TANDEMCOMPUTERS

A Benchmark of NonStop SQL on the Debit Credit Transaction

The Tandem Performance Group

Technical Report 88.2
March 1987
Part Number 12523

A Benchmark of NonStop SQL on the Debit Credit Transaction

The Tandem Performance Group

Technical Report 88.2
March 1987
Part Number 12523

A Benchmark of NonStop SQL on the Debit Credit Transaction

The Tandem Performance Group
March 1987

Abstract: NonStop SQL is an implementation of ANSI SQL on Tandem Computer Systems. Debit Credit is a widely-used industry-standard transaction. This paper summarizes a benchmark of NonStop SQL which demonstrated linear growth from 2 to 32 processors, and 14 to 208 Debit Credit transactions per second. The benchmark also compared the performance of NonStop SQL to the performance of a record-at-a-time file system interface.

TABLE OF CONTENTS

Tandem Technical Report 88.2, Part Number 12523

INTRODUCTION

NonStop SQL is an implementation of ANSI SQL [ANSI]. In contrast to other SQL implementations, NonStop SQL is designed for on-line transaction processing as well as for information-center use. It delivers high performance and good price performance. In addition, it supports distributed data with local autonomy.

Tandem's Performance Assurance Group stress tested NonStop SQL and compared its performance to that of the Tandem file system. Parts of the benchmark were audited and certified by an independent auditor. The benchmark workbook is over one thousand pages long [Workbook]. This paper summarizes the benchmark results.

The benchmark demonstrated the following aspects of NonStop SQL:

- It is functional and has been stress tested for high-volume transaction processing applications.
- It allows distributed data and distributed transactions.
- It runs on small departmental systems as well as on large mainframes.
- It provides linear increases in throughput when discs, processors, and communications lines are added using the Tandem's LAN (Dynabus and FOX).
- There is no apparent limit to the transaction throughput of NonStop SQL systems. In particular, there are no bottlenecks in systems running hundreds of transactions per second.
- NonStop SQL performs the Debit Credit transaction more quickly than the record-at-a-time file system (Enscribe).
- Both NonStop SQL and Enscribe have impressive price/performance at both low and high transaction volumes.
- There is no economy of scale in transaction processing -- the price performance curve is essentially flat from a departmental system (NonStop EXT10) a large mainframe (NonStop VLX).

THE DEFINITION OF THROUGHPUT AND PRICE/PERFORMANCE

The article "A Measure of Transaction Processing Power" [Anon] defines a standard database, a standard terminal network and a way to scale them to larger systems. It also defines a standard transaction, called the Debit Credit transaction, and specifies how to measure the throughput and price/performance of the resulting system.

Briefly, the database consists of four SQL tables:

- ACCOUNT: a table of bank accounts. Each record is 100 bytes long and holds the account number and balance.
- TELLER: a table describing bank tellers. Each record is 100 bytes long and holds the teller number and cash position.
- BRANCH: a table describing the cash positions of each bank branch. Each record is 100 bytes long and holds the branch number and cash position of all tellers at that branch.
- HISTORY: an entry-sequence table containing records of all transactions. Each record is 50 bytes long and holds the account, teller, branch, delta, and timestamp of the transaction.

The transaction coded in SQL is:

```
READ 100 BYTES FROM TERMINAL;
PERFORM PRESENTATION SERVICES GIVING
                       account, teller, branch, delta;
EXEC SQL  BEGIN WORK;

EXEC SQL  UPDATE ACCOUNT
          SET balance = balance + :delta
          WHERE account_number = :account;
EXEC SQL  UPDATE TELLER
          SET balance = balance + :delta
          WHERE teller_number = :teller;
EXEC SQL  UPDATE BRANCH
          SET balance = balance + :delta
          WHERE branch_number = :branch;
EXEC SQL  INSERT INTO HISTORY VALUES
          (:timestamp,:account,:teller,:branch,:delta);

EXEC SQL  COMMIT WORK;

PERFORM PRESENTATION SERVICES;
WRITE 200 BYTES TO TERMINAL;
```

The system is presented with various average transaction rates, and the response time is measured over 10-minute time windows creating a response time curve. (See Figure 1.)

