surveys conducted on 6,000 NT system managers. This survey was performed and provided by Broadcasters Network International.

NSTL performed the performance tests using four applications: Microsoft Excel 2000, SQL Server 7.0, Microsoft Outlook 2000, and Microsoft Exchange 5.5.

### Performance Results Summary

NSTL's results show that a system that was defragmented would perform benchmarks quicker than a system that was fragmented.

The results showed that the workstation in Configuration #1, running Excel and Outlook, showed an increase in performance of 219.6% after defragmentation.

The server in Configuration #1, running Exchange and SQL Server 7.0, showed an increase in performance of 83.5% after defragmentation.

The workstation in Configuration #2, running Excel and Outlook, showed an increase in performance of 85.5% after defragmentation.

The server in Configuration #2, running Exchange and SQL Server 7.0, showed an increase in performance of 61.9% after defragmentation.

---

<sup>1</sup> According to *Microsoft Windows NT Server Resource Guide* and *Running Microsoft Windows Server NT 4.0*.

The Excel 2000 test repeatedly opened and saved several Excel files to the fragmented (or defragmented) partition. The four files that it used (file1, file2, file3 and file4) varied in size from 5MB to 20MB. These large files contained many formulas that were also auto-calculated when the spreadsheet opened. File3 opened itself and used data from file1 and file2.

The SQL Server 7.0 test combined two different types of activities on the database: queries and some database maintenance. The database maintenance was probably the least important, as this type of activity is not usually very common. However, it demonstrated the differences in doing these types of activities on fragmented vs. defragmented systems. The query section executed a simple query on the database that simulated the types of queries typically run by users. The two queries chosen only read from the database, but displayed results from several of the tables.

The Outlook2000/Exchange 5.5 test determined the effects of fragmentation on the personal folder database used by Microsoft Outlook. Several tests were run including: opening 50 messages simultaneously; moving messages from the inbox to a separate folder; opening (and displaying to: from: subject: and date:) a large subfolder; a full text search of all messages in a folder for a specific string; and a filter that displayed all messages in a folder that contained an attachment. Each of these tests was executed on the system when the personal folder file was fragmented, and when it was defragmented.

The Microsoft Exchange (v 5.5) test was identical to the Outlook test (indeed Outlook was used as the client). The difference was that all mail existed on the Exchange database (as opposed to locally) and the Exchange database itself was fragmented.

---

## TESTING ENVIRONMENT

---

Benchmark testing was conducted on four computer systems based on common real-world system configurations as found by surveys conducted on 6,000 NT system managers. Configuration #1: The first configuration consisted of a Compaq Deskpro EP Pentium II 400MHz configured with 128MB of RAM with a 4GB hard drive (workstation), and a Compaq Proliant 2500 Pentium PRO 200 configured with 64MB of RAM and two 4GB hard drives (server). Configuration #2: The second configuration consisted of a Compaq Proliant 2500 Pentium PRO 200 Dual Processor configured with 128MB of RAM, an Ultra Wide SCSI adapter, software RAID 5, with five 4GB hard drives (server) and a Compaq Deskpro Pentium II 266MHz configured with 96MB of RAM with one 2GB hard drive (workstation). All of the above systems contained 100Mbit network cards.

### Preparing the Systems

1. Windows 2000 Professional Release Candidate 2 Build 2128 – Professional was installed on both Compaq Deskpro systems. All defaults were accepted during the install. Each workstation's hard drive was partitioned to have a least two logical drives, the second logical drive was 1GB, and this was the test drive (D drive). Make a new directory on the C drive and label it "Testfiles." Copy all test files into this directory.
2. Compaq Server Setup Utility was used to configure the servers before installing the Operating System. The server for Configuration 2 was setup to use software RAID level 5. Windows 2000 Server Release Candidate 2 Build 2128 was installed on both Compaq Proliant Servers. All defaults were accepted during the install. Each server was setup with at least two logical drives, the second logical drive was 1GB, and this was the test drive (D drive).

### How NSTL Measured Fragmentation

NSTL used Executive Software's Diskeeper 5.0 defragmentation software to measure fragmentation. This utility was used simply because it was more feature enriched than the version included with Windows 2000, both Server and Professional. NSTL first defragmented the C drive using Diskeeper to ensure there was no fragmentation on this drive. Then the test drive (D drive) was analyzed using Diskeeper's analysis tool. Once the analysis was complete a report was viewed and saved and the level of fragmentation was recorded. (All reports are available for each test.) NSTL was supplied multiple versions of Diskeeper for testing. Beta version 5.0.333 was used for all Excel tests. Beta version 5.0.340 was used for the remaining tests. NSTL was assured by Executive Software the release version would be 5.0.340a.

