Network Appliance
TPC Benchmark™ C
Full Disclosure Report for
Network Appliance F820 Filers,
Fujitsu PRIMEPOWER 850,
and Sybase Adaptive Server 12.5.0.1

First Edition
March 2002

Network Appliance, Inc.
495 East Java Drive
Sunnyvale, CA 94089
Telephone: +1 (408) 822-6000
Fax: +1 (408) 822-4501
Support telephone: +1 (888) 4-NETAPP
Information e-mail: info@netapp.com
Information Web: http://www.netapp.co

Copyright and trademark information

Copyright information

Benchmark ® Full Disclosure Report for NetApp filer, March 2002

Network Appliance Inc. believes that the information in this document is accurate as of the publication date. The information in this document is subject to change without notice. The information in this document is provided as is with no warranties of any kind. Network Appliance Inc. assumes no responsibility for any errors that may appear in this document.

The pricing information in this document, if any, is believed to accurately reflect the current prices as of the publication date. However, Network Appliance Inc. provides no warranty of the pricing information in this document.

Benchmark results are highly dependent upon workload, specific application requirements, and system design and implementation. Relative system performance will vary as a result of these and other factors. Therefore, this or any TPC Benchmark® C should not be used as a substitute for a specific customer application benchmark when critical capacity planning and/or product evaluation decisions are contemplated.

All performance data contained in this report was obtained in a rigorously controlled environment. Results obtained in other operating environments may vary significantly. Network Appliance Inc. does not warrant or represent that a user can or will achieve similar performance expressed in transactions per minute (tpmC®) or normalized price/performance ($/tpmC®). No warranty of system performance or price/performance is expressed or implied in this report.

Trademark information

NetApp, the Network Appliance logo are registered trademarks of Network Appliance Inc. in the United States and other countries.

Sybase Adaptive Server Enterprise is a registered trademark of Sybase Inc..

Dell and Dell Power Edge are registered trademarks of Dell Corporation.

PRIMEPOWER,SPARC64 are registered trademarks of Fujitsu Ltd, Fujitsu Technology Solutions and Fujitsu-Siemens Computers

Solaris is a registerd trademark of Sun Microsystems, Inc

TPC Benchmark, TPC-C, and tpmC are registered certification marks of the Transaction Processing Performance Council.

All other brand or product names mentioned herein are trademarks or registered trademarks of their respective owners and should be treated as such.

Preface

The TPC Benchmark™ C Standard Specification was developed by the Transaction Processing Performance Council. This benchmark exercises the system components necessary to perform tasks associated with that class of on-line transaction processing (OLTP) environments emphasizing a mixture of read-only and update intensive transactions. This is a complex OLTP application environment exercising a breadth of system components associated by such environments characterized by:

- The simultaneous execution of multiple transaction types that span a breadth of complexity
- On-line and deferred transaction execution modes
- Multiple on-line terminal sessions
- Moderate system and application execution time
- Significant disk input/output
- Transaction integrity (ACID properties)
- Non-uniform distribution of data access through primary and secondary keys
- Databases consisting of many tables with a wide variety of sizes, attributes, and relationships
- Contention on data access and update

This benchmark defines four on-line transactions and one deferred transaction, intended to emulate functions that are common to many OLTP applications. However, this benchmark does not reflect the entire range of OLTP requirements. The extent to which a customer can achieve the results reported by a vendor is highly dependent on how closely TPC-C approximates the customer application. The relative performance of systems derived from this benchmark does not necessarily hold for other workloads or environments. Extrapolations to any other environment are not recommended.

Benchmark results are highly dependent upon workload, specific application requirements, system design and implementation. Relative system performance will vary as a result of these and other factors. Therefore, TPC-C should not be used as a substitute for a specific customer application benchmarking when critical capacity planning and/or product evaluation decisions are contemplated.

The performance metric reported by TPC-C is a “business throughput” measuring the number of orders processed per minute. Multiple transactions are used to simulate the business activity of processing an order, and each transaction is subjected to a response time constraint. The performance metric for this benchmark is expressed in transactions-per-minute-C (tpmC). To be compliant with the TPC-C standard, all references to tpmC results must include the tpmC rate, the associated price-per-tpmC, and the availability date of the priced configuration.

Fujitsu PRIMEPOWER 850

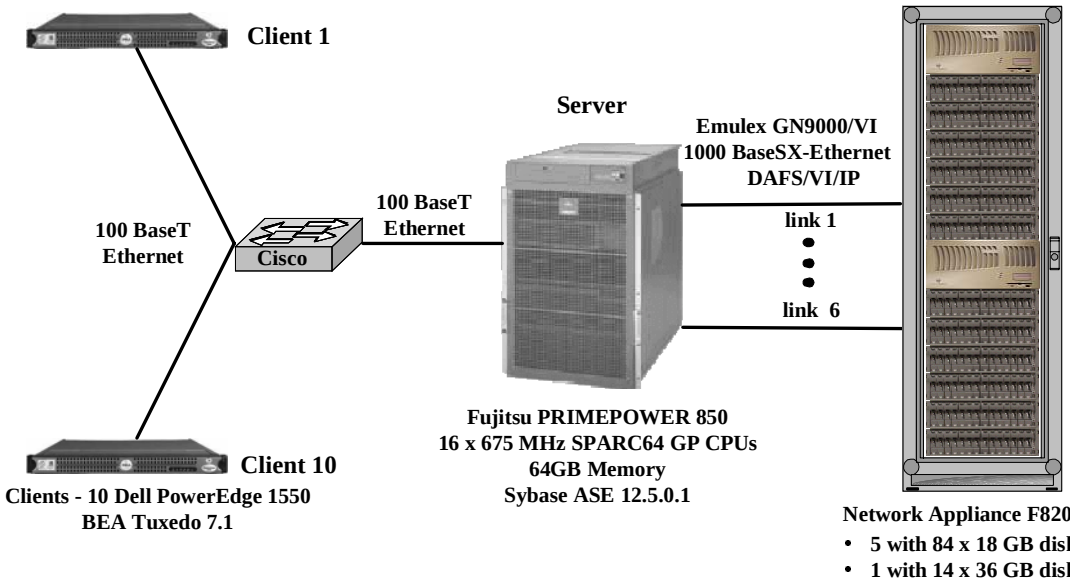
TPC-C Revision 5

Report Date: March 12, 2002

Total System Cost	TPC Throughput	Price/Performance	Availability Date
$1,508,712	112,286.46	$13.44	8/31/02

Processors	Database Manager	Operating System	Other Software	Number of Users
16 x 675 MHz SPARC64 GP CPUs	Sybase Adaptive Server Enterprise v12.5.0.1	Solaris 8	BEA Tuxedo 7.1	90,000

Priced configuration



System Components	Server		Each Client	
	Qty	Type	Qty	Type
Processors	16	675 MHz SPARC64 GP	2	1 GHz Pentium III
Cache memory	2x128K FLC, 8MB SLC per CPU		256 K per CPU	
Memory	64 GB		1 GB	
Disk controllers	1	Ultra SCSI	1	Ultra II SCSI
IP Storage HBA	6	Emulex GN9000/VI HBA		
Disk drives	5 with 84 x 18 GB 1 with 14 x 36 GB	Network Appliance F820		
Internal drive	2	36 GB Ultra III SCSI	1	18 GB SCSI
Total storage	8,136 GB		18 GB	
Tape drives	1	CD-ROM	1	CD-ROM
Terminals	None		None	

Fujitsu PRIMEPOWER 850

TPC-C Revision 5

Report Date: March 12, 2002

Description	Part Number	Brand	Price Key	Unit Price	Qty	Price	3 Year Maintenance
Database Server Hardware							
Fujitsu Technology Solutions PRIMEPOWER 850 includes:	PW0V0AR1U	Fujitsu	2	$180,000	1	$180,000	$92,160
675 MHz SPARC64 GP CPU	PW0V1A11U	Fujitsu	2	Bundled	4	Bundled	Bundled
System Board for Single Power Feed	PW0V7SB11U	Fujitsu	2	$37,500	1	$37,500	Bundled
675 MHz SPARC64 GP CPU	PW0V1A11U	Fujitsu	2	$17,500	12	$210,000	Bundled
4 GB Additional Memory	PW0R2M31U	Fujitsu	2	$3,600	16	$57,600	Bundled
Quad FastEthernet Card	X1034A-U	Fujitsu	2	$1,795	1	$1,795	Bundled
36 GB Hard Disk Drive	PW0R3D21U	Fujitsu	2	$1,600	2	$3,200	Bundled
Expansion Rack	PW-R3D21U	Fujitsu	2	$2,600	1	$2,600	Bundled
Outlet for 200 VAC	GP7N7RK94U	Fujitsu	2	$1,850	2	$3,700	Bundled
Power Conversion Option	PW0R7VC1U	Fujitsu	2	$185	2	$370	Bundled
PCIBOX	PW0R7BE1U	Fujitsu	2	$14,500	2	$29,000	Bundled
40 foot MTRJ to MTRJ Multimode Duplex Blue Fiber Jumper	FGRJ-4848-06-40	Compu-Link	6	$45	6	$270	N/A
Subtotal						**$526,035**	**$92,160**
Database Server Software							
Sybase ASE 12.5.01 for Solaris/SPARC		Sybase	3	$295,000	1	$295,000	$194,700
Solaris8 for SPARC	B23PLX0H0H	Fujitsu	2	$100	1	$100	Bundled
DDA/Software kit for Solaris/SPARC	SW-SPARC-DAFS	NetApp	1	$3,000	1	$3,000	Bundled
Subtotal						**$298,100**	**$194,700**
Network Appliance Storage							
DAFS Database Bundle w/ 4TB, includes:	BUN-DAFS-4TB	NetApp	1	$240,000	2	$480,000	Bundled
F820 Filer	F820-BASE-1GB	NetApp	1	Bundled	6	Bundled	Bundled
Copper FC-AL Adapter with Cable	X2040B-C	Qlogic	1	Bundled	16	Bundled	Bundled
DS14 w/ 18 GB Spindles & Cu LRUs		NetApp	1	Bundled	30	Bundled	Bundled
DS14 w/ 36 GB Spindles & Cu LRUs		NetApp	1	Bundled	1	Bundled	Bundled
Emulex GN9000/VI HBA	X1620	Emulex	1	Bundled	12	Bundled	Bundled
NFS Software F820	SW-F820-NFS	NetApp	1	Bundled	6	Bundled	Bundled
DAFS Software F820	SW-F820-DAFS	NetApp	1	Bundled	6	Bundled	Bundled
Smart UPS 1400 2U RM Powerchute	SU1400RM2U	APC	7	$705	1	$705	N/A
Subtotal						**$480,705**	**$0**
Client Hardware							
PowerEdge 1550, includes:	220-9993	Dell	4	$2,512	10	$25,116	Bundled
1 GHz Pentium III	311-5512	Dell	4	Bundled	20	Bundled	Bundled
1 GB SDRAM, 133 MHz (2 512 MB DIMMs)	311-0855	Dell	4	Bundled	10	Bundled	Bundled
18GB,10000RPM,U160,1 IN SCSI,Hard Drive	340-2468	Dell	4	Bundled	10	Bundled	Bundled
24X IDE Internal CD-ROM	3130317	Dell	4	Bundled	10	Bundled	Bundled
Floppy Disk Drive	340-2474	Dell	4	Bundled	10	Bundled	Bundled
Subtotal						**$25,116**	**$0**

Fujitsu PRIMEPOWER 850

TPC-C Revision 5

Report Date: March 12, 2002

Description	Part Number	Brand	Price Key	Unit Price	Qty	Price	3 Year Maintenance
Client Software							
RedHat Linux 6.2		RedHat	4	Free	10	Free	Bundled (Footnote 1)
Sybase Client Software/ASE 12.5.0.1		Sybase	3	Free	10	Free	$8,700
Tuxedo 7.1. for RedHat 6.2	TUX-CFS	BEA	5	$2,850	10	$28,500	$20,700
Subtotal						**$28,500**	**$29,400**
Client Connectivity							
3500 Ethernet switch, includes:	WS-C3524-XL-EN	Cisco	8	$1,977	2	$3,953	$2,873
24 10/100 ports		Cisco	8	Bundled	2	Bundled	Bundled
15 foot RJ45 to RJ45 Blue with Black Boots	T3131-BQ506-BA-40	Compu-Link	6	$4	11	$44	N/A
Subtotal						**$3,997**	**$2,873**
		Discount on Fujitsu Technology Solutions Hardware (Footnote 2)				**-$157,760**	**-$15,114**
					Total	**$1,204,694**	**$304,019**

Pricing:

1. Network Appliance, Inc.
2. Fujitsu Technology Solutions
3. Sybase
4. Dell
5. BEA Systems
6. Anixter
7. Alphanumeric Systems, Inc.
8. Cisco Systems, Inc.

Three Year Cost of Ownership:	**$1,508,712**
tpmC Rating:	**112,286.46**
$ / tpmC:	**$13.44**

Prices used in TPC benchmarks reflect actual prices a customer would pay for a one-time purchase of the stated components. Individually negotiated discounts are not permitted. Special prices based on assumptions about past or future purchases are not permitted. All discounts reflect standard pricing policies for the listed components. For complete details, see the pricing sections of the TPC benchmark specifications. If you find that the stated prices are not available according to these terms, please inform the TPC at pricing@tpc.org. Thank you.

Footnote 1: Redhat Linux support is bundled with Dell support on PowerEdge 1550.

Footnote 2: Discount level on Fujitsu Technology Solutions hardware is on equipment as configured.

Numerical quantities summary
Fujitsu PRIMEPOWER 850

MQTH, Computed Maximum Qualified Throughput		**112,286 tpmC**	
Response Times (in seconds)	**90th**	**Average**	**Maximum**
Menu	0.00	0.00	0.01
New order	0.55	0.29	13.31
Payment	0.48	0.23	10.13
Order-status	0.58	0.32	12.14
Delivery (interactive)	0.00	0.00	0.01
Delivery (deferred)	0.56	0.42	5.64
Stock level	0.36	0.17	5.04

Transaction Mix, in percent of total transactions	**Percentage**
New order	44.67
Payment	43.02
Order status	4.09
Delivery (deferred)	4.11
Stock level	4.11

Keying/Think Times

	Keying Time			**Think Time**		
	Minimum	Average	Maximum	Minimum	Average	Maximum
New order	18.01	18.01	18.02	0.00	12.28	119.98
Payment	3.01	3.01	3.02	0.00	12.28	120.00
Order Status	2.01	2.01	2.01	0.00	10.29	99.95
Delivery	2.01	2.01	2.02	0.00	5.29	49.98
Stock level	2.01	2.01	2.01	0.00	5.30	49.85

Test Duration	
Rampup	60 minutes
Measurement	120 minutes
Rampdown	3 minutes
Transactions during measurement	30,164,025

Checkpoints	
Number of checkpoints in measurement interval	4
Checkpoint interval	30 minutes

Table of Contents

General Items

Application code and definition statements

The application program (as defined in clause 2.1.7) must be disclosed. This includes, but is not limited to, the code implementing the five transactions and the terminal input output functions.