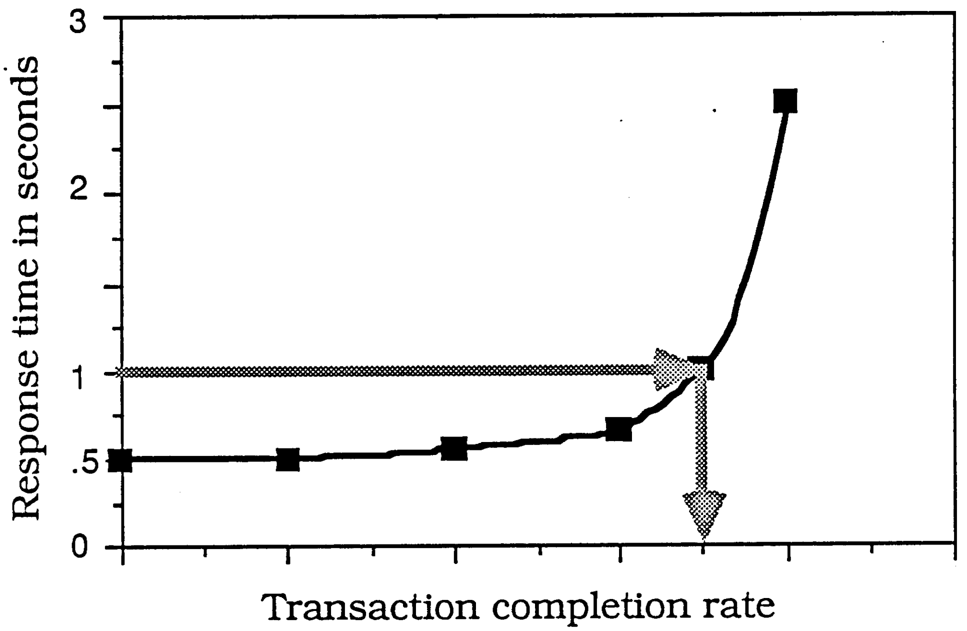

Figure 1: A typical response-time curve showing how response time increases as the transaction load increases. Each data point represents the 95th percentile response time of the system at that transaction rate. The throughput is then defined as the transaction rate at which the 95% response time exceeds 1 second. This throughput computation is demonstrated by the shaded arrows.

A system that can run one such transaction per second, giving less than one-second response time to 95% of the transactions, is defined to be a one transaction per second (tps) system. The database of a 1-tps system is defined as:

100,000	Accounts
100	Tellers
10	Branches
2,590,000	History Records

The History table is sized to accommodate 90 days of history records, assuming the average throughput is one-third of the peak throughput.

The standard also specifies that a 1-tps system has 100 tellers at 10 branches. Each teller thinks for an average of 100 seconds and then submits a transaction. The tellers use block mode terminals (like IBM 3270 terminals) which are configured to have 10 input and output fields. The input message is 100 bytes and the output message is 200 bytes. Messages are transmitted via the X.25 protocol.

Systems that run more than 1 tps have the database and network scaled linearly. For example, a 100 tps system has a database and network 100 times larger.

The standard specifies that accounts belong to branches, and tellers belong to branches. This give some locality and clustering to transactions. If the database is distributed, then 15% of the transactions arrive at branches other than the account's home branch. These 15% are uniformly distributed among the other branches.

The benchmark requires that the transactions be run with undo and redo transaction protection (abort, auto-restart, and rollforward recovery). In addition, it specifies that the transaction log must be duplexed.

DEPARTURES FROM THE DEBIT CREDIT STANDARD

The NonStop SQL benchmark departed from the Datamation standard in the following ways.

1. Terminals were driven in record mode (Intelligent Device Support) rather than in block mode. In effect, this assumes that presentation services are done in the terminal.

2. It was assumed that each branch had a concentrator that multiplexed the teller terminals at the branch. This has the effect of reducing the number of sessions by a factor of ten for the same transaction rate, when compared to the standard.

3. The measure of tps was the throughput when 90% of the transactions got 2-second or less response time. The standard specifies 95% of the transactions getting 1-second response time.

4. The application verified that debits did not cause negative account balances.

5. The system was measured over 10-minute periods, and all transactions for each period were used to compute the response-time curves and consequent tps ratings.

6. Response times were measured with the driver system. The standard specifies that the response time can be measured at the interface to the service system.

7. All devices were driven by NonStop process pairs [Bartlett] so that no single failure would cause a denial of service.

8. All discs were mirrored (duplexed), as is typical of Tandem systems.

9. Standard products were used. All applications were coded in Cobol85.

Items 1, 2, and 3 tend to create an optimistic tps rating. Items 4 through 9 tend to reduce the system's tps rating.

THE BENCHMARK SYSTEM DESIGN

The benchmark hardware consisted of 32 VLX processors. Each processor had a 5-megabyte-per-second channel, 8Mb of main memory, and is rated at about 3 Tandem MIPS.

In addition, an NonStop EXT10 system was included in the benchmark hardware to demonstrate scaleability. The EXT10 is Tandem's smallest available NonStop system. It consists of two processors, each with 4 Mbytes of main memory and 4 discs. The EXT10 processors together are approximately half as powerful as a single VLX processor. Thus th complex was sized as a 32.5 VLX system. See.Figure 2.)