Testing was conducted using, in most cases, a multiple-GB hard drive. This hard drive was split into two drives, one partition containing the OS (drive C) and one partition for testing (drive D). Prior to any tests, the D drive was formatted to ensure complete defragmentation.

During testing, NSTL's proprietary utility *Fragger* was used to create many small files (each 4-KB's in size). This same utility was then used to delete some of the recently created files in a controlled order. This process created a large amount of fragmented free space available on the drive. Once these steps were accomplished, installing the testing application and test files was performed.

The random levels of fragmentation from one test to another was further influenced by only two additional factors: 1) the size of the application being tested and all of the associated files for that application, and 2) the total amount of data installed, including application and data files. More data means more fragmentation. In addition, the average fragments per file are directly co-related with the average file size. This means the larger the file the more fragments.

## Applications Testing

### *Excel 2000(Client Side Test)*

1. Format the test drive to ensure defragmentation.
2. Install Microsoft Excel on the test drive (D drive), accepting all defaults, with one exception: install the application into the "Microsoft Office" directory on the test drive. Make sure the Office Assistant is not active.
3. Make a new directory on the test drive and label it "Testfiles." Copy all Excel test files into this directory.
4. Copy the "Personal.xls" file into the Microsoft Office\Office\Xlstart directory.
5. Execute Diskkeeper and defragment the C and D drive. (Save the "Report" for the D drive.)
6. Execute Excel. When testing is complete, the results will be placed on the C drive and named "out.out." Run the test three times, saving the results in between every test run. Save the results as Run1.txt, Run2.txt and Run3.txt. (These results should be placed on a floppy in a directory labeled "Nofrag" under a directory labeled "Excel.")
7. Format the test drive.
8. Use the NSTL proprietary utility *Fragger* to create empty files that reserve space on the test drive. Use the following syntax:  

```
Fragger -c -n 2500000 -s 4 -b 1000
```
9. Use the NSTL proprietary utility *Fragger* to delete approximately 50% of the empty files on the test drive. Use the following syntax:  

```
Fragger -d -n 2500000 -p 50 -b 1000 -e 1234 -t special
```
10. Install Microsoft Excel on the test drive (D drive), accepting all defaults, with one exception: install the application into "Microsoft Office" directory on the test drive. Make sure the Office Assistant is not active.
11. Make a new directory on the test drive and label it "Testfiles." Copy all test files into this directory.
12. Copy the "Personal.xls" file into the Microsoft Office\Office\Xlstart directory.
13. Execute Diskkeeper and defragment the C drive. Run the "Analyze" tool on the test drive. (Save the "Report" for the D drive.)
14. Execute Excel. When testing is complete, the results will be placed on the C drive and named "out.out." Run the test three times, saving the results in between every test run. Save the results as Run1.txt, Run2.txt and Run3.txt. (These results should be placed on a floppy in a directory labeled "Appfrag" under a directory labeled "Excel").
15. Format the test drive.
16. Use the NSTL proprietary utility *Fragger* to create empty files that reserve space on the test drive. Use the following syntax:  

```
Fragger -c -n 2500000 -s 4 -b 1000
```
17. Use the NSTL proprietary utility *Fragger* to delete approximately 50% of the empty files on the test drive. Use the following syntax:  

```
Fragger -d -n 2500000 -p 50 -b 1000 -e 1234 -t special
```
18. Change the Page File size: Right click on "My Computer" select "Properties" select "Advanced" then select "Performance Options." From under "Performance Options," select "Change." Make sure the C drive is highlighted, and then insert "0" in the "Initial Size" and the "Maximum Size" input boxes. Select "Set."
19. From the above location, make sure the test drive is highlighted and then insert "256" in the "Initial Size" and the "Maximum Size" input boxes. Select "Set." (Note: you may receive an error message – ignore this. Windows 2000 does not like the Memory size changed. This error message will not have any bearing on the test results.)

20. Install Microsoft Excel on the test drive (D drive), accepting all defaults, with one exception: install the application into "Microsoft Office" directory on the test drive. Make sure the Office Assistant is not active.
21. Make a new directory on the test drive and label it "Testfiles." Copy all test files into this directory.
22. Copy the "Personal.xls" file into the Microsoft Office\Office\Xlstart directory.
23. Execute Diskkeeper and defragment the C drive. Run the "Analyze" tool on the test drive. (Save the "Report" for the D drive.)
24. Execute Excel. When testing is complete, the results are placed on the C drive and named "out.out." Run the test three times, saving the results in between every test run. Save the results as Run1.txt, Run2.txt and Run3.txt. (These results should be placed on a floppy in a directory labeled "Allfrag" under a directory label Excel.)