Appendix A, "Client source code," on page 41 and Appendix B, "Server source code," on page 77 contain the source code implemented in this benchmark.

Test sponsor

A statement identifying the benchmark sponsor(s) and other participating companies must be provided.

Network Appliance, Fujitsu-Siemens, and Sybase sponsored and conducted this TPC Benchmark C.

Parameter settings

Settings must be provided for all customer-tunable parameters and options which have been changed from the defaults found in actual products, including but not limited to:

- *Database options*
- *Recover/commit options*
- *Consistency/locking options*
- *Operating system and application configuration parameter*
- *This requirement can be satisfied by providing a full list of all parameters*

Appendix C, "Database Layout," on page 103 contains the parameters for the database, the operating system, and the configuration for the transaction monitor.

Configuration items

Diagrams of both measured and priced configurations must be provided, accompanied by a description of the differences.

The System Under Test (SUT), a PRIMEPOWER 850 c/s w/ 10 Front-Ends, is depicted in the following diagrams:

- "Figure 1: Tested configuration" on page 2
- "Figure 2: Priced configuration" on page 3

The differences between the configurations are as follows:

- The priced configuration has six filers. Five of the filers have six shelves of 18 GB disks and one filer has one shelf of 36 GB disks.
- The test configuration has four filers with six shelves of 36 GB disks. One filer has six shelves of 18 GB disks, and one filer has one shelf of 36 GB disks.
- In the test configuration, the clients are connected via 100 Base-T to an 100BT Ethernet Cisco switch, which in turn is connected via 100BT-Ethernet to the SUT.

Figure 1: Tested configuration

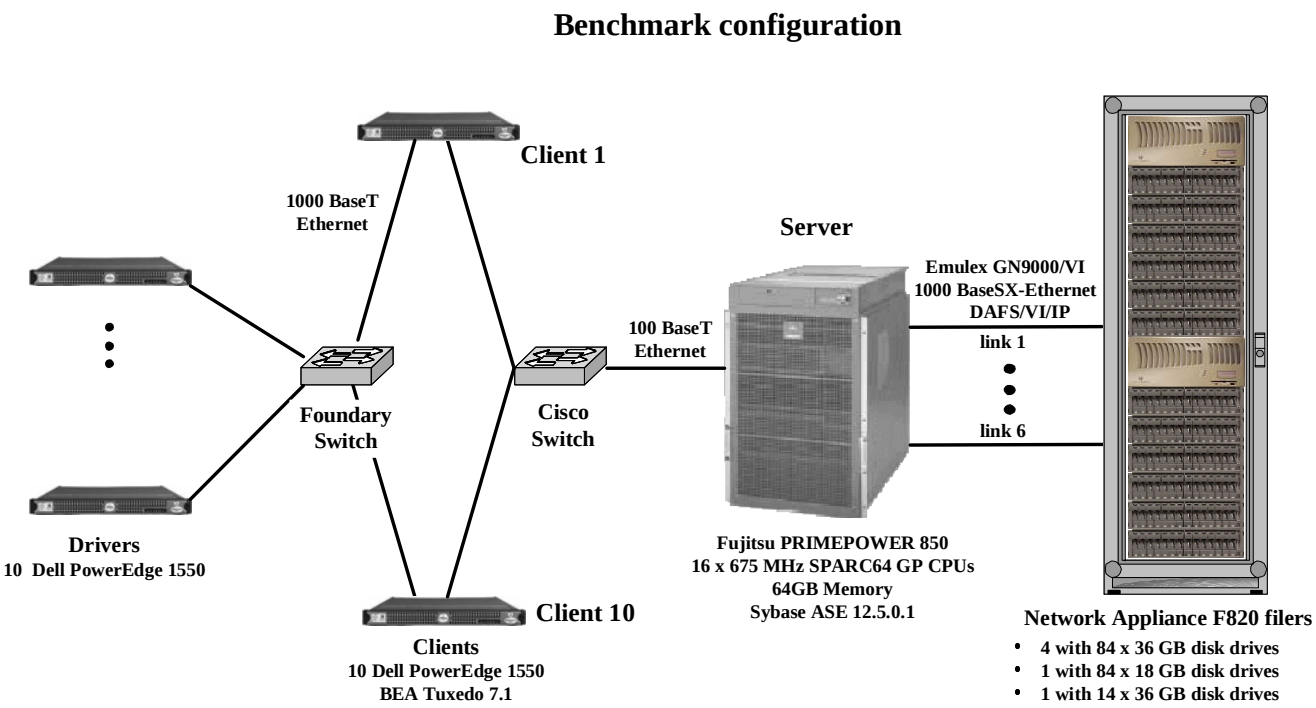
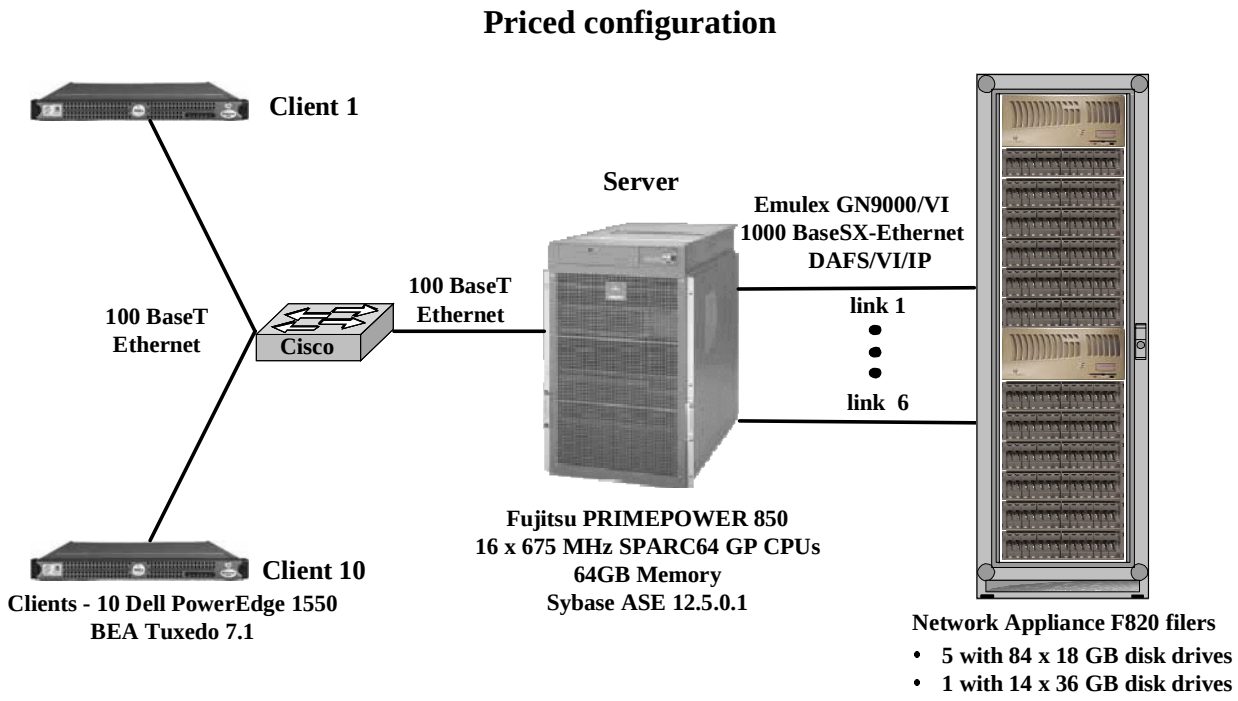


Figure 2: Priced configuration



1. Clause 1 — Logical Database Design

1.1 Table definitions

Listings must be provided for all table definition statements and all other statements used to set up the database. [Clause 8.1.2.1

Appendix C, "Database Layout," on page 103 lists the programs that define, create, and populate the database for this TPC benchmark™ C.

1.2 Physical organization of database

The physical organization of tables and indices, within the database, must be disclosed. [Clause 8.1.2.2]

Space was allocated to the Sybase Adaptive Server Enterprise v 12.5.0.1 according to the data in "Section 4.4. Database partitions/replications mapping" on page 23.

1.3 Insert and delete operations

It must be ascertained that insert and/or delete operations to any of the tables can occur concurrently with the TPC-C transaction mix. Furthermore, any restriction in the SUT database implementation that precludes inserts beyond the limits defined in Clause 1.4.11 must be disclosed. This includes the maximum number of rows that can be inserted and the maximum key value for these new rows. [Clause 8.1.2.3]

There were no restrictions on insert and delete operations to any tables.

1.4 Database partitioning

While there are a few restrictions placed upon horizontal or vertical partitioning of tables and rows in the TPC benchmark™ C (see Clause 1.6), any such partitioning must be disclosed. [Clause 8.1.2.4]

Partitioning was not used in this implementation.

1.5 Replication of tables

Replication of tables, if used, must be disclosed (see Clause 1.4.6). [Clause 8.1.2.5]

Replication was not used in this implementation.

1.6 Additional and/or duplicated attributes

Additional and/or duplicated attributes in any table must be disclosed along with a statement on the impact on performance (see Clause 1.4.7). [Clause 8.1.2.6]

Additional or duplicated attributes were not used in this implementation.

2. Clause 2 — Transaction and Terminal Profiles

2.1 Random number generation

The method of verification for the random number generation must be described. [Clause 8.1.3.1]

lrand48(), a 48-bit linear congruential random number generator provided by RedHat Linux 6.2 was used. A sample of 4000 random numbers from this generator were supplied to the auditor, who certified the suitability of this random number generator for use in TPC-C.

2.2 Input/output screen layout

The actual layouts of the terminal input/output screens must be disclosed. [Clause 8.1.3.2]

The screen layout corresponded exactly to those of the TPC-C Standard Specification (specified in Clause 2.4.3, 2.5.3, 2.6.3, 2.7.3, and 2.8.3).

2.3 Configured terminal features

The method used to verify that the emulated terminals provide all the features described in Clause 2.2.2.4 must be explained. Although not specifically priced, the type and model of the terminals used for the demonstration in 8.1.3.3 must be disclosed and commercially available (including supporting software and maintenance). [Clause 8.1.3.3]

The terminal features were verified by manually exercising each specification on a Dell PowerEdge 1550 workstation running a terminal emulator.

2.4 Presentation manager or intelligent terminal

Any usage of presentation managers or intelligent terminals must be explained. [Clause 8.1.3.4]

Application code running on the client machines implemented the TPC-C user interface. No presentation manager software or intelligent terminal features were used. The source code for the forms application is listed in Appendix A, "Client source code," on page 41.

2.5 Transaction statistics

The numerical quantities which are required are listed in the following table. [Clause 8.1.3.5 to 8.1.3.11]

Table 2 - 1 Transaction runtime statistics

Type	Item	Value
New order	Home warehouse	99.00%
	Remote warehouse	1.00%
	Rollback transactions	1.00%
	Orderline count	10.00%
Payment	Home warehouse	84.99%
	Remote warehouse	15.01%
	Customer name access	60.01%

Table 2 - 1 Transaction runtime statistics

Type	Item	Value
Order status	Customer name access	59.92%
Delivery	Skipped districts	0
Transaction mix	New order	44.67%
	Payment	43.02%
	Order status	4.09%
	Delivery	4.11%
	Stock level	4.11%

2.6 Queueing mechanism

The queueing mechanism used to defer the execution of the Delivery transaction must be disclosed. [Clause 8.1.12]

Delivery transactions were submitted to servers using the same TUXEDO mechanism that other transactions used. The only difference was that the call was asynchronous — control returned to the client process immediately and the deferred delivery completed asynchronously.

The source code is listed in Appendix A, “Client source code,” on page 41 and Appendix B, “Server source code,” on page 77.

3. Clause 3 — Transaction and System Properties

3.1 Transaction system properties (ACID)

The results of the ACID tests must disclosed along with a description of how the ACID requirements were met. This includes disclosing which case was followed for the execution of Isolation Test 7. [Clause 8.1.4.1]

The TPC Benchmark C Standard Specification defines a set of transaction processing system properties that a system under test (SUT) must support during the execution of the benchmark. Those properties are Atomicity, Consistency, Isolation, and Durability (ACID). This section quotes the specification definition of each of these properties and describes the tests done as specified and monitored by the auditor to demonstrate compliance.

3.2 Atomicity

The system under test must guarantee that transactions are atomic; the system will either perform all individual operations on the data, or will assure that no partially-completed operations leave any effects on the data.

3.2.1 Completed Transaction

Perform the Payment transaction for a randomly selected warehouse, district, and customer (by customer number as specified in Clause 2.5.1.2) and verify that the records in the CUSTOMER, WAREHOUSE, and DISTRICT tables have been changed appropriately.

The values of w_ytd, d_ytd, c_balance, c_ytd_payment, c_payment_cnt, c_data1, and c_data2, of a randomly selected warehouse, district, and customer were retrieved. The Payment transaction was executed on the same warehouse, district, and customer. The transaction was committed. The values w_ytd, d_ytd, c_balance, c_ytd_payment, c_payment_cnt, c_data1, and c_data2 were retrieved again. It was verified that all values had been changed appropriately.

3.2.2 Aborted Transaction

Perform the Payment transaction for a randomly selected warehouse, district, and customer (by customer number as specified in Clause 2.5.1.2) and substitute a ROLLBACK of the transaction for the COMMIT of the transaction. Verify that the records in the CUSTOMER, WAREHOUSE, and DISTRICT tables have NOT been changed

The values of w_ytd, d_ytd, c_balance, c_ytd_payment, c_payment_cnt, c_data1, and c_data2 of a randomly selected warehouse, district, and customer were retrieved. The Payment transaction was executed on the same warehouse, district, and customer. The transaction was rolled back. The values of w_ytd, d_ytd, c_balance, c_ytd_payment, c_payment_cnt, c_data1, and c_data2 were retrieved again. It was verified that none of the values had changed.

3.3 Consistency

Consistency is the property of the application that requires any execution of a database transaction to take the database from one consistent state to another assuming the database is initially in a consistent state.

The TPC Benchmark C standard requires the System Under Test to meet the following 12 consistency conditions (c.f. TPC Standard Specification, Clauses 3.3.2.1 to 3.3.2.12):

1. the sum of the district balances in a warehouse is equal to the warehouse balance;
2. for each district, the next order-id minus one is equal to maximum order-id in the ORDER table and equal to the maximum new-order-id in the NEW-ORDER table;
3. for each district, the maximum order-id minus minimum order-id in the ORDER table plus one equals the number of rows in the NEW-ORDER table for that district;

4. for each district, the sum of the order-line counts equals the number of rows in the ORDER-LINE table for that district;
5. for each row in the ORDER table, the carrier-id is set to a null value only if there is a corresponding row in the NEW-ORDER table;
6. for each row in the ORDER table, the order-line count must equal the number of rows in the ORDER-LINE table for that order;
7. for any row in the ORDER-LINE table, the delivery date/time is set to a null value only if the corresponding row in the ORDER table has the carrier-id set to a null value;
8. for each warehouse, the year-to-date amount must equal the sum of the amounts in the HISTORY table for that warehouse;
9. for each district, the year-to-date amount must equal the sum of the amounts in the HISTORY table for that district;
10. for each customer, the balance must equal the sum of the order-line amount minus the sum of the history amount for that customer;
11. for each district, the total orders minus the total new-orders must equal the sum of the customer delivery count;
12. for any randomly selected customer, the balance plus the year-to-date payment must equal the sum of the order-line amount.