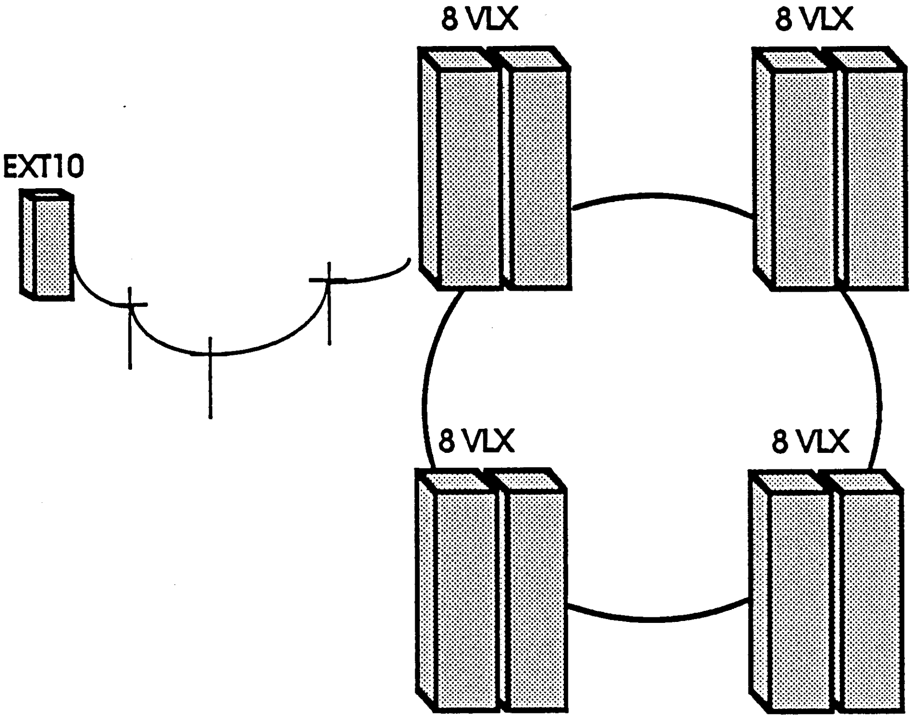

Figure 2. The benchmark configuration. The two processor EXT10 is connected to one VLX node via a 9.6 Kbit line. The 30 VLX processors are divided into four nodes of 8 processors each The nodes are connected via a high speed (32 Mbits/second) FOX ring. The whole complex looks like a single system to an application programmer. A separate driver system submits transactions to this system via 56-Kbit X.25 lines.

The VLX processors were divided into four nodes of eight VLX CPUs each. Each node connected its eight processors via dual 160-Mbit-per-second inter-processor busses, and the nodes were interconnected via FOX, a dual bi-directional fiber-optic ring.

To model a remote departmental node, the EXT10 was connected to a VLX via a 9.6 Kbit-per-second communication line.

The operating system made the entire 32VLX plus EXT10 configuration appear to be a single system with location transparency.

The database and transaction load were partitioned into 33 parts. Based on preliminary tests, each VLX processor would process up to 8tps, and the EXT10 could process up to 4tps. All sizings were based on these preliminary measurements.

Each transaction input message was 100 bytes long; the reply was 200 bytes long. With X.25 overheads, this translates to 340 bytes in all. At 8tps, this is 22 Kbits per second. Therefore each VLX processor was configured with a 56 Kbit line, and the EXT10 was configured with a single 56-Kbit line for its terminals.

The database for each eight processor VLX was sized as follows:

- A mirrored disc to store the programs and transaction log.
- A mirrored disc dedicated to the history table. In the benchmark only one volume was configured, but in pricing the system, the 90-day history file of each node was sized at 6.7 gigabytes (14 mirrored volumes).

According to the standard, 8tps implies:

80	branches	8	Kbytes
800	tellers	80	Kbytes
800,000	accounts	80	Mbytes

These records, along with their indexes and some slack, fit comfortably on Tandem's smallest disc. Therefore, each VLX processor was given a mirrored disc volume to hold its part of the Account, Branch, and Teller (ABT) tables. A 1.6-megabyte disc buffer pool (main memory disc cache) per disc volume was sufficient to keep all branches, tellers and the account index pages resident in main memory.

Aggregating all this over the 32VLX system gives the size of the benchmark system

NETWORK		DATABASE		
		RECORDS		SIZE
25,600	tellers	25,600	tellers	2.6 Mbytes
2,560	branches	2560	branches	0.3 Mbytes
		25,600,000	accounts	2.6 Gbytes
		54,000,000,000	history	27.0Gbytes

Figure 3 shows the disc and communications line layout of each of the four VLX nodes.