## Microsoft Exchange 5.5 and Outlook 2000 (Client and Server Side tests)

1. Format the test drive on both the server and the workstation to ensure complete defragmentation.
2. Install Microsoft Exchange on the test drive of the server.
3. Setup up ten new user mail accounts in Exchange. (Use user01, user02, etc.)
4. Install Microsoft Outlook on the test drive of the workstation.
5. Copy the "Test PST" file from the "TestFiles" directory on the C drive of the server to the test drive under the Outlook folder.
6. Open Outlook, then select "Open a Mailbox." Point to the server for the mailbox. Do not open any mail.
7. Once the mailboxes are set up, you can start testing.
8. Use a stopwatch to time the events. Select the first 50 messages and then select "Open." Start timing when you select "open" and end timing when the last message is opened.
9. Select to move the inbox mail into a new folder. (The folder name is not important; we used "Test.") Start timing when the move starts and end when the pointer is back to the inactive state.
10. Select the Business folder. Start timing when you select this folder; stop timing when you can see the folder contents.
11. Search all messages for the text "word." Start timing after you select "Search," and stop timing when the magnifying glass stops moving in the lower right hand-corner.
12. Search all messages for attachments. Start timing after you select "Search," and stop timing when the magnifying glass stops moving in the lower right hand-corner.
13. Copy the inbox to a "PST" file. Go to "File," then "Export." Select "Inbox" to export the entire inbox. Start timing when you select "Export," and end timing when the pointer is back to the inactive state.
14. From the client side, select the first 50 messages, and then select "Open." Start timing from when you select "Open," and end timing when the last message is opened.
15. From the client side, select to move the inbox mail into a new folder. (The folder name is not important; we used "Test.") Start timing when the move starts, and end when the pointer is back to the inactive state.
16. From the client side, select the Business subdirectory. Start timing when you select this folder, stop timing when you can see the folder contents.
17. From the client side, search all messages for the text "word." Start timing after you select "Search," and stop timing when the magnifying glass stops moving in the lower right-hand corner.
18. From the client side, search all messages for attachments. Start timing after you select "Search," and stop timing when the magnifying glass stops moving in the lower right-hand corner.
19. Repeat steps 1 through 18 three times.
20. Format the test drive on both the server and the workstation to ensure complete defragmentation.
21. Use the NSTL proprietary utility *Fragger* to create empty files that reserve space on the test drive. Use the following syntax:  

```
Fragger -c -n 2500000 -s 4 -b 1000
```
22. Use the NSTL proprietary utility *Fragger* to delete approximately 50% of the empty files on the test drive. Use the following syntax:  

```
Fragger -d -n 2500000 -p 50 -b 1000 -e 1234 -t special
```

23. Change the Page File size: Right click on "My Computer" select "Properties" select "Advanced" then select "Performance Options." From under "Performance Options," select "Change." Make sure the C drive is highlighted, and then insert "0" in the "Initial Size" and the "Maximum Size" input boxes. Select "Set."
24. From the above location, make sure the test drive is highlighted, and then insert "256" in the "Initial Size" and the "Maximum Size" input boxes. Select "Set." (Note: you may receive an error message – ignore this. Windows 2000 does not like the Memory size changed. This error message will not have any bearing on the test results.)
25. Install Microsoft Exchange on the test drive of the server.
26. Setup up ten new users for mail accounts in Exchange. (Use user01, user02, etc.)
27. Install Microsoft Outlook on the test drive of the workstation.
28. Copy the "Test PST" file from the "TestFiles" directory on the C drive of the server to the test drive under the Outlook folder.
29. Open Outlook, and then select "Open a Mailbox." Point to the server for the mailbox. Do not open any mail.
30. Once the mailboxes are set up, you can start testing.
31. Use a stopwatch to time the events. Select the first 50 messages, and then select "Open." Start timing when you select "Open," and end timing when the last message is opened.
32. Create a new folder. Select the inbox and move it to the new folder. (The folder name is not important; we used "Test.") Start timing when the move starts, and end when the pointer is back to the inactive state.
33. Select the Business folder. Start timing when you select this folder; stop timing when you can see the folder contents.
34. Search all messages for the text "word." Start timing after you select "Search," and stop timing when the magnifying glass stops moving in the lower right-hand corner.
35. Search all messages for attachments. Start timing after you select "Search," and stop timing when the magnifying glass stops moving in the lower right-hand corner.
36. Copy the inbox to a "PST" file. Go to "File," then "Export." Select the "Inbox" to export the entire inbox. Start timing when you select "Export," and end timing when the pointer is back to the inactive state.
37. From the client side, select the first 50 messages and then select "Open." Start timing from when you select "Open," and end timing when the last message is opened.
38. From the client side, create a new folder. Select the inbox and move it to the new folder. (The folder name is not important; we used "Test.") Start timing when the move starts and end when the pointer is back to the inactive state.
39. From the client side, select the Business folder. Start timing when you select this folder; stop timing when you can see the folder contents.
40. From the client side, search all messages for the text "word." Start timing after you select "Search," and stop time when the magnifying glass stops moving in the lower right-hand corner.
41. From the client side, search all messages for attachments. Start timing after you select "Search," and stop timing when the magnifying glass stops moving in the lower right-hand corner.
42. Repeat steps 20 through 41 three times.
43. Format the test drive.