The TPC Benchmark C Standard Specification requires explicit demonstration that the conditions are satisfied for the first four conditions only.

To demonstrate that consistency is maintained, conditions 1-4 were verified for a sample of warehouses. Condition 3 was verified both before and after the durability tests, as required in the durability test section 3.5.4.

3.4 Isolation

Operations of concurrent transactions must yield results which are indistinguishable from the results which would be obtained by forcing each transaction to be serially executed to completion in some order. This property is commonly called serializability. Sufficient conditions must be enabled at either the system or application level to ensure serializability of transactions under any arbitrary mix of TPC-C transactions, unless otherwise specified by the transaction profile. The system or application must have full serializability enabled (i.e., repeated reads of the same rows within any committed transaction must return identical data when run concurrently with any arbitrary mix of TPC-C transactions), except in the case of Stock-Level transaction. For the Stock-Level transaction, the isolation requirement is relaxed to simply require that the transaction see only committed data.

The TPC Benchmark C Standard (Revision 3) defines nine required tests performed to demonstrate that the required levels of transaction isolation are met.

For conventional locking schemes, isolation should be tested as described below. Systems that implement other isolation schemes may require different validation techniques. It is the responsibility of the test sponsor to disclose those techniques and the tests for them. If isolation schemes other than conventional locking are used, it is permissible to implement these tests differently provided full details are disclosed. (Examples of different validation techniques are shown in Isolation Test 7, Clause 3.4.2.7).

3.4.1 Isolation test 1

This test demonstrates isolation for read-write conflicts of Order-Status and New-Order transactions.

The execution of the above test proceeded as follows:

1. A New-Order transaction T1 was executed for a randomly selected customer. T1 was stopped prior to commit.

2. An Order-Status transaction T2 was started for the same customer used in T1.
3. Transaction T2 waited.
4. Transaction T1 finished. Immediately after T1 finished, T2 completed. Transaction T2 returned the order inserted by T1.

3.4.2 Isolation test 2

The execution of the above test proceeded as follows:

1. An Order-Status transaction T0 was executed for a randomly selected customer. T0 was allowed to complete.
2. A New-Order transaction T1 was started for the same customer used in T1. T1 was stopped after executing all statements before the final ROLLBACK.
3. An Order-Status transaction T2 was started for the same customer used in T0. T2 attempted to read the data for the order T1 created. T2 waited.
4. Transaction T1 was rolled back.
5. T2 completed. The data returned by T2 matched the data returned by T0.

3.4.3 Isolation test 3

The execution of the above test proceeded as follows:

1. A New-Order transaction T1 was started for a randomly selected customer. T1 was stopped immediately prior to COMMIT.
2. Another New-Order transaction T2 was started for the same customer as in T1.
3. Transaction T2 waited.
4. Transaction T1 completed, and transaction T2 completed immediately afterwards.
5. The order number returned by T2 was one greater than the order number returned by T1. The updated value of D_NEXT_O_ID was incremented by two from its value before the update by T1 and was one greater than the order number of the order created by T2.

3.4.4 Isolation test 4

The execution of the above test proceeded as follows:

1. A New-Order transaction T1 was started for a randomly selected customer. T1 was stopped immediately prior to ROLLBACK.
2. Another New-Order transaction T2 was started for the same customer as in T1.
3. Transaction T2 waited
4. Transaction T1 rolled back and completed. T2 committed and completed immediately afterwards.
5. The order number for T2 was the same as the starting value of D_NEXT_O_ID seen by T1. After T2 was done, the value of D_NEXT_O_ID was incremented by only one from its value at the start of T1.

3.4.5 Isolation test 5

The execution of the above test proceeded as follows:

1. A Delivery transaction T1 was started for a randomly selected warehouse and district.
2. Transaction T1 was stopped immediately prior to COMMIT.
3. A Payment transaction T2 was started for the same customer as the lowest NO_O_ID being delivered by T1.
4. Transaction T2 waited.
5. T1 completed. T2 completed immediately afterwards.
6. C_BALANCE reflected the delivery of the order by T1 and the payment by T2.

3.4.6 Isolation test 6

The execution of the above test proceeded as follows:

1. A Delivery transaction T1 was started for a randomly selected warehouse and district.
2. Transaction T1 was stopped immediately prior to ROLLBACK.
3. A Payment transaction T2 was started for the same customer as the lowest NO_O_ID being delivered by T1.
4. Transaction T2 waited.
5. T1 rolled back and completed. T2 completed immediately afterwards.
6. C_BALANCE reflected the payment by T2 but not the delivery in T1 that was rolled back.

3.4.7 Isolation test 7

The execution of the above test proceeded as follows:

1. Transaction T1 was started, which queried the prices for two items (x and y).
2. A New-Order transaction T2 was started for a group of items including item x twice and item y once.
3. T2 was stopped after querying the price of item x once, but before querying the price of item x again and before querying the price of item y.
4. Transaction T3 started, to update the prices of items x and y.
5. Case A occurred, i.e. T3 waited.
6. T2 committed and completed. The prices read by T2 for x (the second time) and y were the same as the prices read by T1.
7. T3 completed immediately after T2 completed.
8. Transaction T4 started. T4 queried the price for items x and y, and they were the updated prices changed by T3.

3.4.8 Isolation test 8

The execution of the above test proceeded as follows:

1. All the rows for a randomly selected district and warehouse were removed from the NEW-ORDER table.
2. A delivery transaction T1 was started for the selected warehouse.

3. Inside T1, the NEW-ORDER table was read, but no qualifying rows were found. T1 was stopped at this point.
4. A New-Order transaction T2 was started for the same warehouse and district and a randomly selected customer.
5. Transaction T2 stalled (this is Case A).
6. Transaction T1 started again and again tried to read the NEW-ORDER table. Again, no qualifying rows were found.
7. T1 committed and completed, then T2 completed.

3.4.9 Isolation test 9

The execution of the above test proceeded as follows:

1. An Order-Status transaction T1 was started for a randomly selected customer.
2. T1 was stopped immediately after reading the ORDERS table for the selected customer and finding his most recent order.
3. A New-Order transaction T2 was started for the same customer as in T1.
4. Transaction T2 stalled (this is Case A).
5. T1 repeated the read of the ORDERS table for the selected customer. The most recent order found was the same as the one found in the first read.
6. T1 committed and completed.
7. T2 then completed.

3.5 Durability

The tested system must guarantee durability: the ability to preserve the effects of committed transaction and insure database consistency after recovery from any one of the failures listed in Clause 3.5.3.

List of single failures:

- *Permanent irrecoverable failure of any single durable medium containing TPC-C database tables or recovery log data.*
- *Instantaneous interruption (system crash / system hang) in processing which requires system reboot to recover.*
- *Failure of all or part of memory (loss of contents)…*

[Clause 3.5.3]

The intent of these tests is to demonstrate that all transactions whose output messages have been received at the terminal or RTE have in fact been committed in spite of any single failure from the list in Clause 3.5.3 and that all consistency conditions are still met after the database is recovered.

It is required that the system crash test(s) and the loss of memory test(s) described in Clause 3.5.3.2 and 3.5.3.3 be performed under full terminal load and a fully scaled database. The durable media failure test(s) described in Clause 3.5.3.1 may be performed on a subset of the SUT configuration and database. For the SUT subset, all multiple hardware components, such as processors and disk / controllers in the full SUT configuration, must be represented by the greater of 10% of the configuration or two of each of the multiple hardware components. The database must be scaled to at least 10% of the fully scaled database, with a minimum of two warehouses.... Furthermore, the standard driving mechanism must be used in this test. The test sponsor must state that to the best of their knowledge, a fully scaled test would also pass all durability tests. [Clause 3.5.4]

3.5.1 Loss of data disk or log disk

Because the log and data-storage devices are Redundant Disk Arrays which each function independently of the rest of the system in ensuring data integrity under loss and/or replacement of any individual disk drive (and other failures as well), integrity under such failure and replacement does not entail any interruption in processing. The test below validates the durability by demonstrating persistence of the results of transactions processed both before and during these failures, validating the durability upon database recovery (in this instance, forced) of transactions which completed before the failure and the non-effect of transactions which did not complete.

1. The D_NEXT_O_ID fields for all rows in the DISTRICT table were summed up to determine the initial count of the total number of orders (count1).
2. A test was initiated with 20,000 terminals. On the driver system, completed/rolled-back New-Order transactions were recorded in a “success” file.
3. After 15 minutes, one of the individual disks containing the Sybase recovery logs was unplugged from its array. After unplugging the disk, system processing continued normally, given the array was configured with redundancy. After another 5 minutes, a disk was pulled from an array containing data. After unplugging the disk, system processing continued normally, given the array was configured with redundancy.
4. The test was halted.
5. Step 1 was repeated to determine the total number of orders (count2). Count2-count1 was 2 greater than the number of records for successful New Orders in the RTE “success” file which is within the “in-flight” limits allowed given 114,400 users.
6. Consistency check 3 was run before and after the benchmark run and the results were verified.

3.5.2 Instantaneous interruption and loss of memory

Instantaneous interruption and loss of memory tests were combined because the loss of power erases the contents of memory. This failure was induced while the benchmark was running by turning off the power supplies to the server. These tests were performed on a full-scale database with 90,000 users.

1. The D_NEXT_O_ID fields for all rows in district table were summed up to determine the initial count of the total number of orders (count1).
2. Transactions were started at full load. On the driver system, completed/rolled-back New-Order transactions were recorded in a “success” file.
3. After fifteen minutes the server system was de-powered.
4. The test was aborted on the driver.
5. The server system was restarted.
6. The database was restarted and a recovery performed using the transaction log.
7. The contents of the “success” file on the driver and the orders table were compared to verify that records in the “success” file for completed New-Order transactions had corresponding records in the orders table.
8. The auditor verified that the total number of orders committed by the database was within the “in-flight” limit allowed based on the number of New Orders in the RTE “success” file.
9. Consistency check 3 was run before and after the benchmark run and the results were verified.

3.5.3 Loss of log filer

The loss of log filer test was performed because the memory of the filer is considered stable storage with UPS protection. This failure was induced while the benchmark was running by turning off the power supplies to the server. This test was performed on a full-scale database with 90,000 users.

1. The D_NEXT_O_ID fields for all rows in district table were summed up to determine the initial count of the total number of orders (count1).
2. Transactions were started at full load. On the driver system, completed/rolled-back New-Order transactions were recorded in a “success” file.
3. After fifteen minutes the filer was de-powered.
4. The test was aborted on the driver.
5. The server system was restarted.
6. The database was restarted and a recovery performed using the transaction log.
7. The contents of the “success” file on the driver and the orders table were compared to verify that records in the “success” file for completed New-Order transactions had corresponding records in the orders table.
8. The auditor verified that the total number of orders committed by the database was within the “in-flight” limit allowed based on the number of New Orders in the RTE “success” file.
9. Consistency check 3 was run before and after the benchmark run and the results were verified.

4. Clause 4 — Scaling and Database Population

4.1 Initial cardinality of tables

The cardinality (e.g., the number of rows) of each table, as it existed at the start of the benchmark run (see Clause 4.2), must be disclosed. If the database was over-scaled and inactive rows of the WAREHOUSE table were deleted (see Clause 4.2.2), the cardinality of the WAREHOUSE table as initially configured and the number of rows deleted must be disclosed. [Clause 8.1.5.1]

The TPC-C database for this test was configured with 10,000 warehouses.

Table 4 - 1 Number of rows

Table	**Number of records**
Warehouse	10,000
District	100,000
Customer	300,000
History	300,000
Orders	30,000
New orders	90,000
Order line	3,000,000
Item	1,000,000
Stock	1,000,000
Deleted warehouses	0

4.2 Distribution of tables and logs

The distribution of tables and logs across all media must be explicitly depicted for the tested and priced systems. [Clause 8.1.5.2]

The following table lists the distribution of tables and logs across media.

Table 4 - 1 Device table

Controller	**Filer**	**Volume**	**File**	**Size**
/dev/orvi0	bashir	tpccvol16	tpcc_cust35	4,818,206,720
/dev/orvi0	bashir	tpccvol16	tpcc_history6	2,526,019,584
/dev/orvi0	bashir	tpccvol16	tpcc_cust32	4,818,206,720
/dev/orvi0	bashir	tpccvol16	tpcc_cust33	4,818,206,720
/dev/orvi0	bashir	tpccvol16	tpcc_cust9	4,818,206,720
/dev/orvi0	bashir	tpccvol16	tpcc_norder8	524,288,000
/dev/orvi0	bashir	tpccvol16	tpcc_ordl16	9,332,326,400

Table 4 - 1 Device table

Controller	Filer	Volume	File	Size
/dev/orvi0	bashir	tpccvol16	tpcc_stock16	7,303,331,840
/dev/orvi0	bashir	tpccvol16	tpcc_stock40	7,303,331,840
/dev/orvi0	bashir	tpccvol16	tpcc_cust11	4,818,206,720
/dev/orvi0	bashir	tpccvol16	tpcc_stock35	7,303,331,840
/dev/orvi0	bashir	tpccvol24	tpcc_cust3	4,818,206,720
/dev/orvi0	bashir	tpccvol24	tpcc_awdino1	460,324,864
/dev/orvi0	bashir	tpccvol24	tpcc_cust1	4,818,206,720
/dev/orvi0	bashir	tpccvol24	tpcc_cust25	4,818,206,720
/dev/orvi0	bashir	tpccvol24	tpcc_cust8	4,818,206,720
/dev/orvi0	bashir	tpccvol24	tpcc_orders8	2,186,280,960
/dev/orvi0	bashir	tpccvol24	tpcc_ordl24	9,332,326,400
/dev/orvi0	bashir	tpccvol24	tpcc_stock24	7,303,331,840
/dev/orvi0	bashir	tpccvol24	tpcc_stock48	7,303,331,840
/dev/orvi0	bashir	tpccvol24	tpcc_cust46	4,818,206,720
/dev/orvi0	bashir	tpccvol24	tpcc_ordl3	9,332,326,400
/dev/orvi0	bashir	tpccvol8	tpcc_cust19	4,818,206,720
/dev/orvi0	bashir	tpccvol8	tpcc_cust16	4,818,206,720
/dev/orvi0	bashir	tpccvol8	tpcc_cust17	4,818,206,720
/dev/orvi0	bashir	tpccvol8	tpcc_cust41	4,818,206,720
/dev/orvi0	bashir	tpccvol8	tpcc_history8	2,526,019,584
/dev/orvi0	bashir	tpccvol8	tpcc_ordl8	9,332,326,400
/dev/orvi0	bashir	tpccvol8	tpcc_stock32	7,303,331,840
/dev/orvi0	bashir	tpccvol8	tpcc_stock8	7,303,331,840
/dev/orvi0	bashir	tpccvol8	tpcc_history3	2,526,019,584
/dev/orvi0	bashir	tpccvol8	tpcc_cust27	4,818,206,720
/dev/orvi5	betor	tpccvol12	tpcc_awdino3	460,324,864
/dev/orvi5	betor	tpccvol12	tpcc_norder6	524,288,000
/dev/orvi5	betor	tpccvol12	tpcc_cust13	4,818,206,720
/dev/orvi5	betor	tpccvol12	tpcc_cust37	4,818,206,720