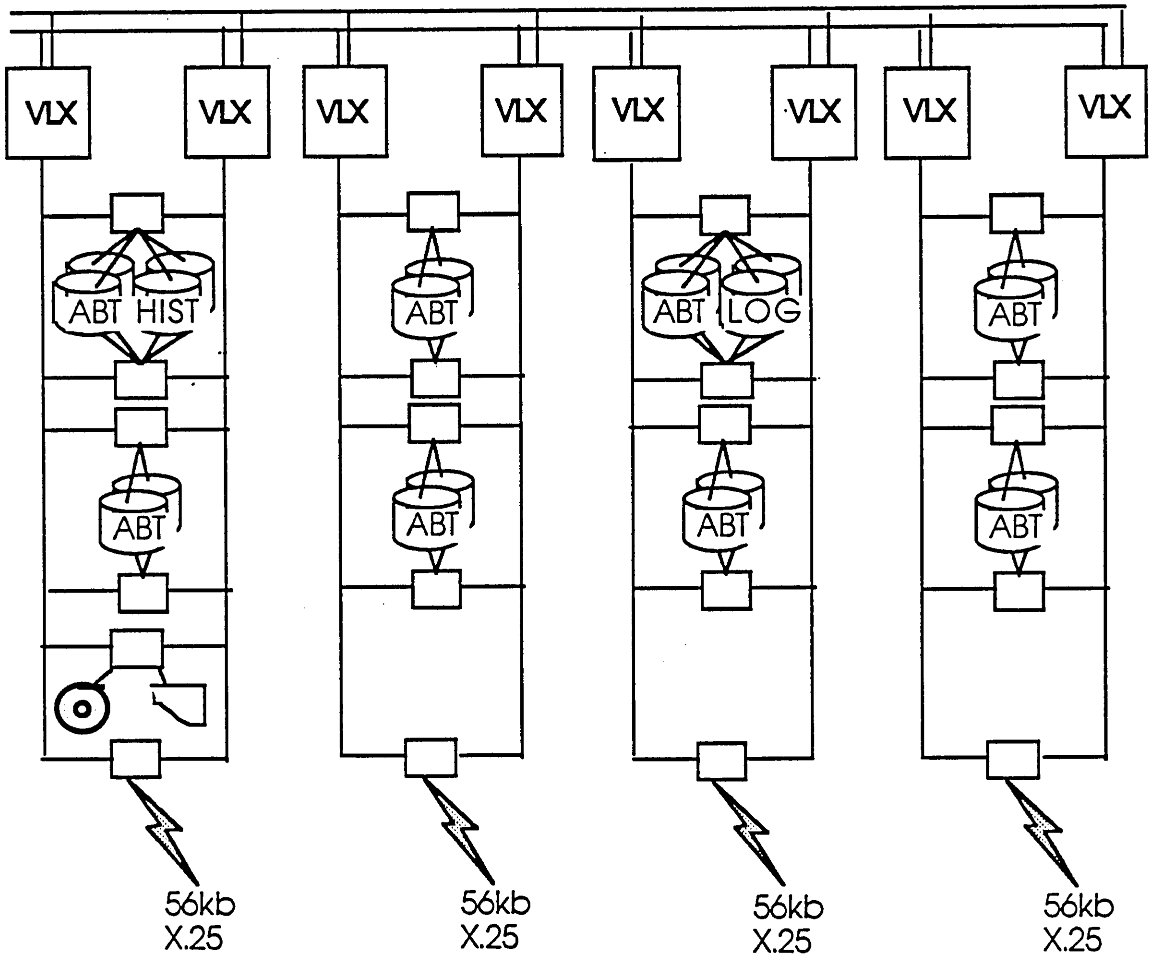

Figure 3. The configuration of each VLX node in the ring. Each processor has a 56 Kbit-per-second X.25 communications line. In addition each processor has a partition of the Account, Branch, and Teller tables (ABT). One of the processors also supports the duplexed log and another supports the history files. As illustrated, all discs are duplexed for fault tolerance.

The SQL tables were defined to be partitioned by the appropriate key values. The approximate syntax for creating the account table with 33 partitions is:

```
EXEC SQL  CREATE TABLE ACCOUNT ( number        DECIMAL(12),
                                 balance       DECIMAL(9,2),
                                 ...
                                 PRIMARY KEY(number)
                                 )
                    PARTITION \a.$vlx2  FIRST KEY   800000,
                    PARTITION \a.$vlx3  FIRST KEY  1600000,
                    ...
                    PARTITION \c.$vlx32 FIRST KEY 24800000,
                    PARTITION \ext.$ext FIRST KEY 25600000;
```

The branch and teller tables are similarly partitioned. NonStop SQL hides this partitioning from the application programmer. Programs access partitioned tables as though they were all on one local disc.

NonStop SQL imposes a limit of a few hundred partitions. The record-at-a-time file system (Enscribe) imposed a limit of 16 partitions. Consequently, the Enscribe database was limited to a 16 VLX processor configuration.

Terminal control and presentation services were handled by Pathway [Pathway] -- Tandem's analog of IBM's CICS or IMS/DC. Since each cpu was sized at 8tps, each cpu supported 800 teller records and 80 branches. Thirty two branches (320 tellers) were configured per Pathway terminal control program (TCP). This resulted in 2.5 TCP programs per VLX CPU.

To avoid queueing on database servers at 8tps with a 2 second response time, 20 Debit Credit servers were configured per CPU (20 ~ 2×8). The code and data for the TCPs were approximately .5Mb per processor. The Enscribe servers consumed .25Mb per processor. The NonStop SQL servers used 10Kb more memory -- altogether 20 NonStop SQL servers used about 0.5Mb per processor.

The EXT10 system was sized at half of a VLX processor. The programs and log were on one mirrored disc and the database resided on a second mirrored disc.

A second complex of 12 NonStop TXP processors simulated the network of 2,560 branches and 25,600 tellers by submitting transactions via X.25 lines. Typically, transactions were submitted on each line at average rates between 4tps and 8tps, with exponentially distributed interarrival times. The EXT10 was driven at about 4TPS. The driver system measured response time as the time between the start of its sending the input message and the completion of the receipt of the transaction response.