## SQL Server 7.0 (Server Side Test)

1. Format the test drive.
2. Install SQL Server 7.0 with all defaults, with the exception of the file locations. All files should be installed on the test drive.
3. Run Diskkeeper to ensure complete defragmentation of the both the C drive and the Test drive. (Save the analysis "report" for the test drive.)
4. Create a new database called "Test." Set the size of the database to 400MB, and the size of the log to 50MB.
5. Import the test database. Use all defaults during the import. Use the SQL's internal timing utility to time this event.
6. Open SQL's Administrative utility.
7. Select the test database, and open it.
8. Open table 1 and select "Authorid," make this the key, and then save. Use SQL's internal timing utility to time this event.
9. Open table 2, change the ID for "Entry," and then save. Use SQL's internal timing utility to time this event.
10. Open table 3 and select "ISBN," make this the key, and then save. Use SQL's internal timing utility to time this event.
11. Open table 4 and select "Authorid" and "ISBN," make this the key, and then save. Use SQL's internal timing utility to time this event.
12. Open table 5 and select "Ordernum," make this the key, and then save. Use SQL's internal timing utility to time this event.
13. Open SQL's Query utility.
14. Use the Test Query 1 from the "Testfiles" directory. The time is recorded in the lower right-hand corner of the Query utility.
15. Use the Test Query 1 from the "Testfiles" directory. The time is recorded in the lower right-hand corner of the Query utility.
16. Repeat steps 1 through 15 three times. Then continue.
17. Format the test drive on the server to ensure complete defragmentation.
18. Use the NSTL proprietary utility *Fragger* to create empty files that reserve space on the test drive. Use the following syntax:  

```
Fragger -c -n 2500000 -s 4 -b 1000
```
19. Use NSTL proprietary utility *Fragger* to delete approximately 50% of the empty files on the test drive. Use the following syntax:  

```
Fragger -d -n 2500000 -p 50 -b 1000 -e 1234 -t special
```
20. Install SQL Server 7.0 with all defaults, with the exception of the file locations. All files should be installed on the test drive.
21. Run Diskkeeper on the C drive, and use the "Analyzer" on the test drive. (Do not run the defragmentation tool on the test drive. Save the "report" for the test drive).
22. Create a new database called "Test." Set the size of the database to 400MB, and the size of the log to 50MB.
23. Import the test database. Use all defaults during the import. Use SQL's internal timing utility to time this event.
24. Open SQL's Administrative utility.
25. Select the test database and open it.
26. Open table 1 and select "Authorid," make this the key, and then save. Use SQL's internal timing utility to time this event.