Table 4 - 1 Device table

Controller	Filer	Volume	File	Size
/dev/orvi5	betor	tpccvol12	tpcc_norder4	524,288,000
/dev/orvi5	betor	tpccvol12	tpcc_ordl1	9,332,326,400
/dev/orvi5	betor	tpccvol12	tpcc_ordl12	9,332,326,400
/dev/orvi5	betor	tpccvol12	tpcc_ordl17	9,332,326,400
/dev/orvi5	betor	tpccvol12	tpcc_stock12	7,303,331,840
/dev/orvi5	betor	tpccvol12	tpcc_stock36	7,303,331,840
/dev/orvi5	betor	tpccvol12	tpcc_norder1	524,288,000
/dev/orvi5	betor	tpccvol12	tpcc_cust30	4,818,206,720
/dev/orvi5	betor	tpccvol20	tpcc_ordl14	9,332,326,400
/dev/orvi5	betor	tpccvol20	tpcc_ordl6	9,332,326,400
/dev/orvi5	betor	tpccvol20	tpcc_awdino5	460,324,864
/dev/orvi5	betor	tpccvol20	tpcc_cust29	4,818,206,720
/dev/orvi5	betor	tpccvol20	tpcc_cust5	4,818,206,720
/dev/orvi5	betor	tpccvol20	tpcc_orders4	2,186,280,960
/dev/orvi5	betor	tpccvol20	tpcc_ordl20	9,332,326,400
/dev/orvi5	betor	tpccvol20	tpcc_stock20	7,303,331,840
/dev/orvi5	betor	tpccvol20	tpcc_stock44	7,303,331,840
/dev/orvi5	betor	tpccvol20	tpcc_temp1	2,097,152,000
/dev/orvi5	betor	tpccvol20	tpcc_orders1	2,186,280,960
/dev/orvi5	betor	tpccvol20	tpcc_cust14	4,818,206,720
/dev/orvi5	betor	tpccvol4	tpcc_cust43	4,818,206,720
/dev/orvi5	betor	tpccvol4	tpcc_cust21	4,818,206,720
/dev/orvi5	betor	tpccvol4	tpcc_cust45	4,818,206,720
/dev/orvi5	betor	tpccvol4	tpcc_history4	2,526,019,584
/dev/orvi5	betor	tpccvol4	tpcc_ordl4	9,332,326,400
/dev/orvi5	betor	tpccvol4	tpcc_stock28	7,303,331,840
/dev/orvi5	betor	tpccvol4	tpcc_stock4	7,303,331,840
/dev/orvi5	betor	tpccvol4	tpcc_stock33	7,303,331,840
/dev/orvi5	betor	tpccvol4	tpcc_temp2	2,097,152,000

Table 4 - 1 Device table

Controller	Filer	Volume	File	Size
/dev/orvi8	decker	tpccvol13	tpcc_stock6	7,303,331,840
/dev/orvi8	decker	tpccvol13	tpcc_cust12	4,818,206,720
/dev/orvi8	decker	tpccvol13	tpcc_cust36	4,818,206,720
/dev/orvi8	decker	tpccvol13	tpcc_norder5	524,288,000
/dev/orvi8	decker	tpccvol13	tpcc_ordl13	9,332,326,400
/dev/orvi8	decker	tpccvol13	tpcc_stock1	7,303,331,840
/dev/orvi8	decker	tpccvol13	tpcc_stock13	7,303,331,840
/dev/orvi8	decker	tpccvol13	tpcc_stock37	7,303,331,840
/dev/orvi8	decker	tpccvol13	tpcc_stock3	7,303,331,840
/dev/orvi8	decker	tpccvol21	tpcc_stock30	7,303,331,840
/dev/orvi8	decker	tpccvol21	tpcc_awdino4	460,324,864
/dev/orvi8	decker	tpccvol21	tpcc_cust28	4,818,206,720
/dev/orvi8	decker	tpccvol21	tpcc_cust4	4,818,206,720
/dev/orvi8	decker	tpccvol21	tpcc_orders5	2,186,280,960
/dev/orvi8	decker	tpccvol21	tpcc_ordl21	9,332,326,400
/dev/orvi8	decker	tpccvol21	tpcc_stock21	7,303,331,840
/dev/orvi8	decker	tpccvol21	tpcc_stock45	7,303,331,840
/dev/orvi8	decker	tpccvol21	tpcc_stock27	7,303,331,840
/dev/orvi8	decker	tpccvol21	tpcc_awdino6	460,324,864
/dev/orvi8	decker	tpccvol21	tpcc_master	314,572,800
/dev/orvi8	decker	tpccvol5	tpcc_cust48	4,818,206,720
/dev/orvi8	decker	tpccvol5	tpcc_cust20	4,818,206,720
/dev/orvi8	decker	tpccvol5	tpcc_cust24	4,818,206,720
/dev/orvi8	decker	tpccvol5	tpcc_cust44	4,818,206,720
/dev/orvi8	decker	tpccvol5	tpcc_history5	2,526,019,584
/dev/orvi8	decker	tpccvol5	tpcc_ordl5	9,332,326,400
/dev/orvi8	decker	tpccvol5	tpcc_stock29	7,303,331,840
/dev/orvi8	decker	tpccvol5	tpcc_stock5	7,303,331,840
/dev/orvi8	decker	tpccvol5	tpcc_cust22	4,818,206,720

Table 4 - 1 Device table

Controller	Filer	Volume	File	Size
/dev/orvi8	decker	tpccvol5	tpcc_orders6	2,186,280,960
/dev/orvi6	keiko	tpccvol10	tpcc_stock22	7,303,331,840
/dev/orvi6	keiko	tpccvol10	tpcc_cust15	4,818,206,720
/dev/orvi6	keiko	tpccvol10	tpcc_cust39	4,818,206,720
/dev/orvi6	keiko	tpccvol10	tpcc_history10	2,526,019,584
/dev/orvi6	keiko	tpccvol10	tpcc_norder2	524,288,000
/dev/orvi6	keiko	tpccvol10	tpcc_ordl10	9,332,326,400
/dev/orvi6	keiko	tpccvol10	tpcc_ordl9	9,332,326,400
/dev/orvi6	keiko	tpccvol10	tpcc_stock10	7,303,331,840
/dev/orvi6	keiko	tpccvol10	tpcc_stock34	7,303,331,840
/dev/orvi6	keiko	tpccvol10	tpcc_stock11	7,303,331,840
/dev/orvi6	keiko	tpccvol18	tpcc_stock38	7,303,331,840
/dev/orvi6	keiko	tpccvol18	tpcc_awdino7	460,324,864
/dev/orvi6	keiko	tpccvol18	tpcc_cust31	4,818,206,720
/dev/orvi6	keiko	tpccvol18	tpcc_cust7	4,818,206,720
/dev/orvi6	keiko	tpccvol18	tpcc_orders2	2,186,280,960
/dev/orvi6	keiko	tpccvol18	tpcc_ordl18	9,332,326,400
/dev/orvi6	keiko	tpccvol18	tpcc_stock17	7,303,331,840
/dev/orvi6	keiko	tpccvol18	tpcc_stock18	7,303,331,840
/dev/orvi6	keiko	tpccvol18	tpcc_stock42	7,303,331,840
/dev/orvi6	keiko	tpccvol18	tpcc_history1	2,526,019,584
/dev/orvi6	keiko	tpccvol18	tpcc_stock43	7,303,331,840
/dev/orvi6	keiko	tpccvol2	tpcc_stock41	7,303,331,840
/dev/orvi6	keiko	tpccvol2	tpcc_cust23	4,818,206,720
/dev/orvi6	keiko	tpccvol2	tpcc_cust47	4,818,206,720
/dev/orvi6	keiko	tpccvol2	tpcc_history2	2,526,019,584
/dev/orvi6	keiko	tpccvol2	tpcc_ordl2	9,332,326,400
/dev/orvi6	keiko	tpccvol2	tpcc_stock2	7,303,331,840
/dev/orvi6	keiko	tpccvol2	tpcc_stock25	7,303,331,840

Table 4 - 1 Device table

Controller	Filer	Volume	File	Size
/dev/orvi6	keiko	tpccvol2	tpcc_stock26	7,303,331,840
/dev/orvi6	keiko	tpccvol2	tpcc_awdino8	460,324,864
/dev/orvi6	keiko	tpccvol2	tpcc_stock19	7,303,331,840
/dev/orvi2	pike	logs	tpcc_log1	33,554,432,000
/dev/orvi2	pike	logs	tpcc_log2	33,554,432,000
/dev/orvi2	pike	logs	tpcc_log3	33,554,432,000
/dev/orvi7	wynn	tpccvol15	tpcc_ordl22	9,332,326,400
/dev/orvi7	wynn	tpccvol15	tpcc_cust10	4,818,206,720
/dev/orvi7	wynn	tpccvol15	tpcc_cust34	4,818,206,720
/dev/orvi7	wynn	tpccvol15	tpcc_cust40	4,818,206,720
/dev/orvi7	wynn	tpccvol15	tpcc_norder7	524,288,000
/dev/orvi7	wynn	tpccvol15	tpcc_ordl15	9,332,326,400
/dev/orvi7	wynn	tpccvol15	tpcc_stock15	7,303,331,840
/dev/orvi7	wynn	tpccvol15	tpcc_stock39	7,303,331,840
/dev/orvi7	wynn	tpccvol15	tpcc_cust38	4,818,206,720
/dev/orvi7	wynn	tpccvol15	tpcc_norder3	524,288,000
/dev/orvi7	wynn	tpccvol23	tpcc_stock14	7,303,331,840
/dev/orvi7	wynn	tpccvol23	tpcc_awdino2	460,324,864
/dev/orvi7	wynn	tpccvol23	tpcc_cust2	4,818,206,720
/dev/orvi7	wynn	tpccvol23	tpcc_cust26	4,818,206,720
/dev/orvi7	wynn	tpccvol23	tpcc_orders7	2,186,280,960
/dev/orvi7	wynn	tpccvol23	tpcc_ordl23	9,332,326,400
/dev/orvi7	wynn	tpccvol23	tpcc_stock23	7,303,331,840
/dev/orvi7	wynn	tpccvol23	tpcc_stock47	7,303,331,840
/dev/orvi7	wynn	tpccvol23	tpcc_cust6	4,818,206,720
/dev/orvi7	wynn	tpccvol23	tpcc_orders3	2,186,280,960
/dev/orvi7	wynn	tpccvol7	tpcc_stock46	7,303,331,840
/dev/orvi7	wynn	tpccvol7	tpcc_cust18	4,818,206,720
/dev/orvi7	wynn	tpccvol7	tpcc_cust42	4,818,206,720

Table 4 - 1 Device table

Controller	Filer	Volume	File	Size
/dev/orvi7	wynn	tpccvol7	tpcc_history7	2,526,019,584
/dev/orvi7	wynn	tpccvol7	tpcc_ordl7	9,332,326,400
/dev/orvi7	wynn	tpccvol7	tpcc_stock31	7,303,331,840
/dev/orvi7	wynn	tpccvol7	tpcc_stock7	7,303,331,840
/dev/orvi7	wynn	tpccvol7	tpcc_stock9	7,303,331,840
/dev/orvi7	wynn	tpccvol7	tpcc_ordl11	9,332,326,400
/dev/orvi7	wynn	tpccvol7	tpcc_ordl19	9,332,326,400

4.3 Database model, interface, and access language

A statement must be provided that describes:

1. *The data model implemented by the DBMS used (e.g., relational, network, hierarchical)*
2. *The database interface (e.g., embedded, call level) and access language (e.g., SQL, DL/1, COBOL read/write) used to implement the TPC-C transactions. If more than one interface/access language is used to implement TPC-C, each interface / access language must be described and a list of which interface/access language is used with which transaction type must be disclosed.*

[Clause 8.1.5.3]

Sybase Adaptive Server Enterprise v12.5.0.1 is a relational DBMS. SQL stored procedures were used, invoked through the Sybase Open Client DB-Library; the application code is in Appendix B, “Server source code,” on page 77.

4.4 Database partitions/replications mapping

The mapping of database partitions/replications must be explicitly described.

No partitioning or replication was used.

4.5 60-day space calculation

Details of the 60 day space computations along with proof that the database is configured to sustain 8 hours for the dynamic tables (Order, Order-Line, and History) must be disclosed.

The 60-day space calucation is listed in Appendix E, “60-Day Space Calculations,” on page 183.

5. Clause 5 — Performance Metrics and Response Times

5.1 Measured tpmC

Measured tpmC must be reported. [Clause 8.1.6.1]

During the 120-minute measurement period on the PRIMEPOWER 850 the throughput measured was 112,286.46 tpmC.

5.2 Response times

Ninetieth percentile, maximum and average response times must be reported for all transaction types as well as for the Menu response time. [Clause 8.1.6.2]

Table 5 - 1 Response times

Response times	90th Percentile	Average	Maximum
Menu	0.00	0.00	0.01
New order	0.55	0.29	13.31
Payment	0.48	0.23	10.13
Order-status	0.58	0.32	12.14
Delivery (interactive)	0.00	0.00	0.01
Delivery (deferred)	0.56	0.42	5.64
Stock level	0.36	0.17	5.04

5.3 Keying and think times

The minimum, the average, and the maximum keying and think times must be reported for each transaction type. [Clause 8.1.6.3]

Table 5 - 1 Keying times

Keying times	Minimum	Average	Maximum
New order	18.01	18.01	18.02
Payment	3.01	3.01	3.02
Order status	2.01	2.01	2.01
Delivery	2.01	2.01	2.02
Stock level	2.01	2.01	2.01

Table 5 - 2 Think times

Think times	Minimum	Average	Maximum
New order	0.00	12.28	119.99
Payment	0.00	12.28	120.00
Order status	0.00	10.29	99.95
Delivery	0.00	5.29	49.98
Stock level	0.00	5.30	49.85

5.4 Response time frequency distribution curves and other graphs

Response Time frequency distribution curves (see Clause 5.6.1) must be reported for each transaction type. The performance curve for response times versus throughput (see Clause 5.6.2) must be reported for the New-Order transaction. The Think Time frequency distribution curve (see Clause 5.6.3) must be reported for the New-Order transaction. A graph of throughput versus elapsed time (see Clause 5.6.5) must be reported for the New-Order transaction, and the measurement interval indicated.