Each Transaction message named an account, branch, teller, and debit amount. As specified by the standard, in 85% of the cases, the account and branch were local; in 15% of the cases the account was in a branch other than the local branch of the teller. Some of these "nonlocal" transaction were in branches at the same node, but most went to other nodes of the network. For example, when the system was running over 200tps, the EXT10 might originate or receive one distributed tps, while each VLX node might originate or receive 15 distributed transactions per second.

The total hardware configuration used in the benchmark was:

DRIVER SYSTEM		BENCHMARKED SYSTEM	
processors	lines	processors	storage
12 TXP	33 56Kbit X.25 lines	32VLX + EXT10	86 Discs
24 MIPS		100 MIPS	20 Gbytes
		264Mbytes	

Assembling this hardware was difficult because the VLX had a backlog of orders. The equipment was only available for 50 days. In that time the system had to be assembled, benchmarked, and then disassembled. Fortunately, the equipment was installed on schedule. There were no hardware faults during the benchmark, and no critical software errors were discovered in the SQL product. The only problem was a double power failure at the facility early in the benchmark.

THE EXPERIMENTS

The benchmark measured the ability to grow a NonStop VLX system from an eight processor VLX cluster to a 32 processor VLX cluster using Tandem's fiber optic ring (FOX). In addition, to demonstrate that NonStop SQL can run on a small departmental system , an EXT10 was given a proportionate part of the database and attached to the VLX cluster via a 9.6Kbit line. It processed local transactions and received and sent 15% distributed transactions.

First the throughput of a single 8VLX node was measured for both Enscribe (the record-at-a-time file system), and NonStop SQL. Then a pair of 8VLX nodes were connected via FOX and their throughput measured. Then the configuration was doubled to four nodes of 8VLX processors each and the EXT10 was added. Because Enscribe is limited to 16 partitions, this last experiment was only performed for NonStop SQL. The final configuration is shown in Figure 2.

Figure 4 shows the response time curves for Enscribe and SQL on the the 8VLX node. Notice that SQL has slightly shorter response times. This is because SQL uses fewer instructions to perform the Debit Credit transaction. The reasons for this will be explained later.

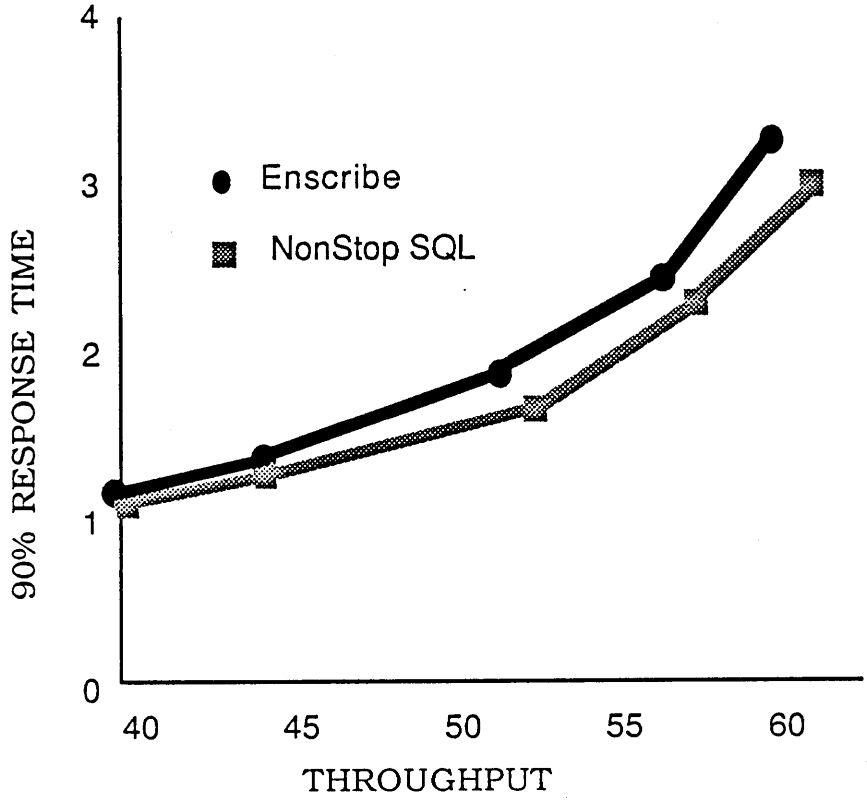

Figure 4. The 90% response time vs throughput of a 8VLX node for the Enscribe file system and for NonStop SQL.

At equilibrium, each Debit Credit transaction generated one physical read and two physical writes of the account file (one read and one mirrored write.) The Branch, Teller, and History files are Hot Spots [Gawlick] and so the associated IO for them is amortized over many transactions. In addition, each transaction contributes .4 IOs to the transaction log (audit trail), because Group Commit [Gawlick] typically batches 5 transactions per log flush. A detailed breakdown of CPU utilization by function is shown in Figure 5.