27. Open table 2, change the ID for "Entry," and then save. Use SQL's internal timing utility to time this event.
28. Open table 3 and select "ISBN," make this the key, and then save. Use SQL's internal timing utility to time this event.
29. Open table 4 and select "Authorid" & "ISBN," make this the key, and then save. Use SQL's internal timing utility to time this event.
30. Open table 5 and select "Ordernum," make this the key, and then save. Use SQL's internal timing utility to time this event.
31. Open SQL's Query utility.
32. Use the Test Query 1 from the "Testfiles" directory. The time is recorded in the lower right-hand corner of the Query utility.
33. Use the Test Query 1 from the "Testfiles" directory. The time is recorded in the lower right-hand corner of the Query utility.
34. Repeat steps 17 through 33 three times. Then continue.
35. Format the test drive on the server to ensure complete defragmentation.
36. Use the NSTL proprietary utility *Fragger* to create empty files that reserve space on the test drive. Use the following syntax:  

```
Fragger -c -n 2500000 -s 4 -b 1000
```
37. Use the NSTL proprietary utility *Fragger* to delete approximately 50% of the empty files on the test drive. Use the following syntax:  

```
Fragger -d -n 2500000 -p 50 -b 1000 -e 1234 -t special
```
38. Change the Page File size: Right click on "My Computer" select "Properties" select "Advanced" then select "Performance Options." From under "Performance Options," select "Change." Make sure the C drive is highlighted, and then insert "0" in the "Initial Size" and the "Maximum Size" input boxes. Select "Set."
39. From the above location, make sure the test drive is highlighted, and then insert "256" in the "Initial Size" and the "Maximum Size" input boxes. Select "Set." (Note: you may receive an error message – ignore this. Windows 2000 does not like the Memory size changed. This error message will not have any bearing on the test results.)
40. Install SQL Server 7.0 with all defaults, with the exception of the file locations. All files should be installed on the test drive.
41. Run Diskkeeper on the C drive, and use the "Analyzer" on the test drive. (Do not run the defragmentation tool on the test drive. Save the "report" for the test drive).
42. Create a new database called "Test." Set the size of the database to 400MB and the size of the log to 50MB.
43. Import the test database. Use all defaults during the import. Use SQL's internal timing utility to time this event.
44. Open SQL's Administrative utility.
45. Select the test database, and open it.
46. Open table 1 and select "Authorid," make this the key, and then save. Use SQL's internal timing utility to time this event.
47. Open table 2, change the ID for "Entry," and then save. Use SQL's internal timing utility to time this event.
48. Open table 3 and select "ISBN," make this the key, and then save. Use SQL's internal timing utility to time this event.
49. Open table 4 and select "Authorid" and "ISBN," make this the key, and then save. Use SQL's internal timing utility to time this event.
50. Open table 5 and select "Ordernum," make this the key, and then save. Use SQL's internal timing utility to time this event.
51. Open SQL's Query utility.
52. Use the Test Query 1 from the "Testfiles" directory. The time is recorded in the lower right-hand corner of the Query utility.
53. Use the Test Query 1 from the "Testfiles" directory. The time is recorded in the lower right-hand corner of the Query utility.

54. Repeat steps 35 through 53 three times.

## **ABOUT THE TESTS**

---

### **Excel 2000 Test**

This test repeatedly opened and saved several Excel files to the fragmented (or not) partition. The four files that it used (file1, file2, file3, and file4) varied in size from 5MB to 20MB. These large files contained many formulas that were also auto-calculated when the spreadsheet opened. File3 opened itself and used data from file1 and file2.

### **SQL Server 7.0 Test**

This test combined two different types of activities on the database: queries and some database maintenance. The database maintenance was probably the least important, as this type of activity is not usually very common. However, it demonstrated the differences of doing these types of activities on fragmented vs. defragmented systems. The query section executed a simple query on the database, which simulated the types of queries typically run by users. The two queries chosen only read from the database, but displayed results from several of the tables.

### **Outlook 2000/Exchange 5.5 Test**

The purpose of this test was to determine the effect of fragmentation on the personal folder database used by Microsoft Outlook. Several tests were run including: opening 50 messages simultaneously; moving messages from the inbox to a separate folder; opening (and displaying to: from: subject: and date:) a large subfolder; a full text search of all messages in a folder for a specific string; and a filter that displayed all messages in a folder that contained an attachment. Each of these tests were executed on the system when the personal folder file was fragmented, and when it was not.

The Microsoft Exchange (v 5.5) test was identical to the Outlook test (indeed Outlook was used as the client). The difference was that all mail existed on the Exchange database (as opposed to locally) and the Exchange database itself was fragmented.

## RESULTS

### Excel 2000 Benchmarks

Configuration #	Run#	Defragmented	Application Fragmented	Application & Swap File Fragmented
Amount of Fragmentation		0%	28%	36%
Configuration 1	Run 1	377042	917900	1178365
	Run 2	391032	897371	1198944
	Run 3	380477	908286	1170513
	Average	<b>382850</b>	<b>907852</b>	<b>1182607</b>
Amount of Fragmentation		0%	33%	20%
Configuration 2	Run 1	433534	762746	1077369
	Run 2	427084	773813	1106882
	Run 3	442657	790276	1093943
	Average	<b>434425</b>	<b>775612</b>	<b>1092731</b>

*Note: All Run times are in milliseconds. Smaller numbers indicate improved performance.*