Figure 3: Menu Response Time

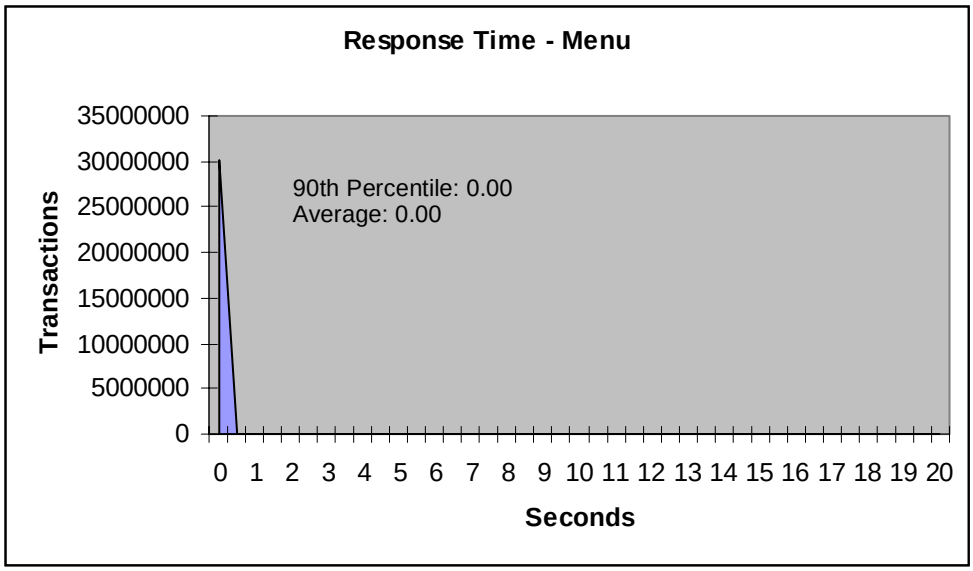


Figure 4: New Order Response Time

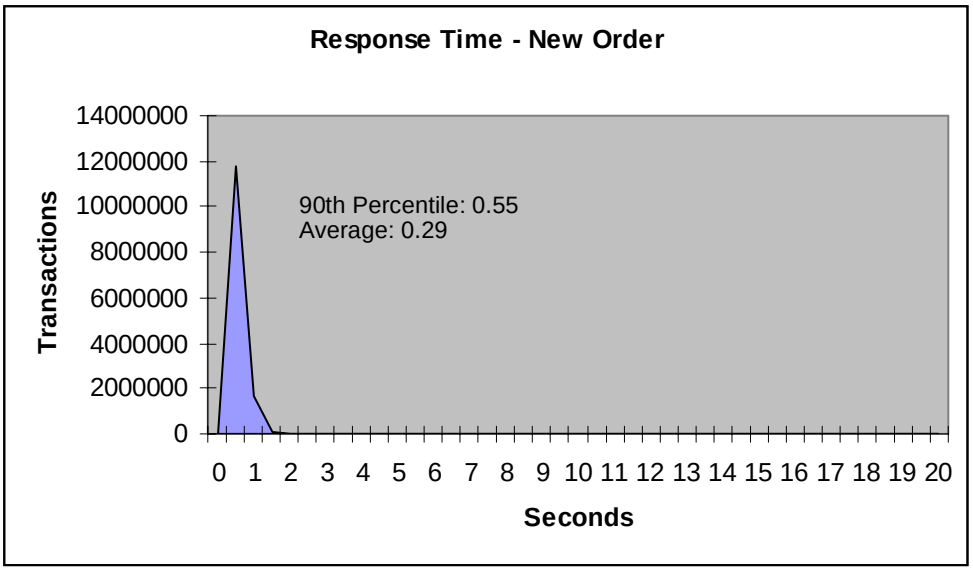


Figure 5: Delivery Response Time (Interactive)

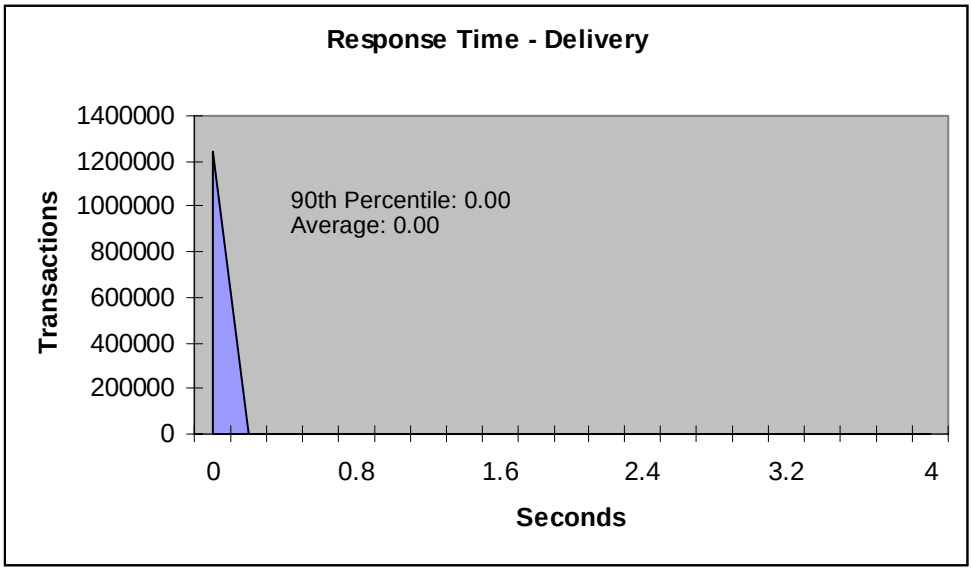


Figure 6: Order Status Response Times

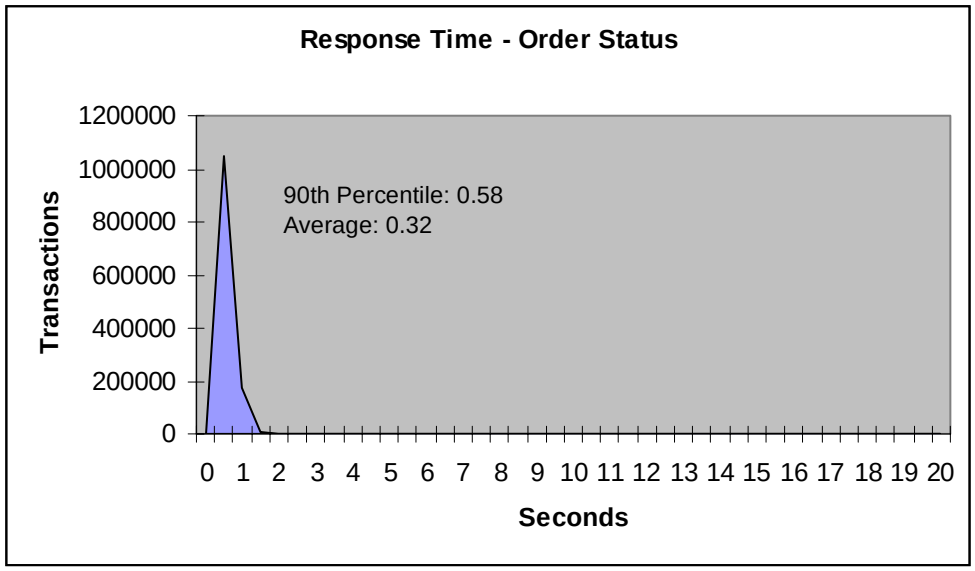


Figure 7: Payment Response Times

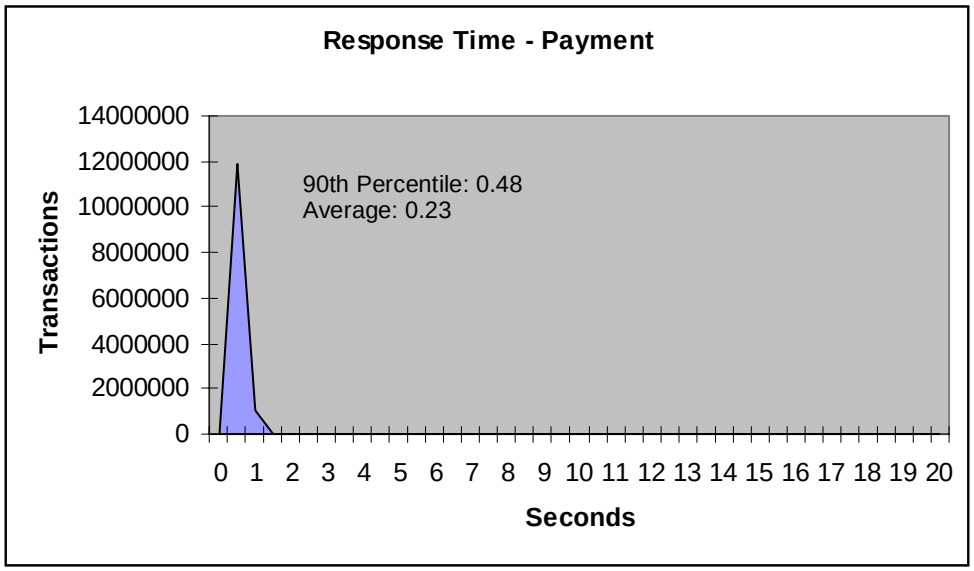


Figure 8: Stock Level Response Times

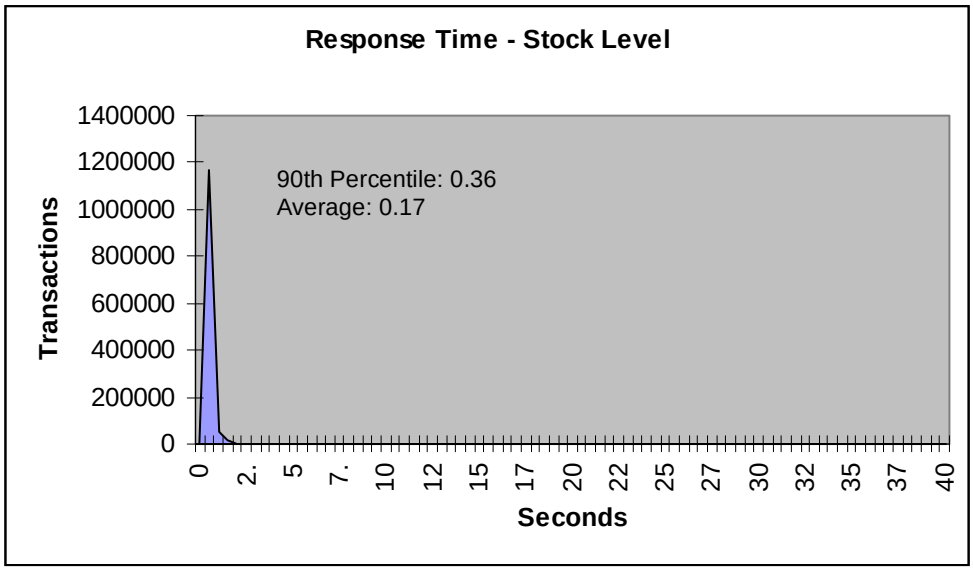


Figure 9: Response Time vs. Throughput

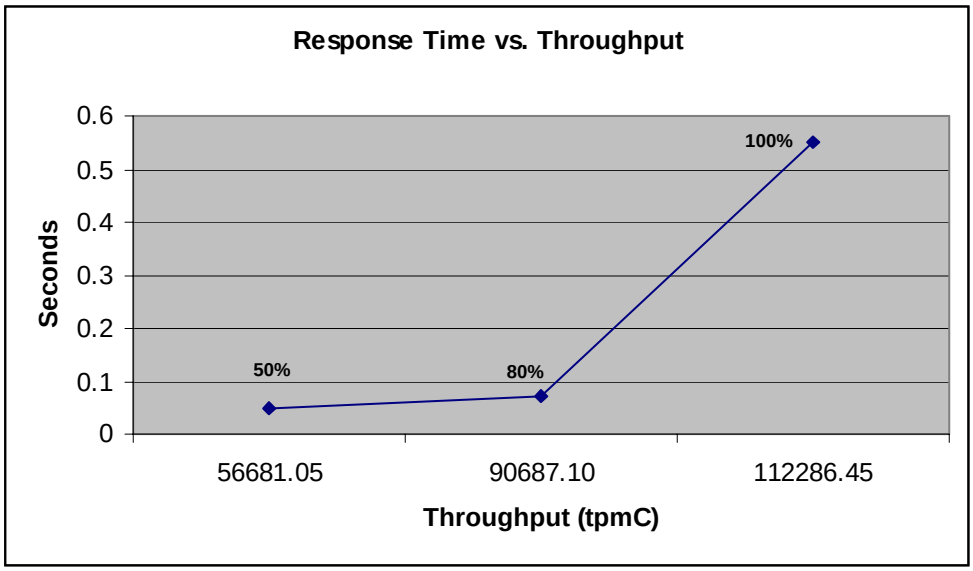


Figure 10: Throughput vs Time: New Order

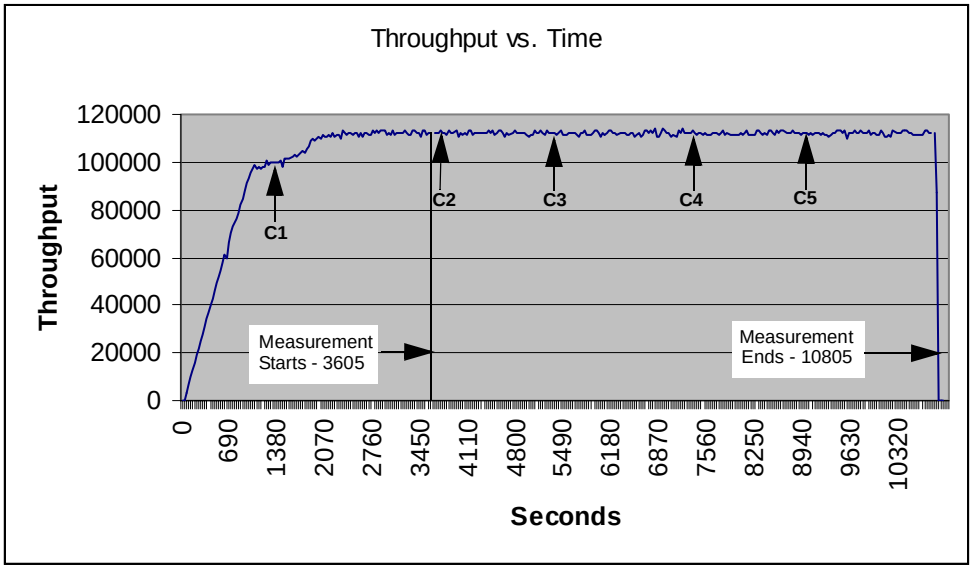
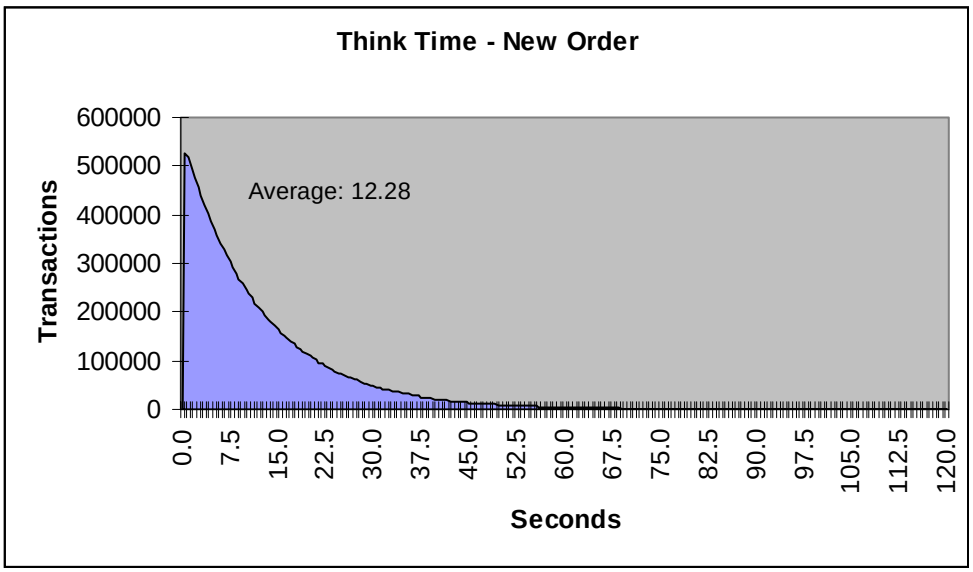