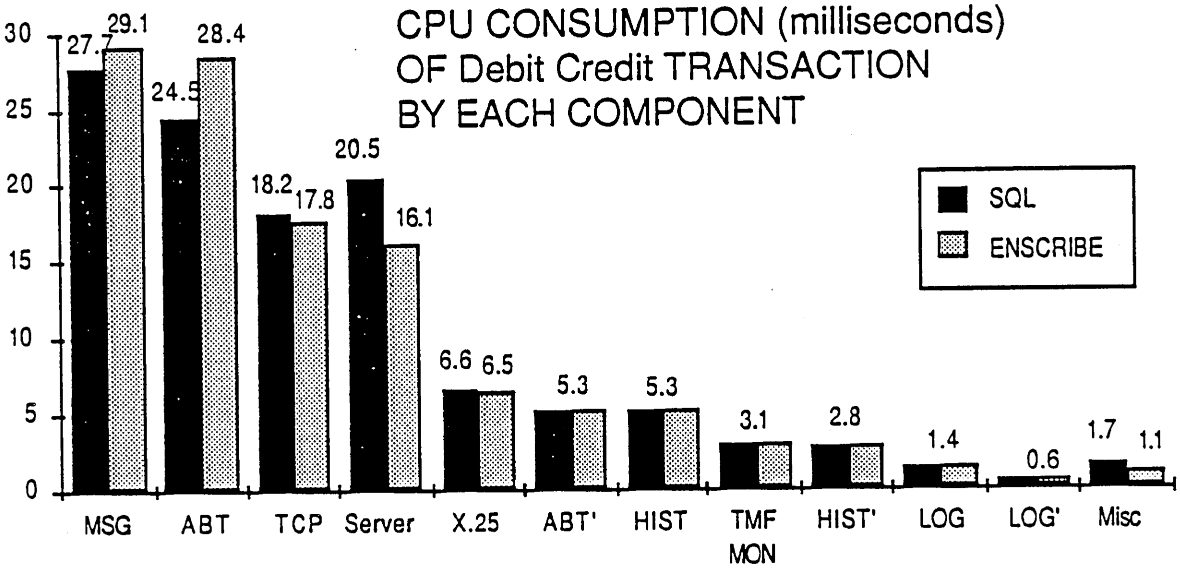

Figure 5. An analysis of the CPU time consumed per component processing the Debit Credit transaction by SQL and by the Enscribe file system. MSG referrs to message system time, ABT refers to handling of the account, branch, and teller databases. Server is the application program. TMF-MON is the transaction group commit coordinator. ABT', HIST', and LOG' are the backup disc processes of the corresponding primary disc processes.

These experiments were repeated for a 16VLX cluster and then for a 32VLX cluster. The measured response times are displayed in Figure 6. The resulting throughput curves are shown in Figure 7.

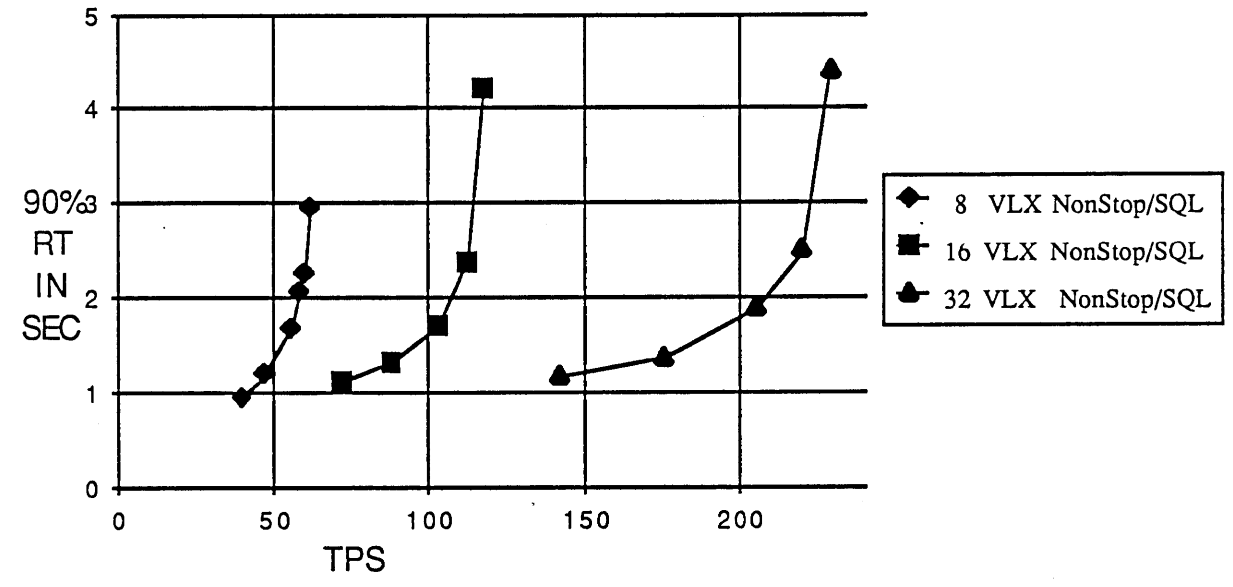

Figure 6. The response time curves for NonStop SQL on the three different configurations.