## Outlook 2000/Exchange 5.5 Benchmarks

<b>Configuration 1</b>				
<b>Mail Sever fragmentation 37%</b>			<i>Time is in milli-seconds</i>	
Event	Run 1	Run 2	Run 3	Average
Open 1st 50 messages	5110	5080	5060	<b>5083</b>
Move inbox mail to new folder	8830	8910	8790	<b>8843</b>
Open Business subfolder (400 messages)	4170	4210	4160	<b>4180</b>
Search all message for word (328 matches)	3775	3699	3741	<b>3738</b>
Search all message for attachments (184 matches)	8040	8100	8170	<b>8103</b>
Copy of inbox items to .pst	637200	649670	637680	<b>641517</b>
<b>Mail Client fragmentation 40%</b>			<i>Time is in milli-seconds</i>	
Event	Run 1	Run 2	Run 3	Average
Open 1st 50 messages	4430	4620	4570	<b>4540</b>
Move inbox mail to new folder	4440	4430	4510	<b>4460</b>
Open Business subfolder (400 messages)	6750	6740	6700	<b>6730</b>
Search all message for word (328 matches)	83790	85010	84240	<b>84347</b>
Search all message for attachments (184 matches)	3230	3270	3200	<b>3233</b>
<b>Mail Sever Defragmented</b>			<i>Time is in milli-seconds</i>	
Event	Run 1	Run 2	Run 3	Average
Open 1st 50 messages	4030	4100	4230	<b>4120</b>
Move inbox mail to new folder	5630	5690	5530	<b>5617</b>
Open Business subfolder (400 messages)	2710	2690	2780	<b>2727</b>
Search all message for word (328 matches)	1755	1659	1750	<b>1721</b>
Search all message for attachments (184 matches)	4930	5260	4910	<b>5033</b>
Copy of inbox items to .pst	359970	351010	356280	<b>355753</b>
<b>Mail Client Defragmented</b>			<i>Time is in milli-seconds</i>	
Event	Run 1	Run 2	Run 3	Average
Open 1st 50 messages	4410	4500	4470	<b>4460</b>
Move inbox mail to new folder	2990	2870	2860	<b>2907</b>
Open Business subfolder (400 messages)	3680	4090	3610	<b>3793</b>
Search all message for word (328 matches)	4613	4292	4321	<b>4409</b>
Search all message for attachments (184 matches)	2500	2770	2720	<b>2663</b>

Note: All Run times are in milliseconds. Smaller numbers indicate improved performance.



## SQL Server 7.0 Benchmarks

### Configuration 1

<i>Database Fragmentation 39%</i>					
<i>Bulk Insert and Table creation</i>					
<i>Run</i>	<i>Time</i>				
Run 1:	331660				
Run 2:	333910				
Run 3:	330840				
Average	332137				
<i>Key Creation</i>					
<i>Run</i>	<i>Table 1</i>	<i>Table 2</i>	<i>Table 3</i>	<i>Table 4</i>	<i>Table 5</i>
Run 1:	3770	8890	239490	6750	169560
Run 2:	3780	8870	234910	6490	167010
Run 3:	3710	9190	236010	6610	169320
Average	3753	8983	236803	6617	168630
<i>Queries</i>					
<i>Query 1</i>				<i>Query 2</i>	
Run 1:	106000		Run 1:	65000	
Run 2:	109000		Run 2:	67000	
Run 3:	107000		Run 3:	64000	
Average	107333		Average	65333	

Note: All Run times are in milliseconds. Smaller numbers indicate improved performance.

**Configuration1 (Continued)**

<b>Database &amp; Page File Fragmentation 38%</b>					
<b>Bulk Insert and Table creation</b>					
<b>Run</b>	<b>Time</b>				
Run 1:	342130				
Run 2:	348010				
Run 3:	348960				
<b>Average</b>	<b>346367</b>				
<b>Key Creation</b>					
<b>Run</b>	<b>Table 1</b>	<b>Table 2</b>	<b>Table 3</b>	<b>Table 4</b>	<b>Table 5</b>
Run 1:	3850	8910	263480	8910	178170
Run 2:	3870	8900	269010	8920	179010
Run 3:	3810	8950	267960	8910	178960
<b>Average</b>	<b>3843</b>	<b>8920</b>	<b>266817</b>	<b>8913</b>	<b>178713</b>
<b>Queries</b>					
<b>Query 1</b>			<b>Query 2</b>		
Run 1:	157000		Run 1:	66000	
Run 2:	160000		Run 2:	69000	
Run 3:	159000		Run 3:	69000	
<b>Average</b>	<b>158667</b>		<b>Average</b>	<b>68000</b>	

Note: All Run times are in milliseconds. Smaller numbers indicate improved performance.