Figure 11: New Order Think Time Frequency Distribution



5.5 Steady state determination

The method used to determine that the SUT had reached a steady state prior to commencing the measurement interval must be disclosed.

Synchronization techniques employed in the benchmark process ensure that all emulated users are logged into the client and have opened the application before submitting transactions. Once all users are connected, each pauses a variable amount of time before submitting transactions. The pause times are controlled by a benchmark input parameter. The ramp-up interval is discernible during the first 20 minutes of the graph of Throughput vs. Time (). The data reduction also tracks

the user load and indicates the point in time at which all users have submitted at least one transaction. The throughput is observed to be steady within the systematic and statistical variability of the measurement after all users are submitting transactions. A checkpoint is initiated upon the end of ramp-up.

5.6 Work performed during steady state

A description of how the work normally performed during a sustained test (for example checkpointing, writing redo/undo log records, etc.), actually occurred during the measurement interval must be reported. [Clause 8.1.6.10]

Modified database buffers were migrated to disk on a least-recently-used basis independent of transaction commits. In addition, every block modification was protected by redo log records. These redo log records were written to the redo log buffer (in memory) and were flushed to a redo log file on disk either when the transaction committed or when the redo log buffer became full. However, due to the rapid commit during this benchmark, the redo log buffer was always flushed by a commit before it became full. Also, because many transactions were committing in a short period of time, a single flush of the redo log buffer resulted in many transactions' redo log data being written to disk. This is called group commit.

5.6.1 Checkpoint

During a Sybase Adaptive Server Enterprise v12.5.0.1 checkpoint, all modified blocks in the shared buffer cache which were not written to disk since the last checkpoint are written to disk.

5.6.2 Checkpoint Conditions

Sybase Adaptive Server Enterprise v12.5.0.1 performs a checkpoint for the following conditions:

1. Automatically, at an interval calculated by Sybase Adaptive Server Enterprise on the basis of system activity and the recovery interval value in the system table syscurconfigs. The recovery interval determines checkpoint frequency by specifying the amount of time it should take the system to recover.
2. Upon an explicit checkpoint request in Transact-SQL.

5.6.3 Checkpoint Implementation

For each benchmark measurement after all users are active, the script checkpoints issues a checkpoint and starts a background process, which sleeps and performs another checkpoint every 30 minutes. The recovery interval is configured large enough that no other checkpoints occur during the measurement.

5.7 Reproducibility

A description of the method used to determine the reproducibility of the measurement results.

No reproducibility run was needed in this revision of the benchmark.

5.8 Measurement period duration

A statement of the duration of the measurement interval for the reported Maximum Qualified Throughput (tpmC®) must be included.

The measurement interval was 120 minutes.

5.9 Regulation of transaction mix

The method of regulation of the transaction mix (e.g., card decks or weighted random distribution) must be described. If weighted distribution is used and the RTE adjusts the weights associated with each transaction type, the maximum adjustments to the weight from the initial value must be disclosed.

The weighted selection method of Clause 5.2.4.1 was used. The weights were adjusted during the run as permitted by Clause 5.2.4.1(2).

The following table shows initial, minimum, and maximum weights used by the mix generator.

Table 5 - 1 Weights used

Transaction type	Initial	Minimum	Maximum
Payment	43.05%	41.328%	44.772%
Order status	4.1%	3.936%	4.264%
Delivery	4.1%	3.936%	4.264%
Stock level	4.1%	3.936%	4.264%

5.10 Transaction mix

The percentage of the total mix for each transaction type must be disclosed.

Table 5 - 1 Transaction mix

Transaction type	Percentage
New order	44.67
Payment	43.02
Order status	4.09
Delivery (deferred)	4.11
Stock level	4.11

5.11 Transaction statistics

The percentage of New-Order transactions rolled back as a result of invalid item number must be disclosed. The average number of order-lines entered per New-Order transaction must be disclosed. The percentage of remote order-lines entered per New-Order transaction must be disclosed. The percentage of remote Payment transactions must be disclosed. The percentage of customer selections by customer last name in the Payment and Order-Status transactions must be disclosed. The percentage of Delivery transactions skipped due to there being fewer than necessary orders in the New-Order table must be disclosed.

See “Table 2 - 1 Transaction runtime statistics” on page 7.

5.12 Duration of checkpoints

The start time and duration in seconds of at least the four (4) longest checkpoints during the Measurement Interval must be disclosed (see Clause 5.5.2.2 (2)). [Clause 8.1.6.11]

There were four checkpoints in the measurement interval. There was one checkpoint during the rampup period. A new checkpoint happened every 30 minutes. The duration of the checkpoints in minutes were as follows:

- 29.5
- 28.8
- 29.1
- 30.2

5.13 Duration of measurement

A statement of the duration of the measurement interval for the reported Maximum Qualified Throughput (tpmC) must be included. [Clause 8.1.6.12]

The measurement interval was 120 minutes.

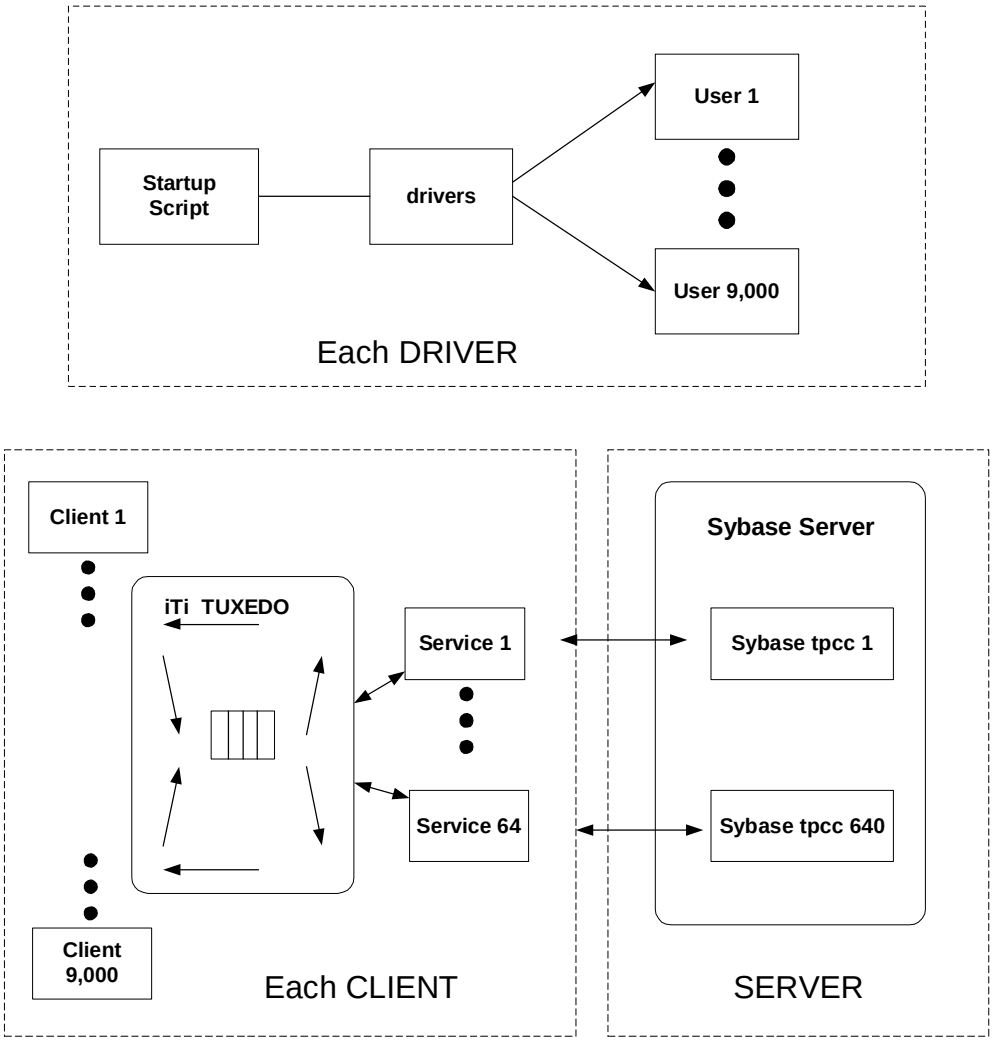
6. Clause 6 — SUT, Driver, and Communication Definition

6.1 RTE inputs

If the RTE is commercially available, then its inputs must be specified. Otherwise, a description must be supplied of what inputs (e.g., scripts) to the RTE had been used. The RTE input parameters, code fragments, functions, et cetera used to generate each transaction input field must be disclosed. Comment: The intent is to demonstrate the RTE was configured to generate transaction input data as specified in Clause 2.

The RTE (Remote Terminal Emulator) on the driver system was developed at Fujitsu-Siemens and is not commercially available. Appendix D, "Tunable Parameters and Options," on page 169 shows RTE input parameters and code fragments used to generate each transaction input field.

For this instance of the TPC-C benchmark, 10 drivers and 10 clients were used. The drivers emulated users logged in to the clients. An overview of the benchmark software on the drivers, clients, and server is shown in the following figure:



6.2 Lost connections

If the RTE is commercially available, then its inputs must be specified. Otherwise, a description must be supplied of what inputs (e.g., scripts) to the RTE had been used. [Clause 8.1.7.1]

There were no lost connections during the measurement interval.

6.3 Emulated components

It must be demonstrated that the functionality and performance of the components being emulated in the Driver System are equivalent to that of the priced system. The results of the test described in Clause 6.6.3.4 must be disclosed. [Clause 8.1.7.3]

It must be demonstrated that the functionality and performance of the components being emulated in the Driver System are equivalent to the priced system. In the benchmark configuration, the 90,000 simulated workstations connected to the clients over 10 100BT Local Area Networks (LANs) through a single Foundry Networks Fastiron switch. In the priced system, the 90,000 workstations would connect to the clients via a combination of hubs and switches, which eventually multiplexed down to 10 100BT LANs.

6.4 Networks

The network configuration of both the tested and proposed services which are being represented and a thorough explanation of exactly which parts are being replaced with the Driver System must be disclosed.

The following figures in "Configuration items" on page 1 show the network configurations of the benchmark and configured systems, and represent the Driver connected via a 100BT switch replacing the workstations and HUBs/switches:

- Figure 2. "Tested configuration" on page 2
- Figure 3. "Priced configuration" on page 3

The clients are connected via 100 Base-T to an 100BT Ethernet Cisco switch which in turn is connected via 100BT-Ethernet to the SUT.

The bandwidth of the networks used in the tested/priced configurations must be disclosed.

Ethernet and 100 Base-T LANs with a bandwidth of 100 megabits per second are used in the tested/priced configurations.

7. Clause 7 — Pricing

7.1 Hardware and software components

A detailed list of hardware and software used in the priced system must be reported. Each item must have vendor part number, description, and release/revision level, and either general availability status or committed delivery data. If package-pricing is used contents of the package must be disclosed. Pricing source(s) and effective date(s) of price(s) must also be reported.

The total 3-year price of the entire configuration must be reported including: hardware, software, and maintenance charges. Separate component pricing is recommended. The basis of all discounts used must be disclosed.

The detailed list of all hardware and software for the priced configuration is listed in the system pricing summary.

7.1.1 Hardware pricing

Network Appliance's TPC Benchmark C tests used packaged hardware systems whenever possible to simplify configurations to the fewest number of line items.

All hardware and software products have been priced to satisfy the 3 year warranty and 7 X 24 with 4 hour response requirements.

7.1.2 Software pricing

The priced system uses the following software products:

- BEA Tuxedo CFS 7.1
- Sybase ASE 12.5.0.1

7.1.3 Price discounts

See Appendix F, "Price Quotations," on page 103.

7.2 Availability status

The committed delivery date for general availability (date) of products used in the price calculations must be reported.When the priced system includes products with different availability dates, the reported availability date for the priced system must be the date at which all components are committed to be available.

All components were available at the time of the publication except for Network Appliance software version 6.2D10, which will be available in August, 2002.

7.3 Performance and Price/Performance

A statement of the measured tpmC, as well as the respective calculations for 3-year pricing and price/performance (price/tpmC) and the availability date must be included.

- Maximum Qualified Throughput — 112, 286.46
- Price per tpmC — $13.44
- Availability — 8/31/02

7.4 Country Specific Pricing

Additional Clause 7 related items may be included in the Full Disclosure Report for each country specific priced configuration.Country specific pricing is subject to Clause 7.1.7.

None.

7.5 Usage Pricing

For any usage pricing, the sponsor must disclose:

- Usage level at which the component was priced.
- A statement of the company policy allowing such pricing.

None.

8. Clause 8 — Audit

8.1 Auditor

The auditor's name, address, phone number, and a copy of the auditor's attestation letter indicating compliance must be included in the Full Disclosure Report.