Figure 7 shows that throughput increased at a linear rate of about .9x. Based on early measurements, the system was expected to perform about 8tps per CPU and have liner growth from 8 to 32 CPUs, i.e., the 32 processor system would handle approximately 256tps. In fact the 8VLX system did 7.2tps per VLX CPU, and there was a 10% dropoff as the the system was scaled to 30 processors. This dropoff is due to the increased cost of distributed transactions. In the one node case, no transactions are distributed to other nodes. In the two node case, 7.5% of the transactions are processed by two nodes (50% of 15%). In the four node case, 11% of the transactions are processed by two nodes (75% of 15%). When transactions do work at multiple nodes they cost extra CPU instructions and extra messages. This affects their response time and reduces overall throughput. In spite of this, the throughput curve is approaching the linear asymptote of .88x.

Because NonStop SQL uses fewer instructions than Enscribe in this benchmark, SQL has slightly higher transaction throughput. Ignoring response time, peak NonStop SQL throughput was 229tps.

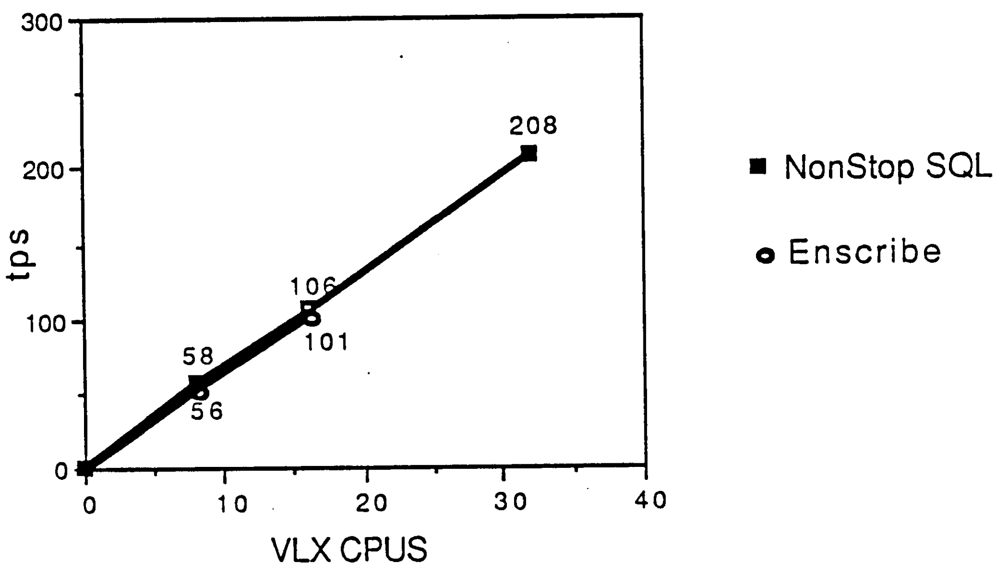

Figure 7. The tps rating of Enscribe and NonStop SQL running the Debit Credit transaction on clusters of 0, 8, 16, and 32 VLX processors.

These experiments were audited by the Codd and Date Consulting Group, which verified that:

- The transaction was correctly implemented.
- The database was properly sized.
- 15% of the transactions were inter-branch.
- Transactions were protected by a dual undo/redo log.
- The response time was measured correctly.
- The measured response time matched the observed response time.
- NonStop SQL and Enscribe have comparable performance.

Given this linear throughput vs hardware, it is possible to quote the price/performance of a system in terms of the price/performance of a single processor and then extrapolate. The Debit Credit standard measures price as the 5-year hardware and software expense of the system. For Tandem systems, the five year cost per transaction is approximately the same for a small system as for a large one. The dominant issue is which processor line the user selects. Generally, newer systems are less expensive. The following table shows the March 1987 price performance for the various Tandem processors. There are two price columns, the first shows the cost per transaction if all discs are mirrored, the second shows the cost if only the audit disc is mirrored. Since these computations were done, Tandem has introduced a new low-end processor which has 20% better price performance and it has lowered prices on "old" equipment by about 10%.

SYSTEM	TPS	K$/TPS	
	90% at 2 sec RT	FAULT TOLERANT	NON-FT
8VLX	58	56K$	52K$
16VLX	106	59K$	54K$
32VLX	208	60K$	55k$
EXT10	4	64K$	60K$

WHY IS NonStop SQL FASTER THAN ENSCRIBE?

Historically, SQL has been confined to information center environments and to low-end systems where programmer productivity is more important than system performance. In these environments, the power of the SQL language and the relational model compensated for the lackluster performance of relational systems.

NonStop SQL is the first SQL system designed to run many short transactions at high degrees of multi-programming. It includes modern techniques for transaction concurrency control and transaction logging. As a consequence, it has no obvious bottlenecks. In addition, it is structured as a distributed system and so can be used at one node or on a multi-node network.

There are many reasons for the success of the benchmark. First, NonStop SQL benefited from the experiences of its predecesssors. The developers had collectively worked on System R, SQL/DS, DB2, R*, IDM, Encompass, Esvel, Wang VS and several systems.