**Configuration1 (Continued)**

<i>Defragmented</i>					
<i>Bulk Insert and Table creation</i>					
<i>Run</i>	<i>Time</i>				
Run 1:	320690				
Run 2:	321340				
Run 3:	317690				
<b>Average</b>	<b>319907</b>				
<i>Key Creation</i>					
<i>Run</i>	<i>Table 1</i>	<i>Table 2</i>	<i>Table 3</i>	<i>Table 4</i>	<i>Table 5</i>
Run 1:	3590	7170	191160	5940	64510
Run 2:	3670	7270	191270	5870	66020
Run 3:	3630	7310	190990	5670	63770
<b>Average</b>	<b>3630</b>	<b>7250</b>	<b>191140</b>	<b>5827</b>	<b>64767</b>
<i>Queries</i>					
<i>Query 1</i>			<i>Query 2</i>		
Run 1:	103000		Run 1:	64000	
Run 2:	100000		Run 2:	64000	
Run 3:	102000		Run 3:	65000	
<b>Average</b>	<b>101667</b>		<b>Average</b>	<b>64333</b>	

Note: All Run times are in milliseconds. Smaller numbers indicate improved performance.



**Configuration2 (Continued)**

<b>Database &amp; Page File Fragmentation 39%</b>					
<b>Bulk Insert and Table creation</b>					
<b>Run</b>	<b>Time</b>				
Run 1:	250470				
Run 2:	252490				
Run 3:	252960				
<b>Average</b>	<b>251973</b>				
<b>Key Creation</b>					
<b>Run</b>	<b>Table 1</b>	<b>Table 2</b>	<b>Table 3</b>	<b>Table 4</b>	<b>Table 5</b>
Run 1:	2170	4910	201950	4140	100420
Run 2:	2190	4900	202010	4190	101010
Run 3:	2200	4790	203960	4170	100090
<b>Average</b>	<b>2187</b>	<b>4867</b>	<b>202640</b>	<b>4167</b>	<b>100507</b>
<b>Queries</b>					
<b>Query 1</b>			<b>Query 2</b>		
Run 1:	71000		Run 1:	66000	
Run 2:	70000		Run 2:	66000	
Run 3:	73000		Run 3:	67000	
<b>Average</b>	<b>71333</b>		<b>Average</b>	<b>66333</b>	

Note: All Run times are in milliseconds. Smaller numbers indicate improved performance.



---

## PERFORMANCE TEST CONCLUSIONS

---

### Excel 2000

Results show that when Configuration #1 was defragmented it ran the Excel Benchmark 208.9% faster than a system that has Excel and the Page File fragmented – a little more than 2 times faster.

Results show that when Configuration #2 was defragmented it ran the Excel Benchmark 151.5% faster than a system that has Excel and the Page File fragmented - a little more than 1 1/2 times faster.

### SQL Server 7.0

#### System Configuration #1

**Table Creation A:** Results show that when Configuration #1 was defragmented it ran the Table Creation Benchmark 8.3% faster than a system that has the Database and the Page File fragmented.

**Table Creation B:** A defragmented system ran the Table Creation Benchmark 3.8% faster than a system where only the Database was fragmented.

**Key Creation 1-5 A:** Results show that when Configuration #1 was defragmented the Key Creation 1 Benchmark ran 5.9% faster, Key Creation 2 Benchmark ran 23% faster, Key Creation 3 Benchmark ran 39.6% faster, Key Creation 4 Benchmark ran 53% faster, and Key Creation 5 Benchmark ran 175.9% faster than a system that had the Database and the Page File fragmented.

**Key Creation 1-5 B:** A defragmented system ran Key Creation 1 Benchmark 3.4% faster, Key Creation 2 Benchmark ran 23.9% faster, Key Creation 3 Benchmark ran 23.9% faster, Key Creation 4 Benchmark ran 13.6% faster, and Key Creation 5 Benchmark ran 160.4% faster a system where only the Database was fragmented.

**Query 1 & 2 A:** Results show that when Configuration #1 was defragmented the Query 1 Benchmark ran 56.1% faster, and Query 2 Benchmark ran 5.7% faster than a system that had the Database and the Page File fragmented.

**Query 1 & 2 B:** A defragmented system ran the Query 1 Benchmark 5.6% faster, and Query 2 Benchmark ran 1.6% faster than a system where only the Database was fragmented.

#### System Configuration #2

**Table Creation A:** Results show that when Configuration #2 was defragmented the Table Creation Benchmark ran 18.2% faster than a system that had the Database and the Page File fragmented.

**Table Creation B:** A defragmented system ran the Table Creation Benchmark 15.8% faster than a system where only the Database was fragmented.