A review of the pricing model is required to ensure that all components required are priced (see Clause 9.2.8). The auditor is not required to review the final Full Disclosure Report or the final pricing prior to issuing the attestation letter. [Clause 8.1.9]

The benchmark test of the PRIMEPOWER 850 system with DAFS was independently audited by Lorna Livingtree, TPC certified auditor of Performance Metrics, Inc.

The attestation letter is included in Appendix on page 193.

Requests for this TPC-C Full Disclosure Report should be sent to:

Network Appliance, Inc.
495 East Java Drive
Sunnyvale, CA 94089

Appendix A. Client source code

tpcc_client.c

```
/*****************************************
*************************
 * tpcc_client.c
 *
 * Multi-threadded client. Comes up and lis-
tens for connections from the
 * user/rte. Spawns a thread for each rte.
 *
 * To use this client,
 *     - make sure the rte is connecting to the
port we are listening on.
 *     - make sure the environment is set
before running this client
*****************************************
*************************/

static const char Usage[] = "\n\
USAGE: %s [-d] [-t threads_per_process] [-p
processes] [-c threads_per_ctx]\n\
where\n\
       -t - number of threads per process,
default is %d\n\
       -p - number of process to fork, default
is %d\n\
       -c - number of threads sharing each
tuxedo context, default is %d\n\
       -d - turn on debugging output";

#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <unistd.h>
#include <signal.h>
#include <sys/types.h>
#include <sys/time.h>
#include <sys/param.h>
#include <sys/socket.h>
#include <netdb.h>
#include <netinet/in.h>
#include <pthread.h>
#include <limits.h>
#include <sys/resource.h>
#include <errno.h>
#include <atmi.h>

#include "tpcc_client.h"
#include "tpcc_tux.h"
#include "tpcc_forms.h"
#include "tpcc_global.h"

#define DEFAULT_USER"tpcc"
#define DEFAULT_LISTEN_PORT21001
#define DEFAULT_THREADS_PER_CTX250
#define DEFAULT_THREADS_PER_PROC1000
#define DEFAULT_PROCS1
#define LISTEN_BACKLOG1000
#define TCP_PROTOCOL6

/*
 * We will fork NumProcs processes.
 * Each process will accept up to ThreadsPer-
Proc connections.
 * ThreadsPerCtx threads will share a single
Tuxedo context.
 */
static int ThreadsPerCtx =
DEFAULT_THREADS_PER_CTX;
static int ThreadsPerProc =
DEFAULT_THREADS_PER_PROC;
static int NumProcs = DEFAULT_PROCS;

/*
 * Data passed to new threads.
 */
typedef struct {
       int           connectionFd;/* socket
connection*/
       TPCONTEXT_TtuxedoCtx;/* tuxedo context
*/
} thread_arg_t ;

static pthread_t
start_client_thread(thread_arg_t *argPtr);
static int    exit_thread(thread_arg_t
*argPtr);

int
main(int argc, char *argv[])
{
       int           listenSock, connSock;
       int           firstTime = 1;
       pid_t         pid;
       int           cnt, numThreads = 0;
       pthread_tlastThreadId;
       TPCONTEXT_TtuxedoCtx;
       thread_arg_t*threadArgPtr;
       pthread_t*threadArray;/* keep thread
ids in here */

       /*
        * ID of our client process, mostly used
in Clog to tag output.
        * 0 is the parent.
        */
       ClientNum = 0;

       /*
        * Parse arguments and setup environ-
```

```
ment.
        */
       if (parse_args(argc, argv) == -1) {
              Clog("Error parsing arguments,
exiting.\n");
              exit(1);
       }

       Clog("%s starting: Debug=%d,
Threads=%d, Procs=%d, Contexts=%d\n",
              argv[0], Debug, ThreadsPerProc,
NumProcs, ThreadsPerCtx);

       /*
        * Make sure we can open a socket per
thread.
        */
       if (check_file_max() == -1) {
              Clog("check_file_max failed, try
setuid root (chmod u+s), exiting.\n");
              exit(1);
       }

       /*
        * Set up socket to listen for connec-
tions.
        */
       listenSock =
initialize_listener(DEFAULT_LISTEN_PORT);
       if (listenSock == -1) {
              Clog("Error initializing lis-
tener, exiting.\n");
              exit(1);
       }

       /*
        * Save the ids of the threads we start
so we can wait
        * for them and cleanup before exiting.
        */
       threadArray = (pthread_t *) malloc(
                           sizeof(*threadAr-
ray) * ThreadsPerProc);
       if (threadArray == NULL) {
              Clog("malloc(%d) of thread array
failed\n",
                     sizeof(*threadArray) *
ThreadsPerProc);
              exit(-1);
       }

       /*
        * Now that we have our address bound to
that socket,
        * fork the number of processes we want.
        */
       pid = 0;
       for (cnt = 0; cnt < NumProcs - 1; cnt++)
{
              pid = fork();
              if (pid == -1) {
                     Clog("fork() failed,
errno=%d\n", errno);
                     exit(1);
              }
              else if (pid != 0) {/* child */
                     ClientNum= cnt + 1;
                     Clog("Started.");
                     break;
              }
       }

       if (pid == 0) {
              Clog("Forked %d processes.\n",
cnt);
       }

       /*
        * Create a key for thread-specific
data.
        */
       if (pthread_key_create(&ThreadKey,
NULL) != 0) {
              Clog("pthread_key_create failed,
exiting.\n");
              exit(1);
       }

       /*
        * Ignore sigpipe - writing to pipe with
no reader.
        * We get this during wind down on occa-
sion.
        */
       signal(SIGPIPE, SIG_IGN);

       /*
        * Wait for a new connection.
        * Create a thread.
        *
        * We accept at most ThreadsPerProc con-
nections.
        *
        * Each set of ThreadsPerCtx gets its
own Tuxedo context.
        *
        * Call Init_Monitor_Once after the
first connection to
        * connect to the monitor and initial-
ize. We have to
```

```
        * wait for the first connection because
we may start
        * before Tuxedo starts.
        */
       while (numThreads < ThreadsPerProc) {
              connSock =
wait_for_connection(listenSock);
              if (connSock == -1) {
                     Clog("Error waiting for
connection, exiting.\n");
                     exit(1);
              }
              if (Debug) {
                     Clog("User connected,
fd=%d\n", connSock);
              }

              if (firstTime) {
                     firstTime = 0;
                      if (Init_Monitor_Once())
{

Clog("Init_Monitor_Once failed\n");
                            exit(1);
                     }
                     Clog("Init_Monitor_Once
succeeded\n");
              }

              threadArgPtr = (thread_arg_t
*)malloc(sizeof(thread_arg_t));
              if (threadArgPtr == NULL) {
                     Clog("malloc(%d) for
thread %d failed.\n",

sizeof(thread_arg_t), numThreads);
                     exit(1);
              }

              /* Create new tuxedo context
every ThreadsPerCtx threads */
              if ((numThreads % ThreadsPerCtx)
== 0) {
                     if
(Init_Monitor_Per_Ctx(&tuxedoCtx)) {

Clog("Init_Monitor_Per_Ctx failed,
numThreads=%d\n",
                            numThreads);
                        exit(1);
                     }
                     Clog("Created context
%d\n",
                            (numThreads /
ThreadsPerCtx) + 1);
              }
              threadArgPtr->tuxedoCtx = tuxe-
doCtx;
              threadArgPtr->connectionFd =
connSock;

              lastThreadId =
start_client_thread(threadArgPtr);
              if (lastThreadId == (pthread_t)-
1) {
                     Clog("Error starting cli-
ent thread, exiting.\n");
                     exit(1);
              }

              threadArray[numThreads] = last-
ThreadId;
              numThreads++;
       }

       Clog("Created all %d threads, wait-
ing\n", numThreads);

       /*
        * If we didn't get ThreadsPerProc con-
nections, then we'll
        * wait in the above loop until someone
kills us.
        * If we do hit the max, then wait here
for the threads to finish.
        * They will close their connections,
then we will call tpterm();
        */

       close(listenSock);

       for (cnt = 0; cnt < numThreads; cnt++) {
              if (pthread_join(threadAr-
ray[cnt], NULL) != 0) {
                     Clog("pthread_join(%ld)
failed, errno=\n",
                            threadArray[cnt],
errno);
              }
              if (Debug) {
                     Clog("pthread_join(%ld)
OK\n", threadArray[cnt]);
              }
              if (((cnt + 1) % 10) == 0) {
                     Clog("%d threads termi-
nated\n", cnt + 1);
              }
       }

       Clog("All %d threads terminated\n",
```

```
numThreads);

       Rundown_Monitor();

       Clog("Done\n");
       exit(0);
}

static void
usage(char *myname)
{
       fprintf(stderr, Usage, myname,
DEFAULT_THREADS_PER_PROC, DEFAULT_PROCS,
              DEFAULT_THREADS_PER_CTX);
}

/* Don't have any args yet */
int
parse_args(int argc, char *argv[])
{
       int          opt;
       extern char*optarg;

       while ((opt = getopt(argc, argv,
"dt:p:c:")) != EOF) {
              switch (opt) {
              case 'd':
                     Debug = 1;
                     break;
              case 't':
                     ThreadsPerProc =
atoi(optarg);
                     break;
              case 'p':
                     NumProcs = atoi(optarg);
                     break;
              case 'c':
                     ThreadsPerCtx =
atoi(optarg);
                     break;
              default:
                     usage(argv[0]);
                     return(-1);
              }
       }
       return(0);
}

/*
 * Set up socket to listen on listenPort for
connections from the rte.
 * Return the listen socket.
 */
int
initialize_listener(int listenPort)
{

       int                 listenSock;
       char                myHostName[256];
       struct hostent*hostEnt;
       struct sockaddr_inmyAddr;

       listenSock = socket(PF_INET,
SOCK_STREAM, TCP_PROTOCOL);
       if (listenSock == -1) {
              Clog("socket() failed,
errno=%d\n", errno);
              return(-1);
       }

       /* Listen on any interface */
       memset((char *)&myAddr, 0,
sizeof(myAddr));
       myAddr.sin_addr.s_addr = INADDR_ANY;
       myAddr.sin_family = AF_INET;
       myAddr.sin_port = htons(listenPort);

       if (bind(listenSock, (struct sockaddr
*)&myAddr, sizeof(myAddr)) == -1){
              Clog("bind() failed,
errno=%d\n", errno);
              return(-1);
       }

       if (listen(listenSock, LISTEN_BACKLOG)
== -1) {
              Clog("listen() failed,
errno=%d\n", errno);
              return(-1);
       }

       Clog("Listening at hostaddr %#lx, port
%ld\n",
              (long)myAddr.sin_addr.s_addr,
listenPort);

       return(listenSock);
}

/*
 * Wait for the next connection request and
accept it.
 * Return the new socket.
 */

int wait_for_connection(int listenSock) {
       int    connectedSock ;

       connectedSock = accept(listenSock,
NULL, NULL);
```

```
	if (connectedSock == -1) {
		Clog("accept() failed,
errno=%d\n", errno);
	}
	return(connectedSock);
}


/*
 * This is where the client thread starts - it
does all the client processing.
 * After the main thread receives a new connec-
tion, it creates a thread that
 * runs this routine.
 *
 * arg is our thread_arg_t struct that was mal-
loced by our parent.
 */
void *
client_thread_main(void *arg)
{
	extern voiddo_transaction(int);
	int		menu_selection;
	thread_arg_t*argPtr = (thread_arg_t
*)arg;

	/*
	 * Initialize our per-thread data.
	 */
	if (initialize_thread_data(argPtr) == -
1) {
		Clog("initialize_thread_data()
failed\n");
		exit_thread(argPtr);
		/** NOTREACHED **/
	}

	if (Debug) {
		Clog("New thread initial-
ized\n");
	}

	/*
	 * Initialize tuxedo for this thread -
allocate buffers and set
	 * our tuexdo context.
	 */
	if (Init_Monitor_Per_Thread(&argPtr-
>tuxedoCtx)) {
		Clog("Init_Monitor_Per_Thread
failed\n");
		exit_thread(argPtr);
		/** NOTREACHED **/
	}

	/*
	 * Conduct the login dialog with the
user or rte.
	 */
	if (tpcc_login_user(argPtr->connec-
tionFd, argPtr->connectionFd,
			DEFAULT_USER) == -1) {

		Clog("tpcc_login_user() failed,
errno=%d\n", errno);

		exit_thread(argPtr);
		/** NOTREACHED **/
	}

	get_wd();
	set_display();

	Send_Menu();

	if (Debug) {
		Clog("Logged in and running.\n");
	}

	while ((menu_selection = sel_trans())
!= 9) {
		if ((menu_selection < 0) ||
(menu_selection > 4))
			continue;
		do_transaction(menu_selection);
		Send_Menu();
	}

	close(argPtr->connectionFd);

	exit_thread(argPtr);
	/** NOTREACHED **/
}

/*
 * Clean up somewhat and exit thread.
 * Closing the socket is important in the case
of a failed login attempt -
 * the rte will disconnect and retry.
 */
static int
exit_thread(thread_arg_t *argPtr)
{
	close(argPtr->connectionFd);
	Clog("Thread exiting.\n");
	pthread_exit(NULL);
}

/*
 * Create a new thread and start it with arg as
```

```
its arg.
 * Return the thread id.
 */
static pthread_t
start_client_thread(thread_arg_t *arg)
{
       int          rc;
       pthread_tthread;

       rc = pthread_create(&thread, NULL,
client_thread_main,
              (void *)arg);

       if (rc != 0) {
              Clog("pthread_create() failed,
errno=%d\n", errno);
              return(-1);
       }
       return(thread);
}


/*
 * Initialize the current thread - set up the
per thread data.
 * This must be called by a thread before it
can access any of its
 * global data in global_data_t!
 */
int
initialize_thread_data(thread_arg_t *argPtr)
{
       global_data_t*myGlobalPtr;

       myGlobalPtr = (global_data_t *)mal-
loc(sizeof(global_data_t));
       if (myGlobalPtr == NULL) {
              Clog("malloc(%d) for global data
failed, errno=%d",
                     sizeof(global_data_t),
errno);
              return(-1);
       }
       memset(myGlobalPtr, 0,
sizeof(global_data_t));

       if (pthread_setspecific(ThreadKey,
myGlobalPtr) != 0) {
              Clog("pthread_setspecific
failed, errno=%d\n", errno);
              return(-1);
       }

       tpcc_forms_init(myGlobalPtr);

       myGlobalPtr->g_tty_in = argPtr->connec-
tionFd;
       myGlobalPtr->g_tty_out = argPtr->con-
nectionFd;
       myGlobalPtr->g_my_thread_id =
pthread_self();

       return(0);
}

get_wd()
{
       int num = 5 ;

       setup_wd();
       display_screen(num);
       get_inputs(num);
}

void
do_transaction(int num)
{
       int             status;

       display_screen(num);
       status = get_inputs(num);
       if (status == 3)
              return;

       if (Snd_Txn_To_Monitor(num))
              return;

       display_output(num);
}

/*
 * Check that we can open enough sockets.
 * If not, try raising the rlimit of per pro-
cess open fds.
 * (Note we must be running as root to do
that.)
 */
int
check_file_max()
{
/*
 * John Scott tried to rebuild the kernel with
this max increased
 * but it had problems coming up to init 3.
 * So he hacked it so anyone can raise the
limit.
 * - buddy 8/23/01
 */
       struct rlimit rlim;
```

```
	if (getrlimit(RLIMIT_NOFILE, &rlim) ==
-1) {
		Clog("getrlimit failed,
errno=%d\n", errno);
	}
	if (rlim.rlim_cur >= (ThreadsPerProc +
5)) {
		return(0);/* OK */
	}

	rlim.rlim_max = rlim.rlim_cur = Thread-
sPerProc + 16;

	if (setrlimit(RLIMIT_NOFILE, &rlim) ==
-1) {
		perror("setrlimit failed");
		Clog("Setting max open files to
%d failed, errno=%d\n", errno);
		Clog("You must run as root to
increase max open files. Try chown root tpcc;
chmod +us tpcc.\n");
		return(-1);
	}

	Clog("Increased max fds to %d\n",
rlim.rlim_cur);

	return(0);
}
```