A second advantage was the close cooperation between the Database Group and the Operating Systems and Languages groups. Unlike most other vendors, the whole Tandem system is geared to distributed transaction processing. As a consequence, development groups are much more cooperative than the corresponding groups of companies which offer "general-purpose" systems. In particular, SQL was able to add SQL programs to the standard object file format and was able to use the operating system's transaction-protected remote procedure call mechanisms.

The most significant advantage of SQL over the record-at-a-time Enscribe interface was SQL's ability to perform set-oriented operations in a single message. SQL can preform a test and update in a single message to a disc server while Enscribe must fetch the record to the application and then deliver the changed record back to the database. Figure 8 illustrates this point in detail.

In general, NonStop SQL subcontracts single-variable queries to remote servers. This allows the disc servers to act as database machines that perform multiple selects, projects, updates and deletes with a single message which returns only the completion code and the desired subset of the database.

In every benchmark performed so far, the message savings of NonStop SQL have compensated for the extra work the system must perform in order to implement the more complex semantics of SQL. For example, on the Wisconsin benchmark [Bitton], NonStop SQL gives a median speedup of two over Enform, Tandem's report writer for Enscribe files. Some of the improvement is due to better algorithms, but most is due to the reduced message traffic.

The record-at-a-time disadvantage of Enscribe is typical of non-relational systems such as DL/1 and DBTG. It suggests that relational systems will dominate the distributed database arena.

```
COBOL and  ENSCRIBE              SQL
START branch                     UPDATE branch
 KEY IS EQUAL TO branch_number.  SET balance = balance + :delta
READ branch RECORD WITH LOCK     WHERE number = :branch_number
ADD delta TO balance OF branch   AND balance  + :delta >= 0;
REWRITE branch
```

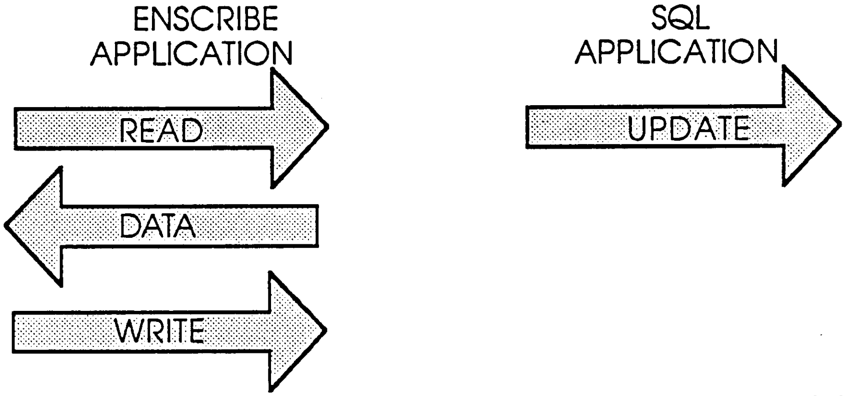

Figure 8. A comparison of a Cobol record-at-a-time update which sends a read followed by a write, and a SQL set-oriented update which sends a single update message. This message savings helps explain why Tandem's SQL is faster than its record-at-a-time Enscribe file manager.

CONCLUSION

Traditionally, relations systems have had a reputation for poor performance. This benchmark demonstrates that there is nothing inherently wrong with the performance of relational systems. In fact SQL has certain performance advantages over the record-at-a-time interfaces of DL/1 and DBTG when considering distributed or clustered systems. In particular SQL needs to send fewer messages in order to perform distributed applications.

This benchmark demonstrated that SQL systems can have good performance. The system was benchmarked at over 200tps with no bottlenecks in sight. We believe that the benchmark could be extended to much larger systems. In addition, the benchmark demonstrated price performance in the range of 50K$ /tps to 60K$/tps for both departmental and mainframe computers. Again, this price performance compares favorably with that of non-relational systems. In particular it is comparable to the price performance of Tandem's "old" Encompass data management system.

REFERENCES

[Anon] Anon et al., "A Measure of Transaction Processing Power", Datamation, V. 31.7, April 1985, pp. 112-118.

[ANSI] "Database Language SQL", American National Standard X3.135-1986.

[Bartlett] J. Bartlett, "A NonStop Kernel", Proc 8th ACM SOSP, Dec. 1981.

[Bitton] D. Bitton, et al., "Benchmarking Database Systems: A Systematic Approach", Proc. 9th VLDB, Nov 1983.

[Date] *An Introduction to Database Systems*, Volume 1, Addison Wesley, April 1986.

[Gawlick] D. Gawlick, "Processing Hot Spots in High Performance Systems", Proc. IEEE Compcon, Feb. 1985.

[NonStop SQL] *Introduction to NonStop SQL*, Part No. 82317, Tandem Computers Inc, Cupertino, CA, March 1987.

[Workbook] *NonStop SQL Benchmark Workbook,* Part No. 84160, Tandem Computers Inc, Cupertino, CA, March 1987.

[Pathway] *Introduction to Pathway*, Part No. 82339, Tandem Computers Inc, Cupertino, CA, June 1985.

Distributed by
TANDEMCOMPUTERS
Corporate Information Center
19333 Vallco Parkway MS3-07
Cupertino, CA 95014-2599