**Key Creation 1-5 A:** Results show that when Configuration #2 was defragmented the Key Creation 1 Benchmark ran 16.1% faster, Key Creation 2 Benchmark ran 69.2% faster, Key Creation 3 Benchmark ran 15.7% faster, Key Creation 4 Benchmark ran 11.9% faster, and Key Creation 5 Benchmark ran 108% faster than a system that had the Database and the Page File fragmented.

**Key Creation 1-5 B:** A defragmented system ran Key Creation 1 Benchmark ran 2.3% faster, Key Creation 2 Benchmark ran 22.5% faster, Key Creation 3 Benchmark ran 6.1% faster, Key Creation 4 Benchmark ran 10.3% faster, and Key Creation 5 Benchmark ran 42.3% faster a system where only the Database was fragmented.

**Query 1 & 2 A:** Results show that when Configuration #2 was defragmented the Query 1 Benchmark ran 1122.9% faster, and Query 2 Benchmark ran 3.6% faster than a system that had the Database and the Page File fragmented.

**Query 1 & 2 B:** A defragmented system ran the Query 1 Benchmark 41208% faster, and Query 2 Benchmark ran 13% faster than a system where only the Database was fragmented.

## **Outlook 2000/Exchange 5.5**

### **Workstation Configuration #1**

The workstation in Configuration #1 that was defragmented opened the first 50 messages 2% faster than a fragmented system.

A defragmented system moved the inbox to a new folder 53% faster than a fragmented system.

A defragmented system opened the Business folder 77% faster than a fragmented system.

A defragmented system searched all messages for a particular word 1813% faster than a fragmented system.

A defragmented system searched all messages for attachments 21% faster than a fragmented system.

### **Workstation Configuration #2**

The workstation in Configuration #2 that was defragmented opened the first 50 messages 16% faster than a fragmented system.

A defragmented system moved the inbox to a new folder 104% faster than a fragmented system.

A defragmented system opened the Business folder 961% faster than a fragmented system.

A defragmented system searched all messages for a particular word 11% faster than a fragmented system.

A defragmented system searched all messages for attachments 41% faster than a fragmented system.

### **Server Configuration #1**

The server in Configuration #1 that was defragmented opened the first 50 messages 23% faster than a fragmented system.

A defragmented system moved the inbox to a new folder 57% faster than a fragmented system.

A defragmented system opened the Business folder 53% faster than a fragmented system.

A defragmented system searched all messages for a particular word 117% faster than a fragmented system.

A defragmented system searched all messages for attachments 61% faster than a fragmented system.

A defragmented system copied the inbox to a PST file 80% faster than a fragmented system.

### **Server Configuration #2**

The server in Configuration #2 that was defragmented opened the first 50 messages 4% faster than a fragmented system.

A defragmented system moved the inbox to a new folder 39% faster than a fragmented system.

A defragmented system opened the Business folder 40% faster than a fragmented system.

A defragmented system searched all messages for a particular word 16% faster than a fragmented system.

A defragmented system searched all messages for attachments 62% faster than a fragmented system.

A defragmented system copied the inbox to PST file 21% faster than a fragmented system.

**SYSTEM COMPONENTS**

Configuration #1	
System	Components
<b>Workstation</b>	Compaq Deskpro EP
	Pentium II 400MHz processor
	128MB SDRAM
	6.4GB hard drive
	ATI 3D PCI Graphics Adapter with 4MB
	ATI Technologies Inc.
	Drivers: ati.sys, ati.dll
	Version: 5.1.126, 4.0.0
	Microsoft Windows 2000 Professional
<b>Server</b>	Compaq Proliant 2500
	Pentium Pro 200MHz Dual processor
	128MB SDRAM
	5 - 4GB Ultra Wide SCSI hard drive
	Cirrus Compatible PCI Display Adapter
	Cirrus Inc.
	Drivers: cirrus.sys, vga.dll, cirrus.dll, vga256.dll
	Microsoft Windows 2000 Server

Configuration #2	
System	Components
<b>Workstation</b>	Compaq Deskpro
	Pentium II 266MHz processor
	96MB SDRAM
	4GB hard drive
	MGA PCI Adapter with 2MB
	Maxtrox Graphics Inc.
	Drivers: mga_mil.sys, mga.dll
	Version: 2.20.4.0.0
	Microsoft Windows 2000 Professional
<b>Server</b>	Compaq Proliant 2500
	Pentium Pro 200MHz Dual processor
	128MB SDRAM
	5 - 4GB Ultra Wide SCSI hard drive
	Cirrus Compatible PCI Display Adapter
	Cirrus Inc.
	Drivers: cirrus.sys, vga.dll, cirrus.dll, vga256.dll
	Microsoft Windows 2000 Server