tpcc_client.h

```
/********** tpcc_client.h ********/

#include <time.h>
#include <sys/types.h>
#include <time.h>

#define BOOLEAN int
#define LINEMAX 256
#define FALSE 0
#define TRUE 1
#define NEWORDER 0
#define PAYMENT 1
#define ORDSTAT 2
#define DELIVERY 3
#define STOCKLEV 4
#define WD 5
#define MAX_OL 15
#define TPM_ERROR 1

/*
 * Data structures of input data for each
transaction type
 */

/*
 * Data structures descriptions for IO data
for each transaction type
 */

struct no_itm_struct {
	int	ol_supply_w_id;
	int	ol_i_id;
	char	i_name[25];
	int	ol_quantity;
	int	s_quantity;
	char	brand[2];
	double i_price;
	double ol_amount;
};
struct no_struct {
	int	w_id;
	int	d_id;
	int	c_id;
	int	o_id;
	int	o_ol_cnt;
	double c_discount;
	double w_tax;
	double d_tax;
	char	o_entry_d[20];
	char	c_credit[3];
	char	c_last[17];
	struct no_itm_struct o_ol[15];
	char	 status[25];
	double total;
};
struct pay_struct {
	int	w_id;
	int	d_id;
	int	c_id;
	int	c_w_id;
	int	c_d_id;
	double h_amount;
	double c_credit_lim;
	double c_balance;
	double c_discount;
	char	h_date[20];
	char	w_street_1[21];
	char	w_street_2[21];
	char	w_city[21];
	char	w_state[3];
	char	w_zip[11];
	char	d_street_1[21];
	char	d_street_2[21];
	char	d_city[21];
	char	d_state[3];
	char	d_zip[11];
	char	c_first[17];
```

```
        char   c_middle[3];
        char   c_last[17];
        char   c_street_1[21];
        char   c_street_2[21];
        char   c_city[21];
        char   c_state[3];
        char   c_zip[11];
        char   c_phone[17];
        char   c_since[11];
        char   c_credit[3];
        char   c_data_1[51];
        char   c_data_2[51];
        char   c_data_3[51];
        char   c_data_4[51];
};

struct ord_itm_struct {
        int    ol_supply_w_id;
        int    ol_i_id;
        int    ol_quantity;
        double ol_amount;
        char   ol_delivery_d[11];
};

struct ord_struct {
        int    ol_cnt;
        int    w_id;
        int    d_id;
        int    c_id;
        int    o_id;
        int    o_carrier_id;
        double c_balance;
        char   c_first[17];
        char   c_middle[3];
        char   c_last[17];
        char   o_entry_d[20];
        struct ord_itm_struct s_ol[MAX_OL];
};

struct del_struct {
        int    w_id;
        int    o_carrier_id;
        time_t queue_time;
};

struct stock_struct {
        int    w_id;
        int    d_id;
        int    threshold;
        int    low_stock;
};

struct menu_struct {
        int    w_id;
        int    d_id;
};

};

/*
 * Data structure for input & output data
 */
typedef union info {
        struct no_struct neworder;
        struct pay_struct payment;
        struct ord_struct ordstat;
        struct del_struct delivery;
        struct stock_struct stocklev;
        struct menu_struct wd;
} info_t;

struct io_tpcc {
        int    type;
        info_t info;
};
```

tpcc_forms.c

```
#include <stdio.h>
#include <termio.h>
#include <stdlib.h>
#include <assert.h>
#include <sys/time.h>
#include <sys/times.h>/* buddy */
#include <pthread.h>
#include <time.h>
#include “tpcc_client.h”
#include “tpcc_forms.h”
#include “tpcc_tux.h”
#include “tpcc_global.h”

extern void Clog(char *, ...);
extern void SCREENlog(int, char *);

const char blanks[1802] = “
“;
        /* All the possible messages to print
out */
static const char MANDATORY_MSG[] =
    “\033[24;1H\033[mMandatory data field!
Please enter data.”;
static const char INVALID_MSG[] =
    “\007\033[24;1HAn invalid character was
entered. Please enter again.”;
static const char ERASE_MSG[] =
“\033[24;1H\033[K\033[4m”;
static const char MIN1DIGIT_MSG[] =
    “\033[24;1H\033[mYou must enter atleast 1
digit. Please reenter.\033[4m\1”;
static const char BAD_INPUTS[] =
   “ #### Bad input data was entered -- Select
```

```
again #### \1";
static const char INCOMPLINE_MSG[] =
    "\033[24;1H\033[mOrder line is incomplete.
Please complete the whole line.\033[4m\1";
static const char ID_OR_LAST_MSG[] =
    "\033[24;1H\033[mYou must enter either the
Last Name or the Customer Number.\033[4m\1";
static const char EXC_MAX_LFT_DEC_DGT_MSG[] =
    "\033[24;1H\033[mMaximum digits left of
decimal point already entered. '.'
expected\033[4m\1";
static const char EXC_FLD_LIM_MSG[] =
"\007\033[24;1H\033[mMaximum digits already
entered. Tab or <CR> expected\033[4m\1";/* jr
*/
static const char EXECUTION_STATUS_MSG[] =
    "\033[m\033[22;18HBad Input Data";
static const char DELIVERY_QUEUED_MSG[] =
    "\033[m\033[6;19HDelivery has been
queued";
static const char menu_buf[] =
    "\033[H\033[J\033[mNew-Order(n)  Pay-
ment(p)  Order-Status(o)  Delivery(d)  Stock-
Level(s)  Exit(e)";


static const io_elem
neworder_inputs_init_value[] = {
    /* y x len flags ptr to data ptr to read
function */
    /* - - --- ----- ----------- -------------
------ */
    2, 29, 2, 0, 0, &read_integer,
    3, 12, 4, 0, 0, &read_integer,
    7, 3, 4, 0, 0, &read_integer,
    7, 10, 6, 0, 0, &read_integer,
    7, 45, 2, 0, 0, &read_integer,
    8, 3, 4, 0, 0, &read_integer,
    8, 10, 6, 0, 0, &read_integer,

    8, 45, 2, 0, 0, &read_integer,
    9, 3, 4, 0, 0, &read_integer,
    9, 10, 6, 0, 0, &read_integer,
    9, 45, 2, 0, 0, &read_integer,
    10, 3, 4, 0, 0, &read_integer,
    10, 10, 6, 0, 0, &read_integer,
    10, 45, 2, 0, 0, &read_integer,
    11, 3, 4, 0, 0, &read_integer,
    11, 10, 6, 0, 0, &read_integer,
    11, 45, 2, 0, 0, &read_integer,
    12, 3, 4, 0, 0, &read_integer,
    12, 10, 6, 0, 0, &read_integer,
    12, 45, 2, 0, 0, &read_integer,
    13, 3, 4, 0, 0, &read_integer,
    13, 10, 6, 0, 0, &read_integer,
    13, 45, 2, 0, 0, &read_integer,
    14, 3, 4, 0, 0, &read_integer,
    14, 10, 6, 0, 0, &read_integer,
    14, 45, 2, 0, 0, &read_integer,
    15, 3, 4, 0, 0, &read_integer,
    15, 10, 6, 0, 0, &read_integer,
    15, 45, 2, 0, 0, &read_integer,
    16, 3, 4, 0, 0, &read_integer,
    16, 10, 6, 0, 0, &read_integer,
    16, 45, 2, 0, 0, &read_integer,
    17, 3, 4, 0, 0, &read_integer,
    17, 10, 6, 0, 0, &read_integer,
    17, 45, 2, 0, 0, &read_integer,
    18, 3, 4, 0, 0, &read_integer,
    18, 10, 6, 0, 0, &read_integer,
    18, 45, 2, 0, 0, &read_integer,
    19, 3, 4, 0, 0, &read_integer,
    19, 10, 6, 0, 0, &read_integer,
    19, 45, 2, 0, 0, &read_integer,
    20, 3, 4, 0, 0, &read_integer,
    20, 10, 6, 0, 0, &read_integer,
    20, 45, 2, 0, 0, &read_integer,
    21, 3, 4, 0, 0, &read_integer,
    21, 10, 6, 0, 0, &read_integer,
    21, 45, 2, 0, 0, &read_integer,
    999, 0, 0, 0, 0, NULL
};

static const io_elem
payment_inputs_init_value[] = {
/* y x len flags ptr to data ptr to read func-
tion */
/* - - --- ----- ----------- -----------------
-- */
    4, 52, 2, 0, 0, &read_integer,
    9, 11, 4, 0, 0, &read_integer,
    9, 33, 4, 0, 0, &read_integer,
    9, 54, 2, 0, 0, &read_integer,
    10, 29, 16, 0, 0, &read_string,
    15, 24, 7, 0, 0, &read_integer,
    999, 0, 0, 0, 0, NULL
};
static const io_elem
ordstat_inputs_init_value[] = {
/* y x len flags ptr to data ptr to read func-
tion */
/* - - --- ----- ----------- -----------------
-- */
    2, 29, 2, 0, 0, &read_integer,
    3, 11, 4, 0, 0, &read_integer,
    3, 44, 16, 0, 0, &read_string,
    999, 0, 0, 0, 0, NULL
};
static const io_elem
delivery_inputs_init_value[] = {
```

```
/* y x len flags ptr to data ptr to read func-
tion */
/* - - --- ----- ----------- -----------------
-- */
    4, 17, 2, 0, 0, &read_integer,
    999, 0, 0, 0, 0, NULL
};
static const io_elem
stocklev_inputs_init_value[] = {
/* y x len flags ptr to data ptr to read func-
tion */
/* - - --- ----- ----------- -----------------
-- */
    4, 24, 2, 0, 0, &read_integer,
    999, 0, 0, 0, 0, NULL
};
static const io_elem wd_inputs_init_value[] =
{
/* y x len flags ptr to data ptr to read func-
tion */
/* - - --- ----- ----------- -----------------
-- */
    2, 16, 4, 0, 0, &read_integer,
    2, 43, 4, 0, 0, &read_integer,
    999, 0, 0, 0, 0, NULL
};

static const text_elem NO_text_elem[] = {
    1, 36, “New Order”,
    2, 1, “Warehouse:”,
    2, 19, “District:”,
    2, 55, “Date:”,
    3, 1, “Customer:”,
    3, 19, “Name:”,
    3, 44, “Credit:”,
    3, 57, “%Disc:”,
    4, 1, “Order Number:”,
    4, 25, “Number of Lines:”,
    4, 52, “W_tax:”,
    4, 67, “D_tax:”,
    6, 2, “Supp_W Item_Id Item Name”,
    6, 45, “Qty Stock B/G Price Amount”,
    22, 1, “Execution Status:”,
    22, 62, “Total:”,
    0, 0, NULL
};
static const text_elem PT_text_elem[] = {
    1, 38, “Payment”,
    2, 1, “Date:”,
    4, 1, “Warehouse:”,
    4, 42, “District:”,
    9, 1, “Customer:”,
    9, 17, “Cust-Warehouse:”,
    9, 39, “Cust-District:”,
    10, 1, “Name:”,
    10, 50, “Since:”,
    11, 50, “Credit:”,
    12, 50, “%Disc:”,
    13, 50, “Phone:”,
    15, 1, “Amount Paid:”,
    15, 23, “$”,
    15, 37, “New Cust-Balance:”,
    16, 1, “Credit Limit:”,
    18, 1, “Cust-Data:”,
    0, 0, NULL
};
static const text_elem OS_text_elem[] = {
    1, 35, “Order-Status”,
    2, 1, “Warehouse:”,
    2, 19, “District:”,
    3, 1, “Customer:”,
    3, 18, “Name:”,
    4, 1, “Cust-Balance:”,
    6, 1, “Order-Number:”,
    6, 26, “Entry-Date:”,
    6, 60, “Carrier_Number:”,
    7, 1, “Supply-W”,
    7, 14, “Item-Id”,
    7, 25, “Qty”,
    7, 33, “Amount”,
    7, 45, “Delivery-Date”,
    0, 0, NULL
};
static const text_elem DY_text_elem[] = {
    1, 38, “Delivery”,
    2, 1, “Warehouse:”,
    4, 1, “Carrier Number:”,
    6, 1, “Execution Status:”,
    0, 0, NULL
};
static const text_elem SL_text_elem[] = {
    1, 38, “Stock-Level”,
    2, 1, “Warehouse:”,
    2, 19, “District:”,
    4, 1, “Stock Level Threshold:”,
    6, 1, “low stock:”,
    0, 0, NULL
};
static const text_elem WD_text_elem[] = {
    2, 1, “Warehouse:”,
    2, 26, “District:”,
    0, 0, NULL
};

static const struct form_info
Forms_init_value[MAX_FORMS] = {
    {NO_text_elem, 0, 0, NULL, 0},
    {PT_text_elem, 0, 0, NULL, 0},
    {OS_text_elem, 0, 0, NULL, 0},
    {DY_text_elem, 0, 0, NULL, 0},
```

```
    {SL_text_elem, 0, 0, NULL, 0},
    {WD_text_elem, 0, 0, NULL, 0}
};

/*
 * Initialize variables.
 * Added when data moved into global struct for
multi-threadding.
 */
void tpcc_forms_init(global_data_t * my)
{
    my->g_OVERFLOW = FALSE;
    my->g_payment_input = FALSE;
    my->g_curbuf_consumed = 0;
    my->g_curbuf_read = 0;
    my->g_read_count = 0;

    memcpy(my->g_neworder_inputs,
neworder_inputs_init_value,
          sizeof(my->g_neworder_inputs));
    memcpy(my->g_payment_inputs,
payment_inputs_init_value,
          sizeof(my->g_payment_inputs));
    memcpy(my->g_ordstat_inputs,
ordstat_inputs_init_value,
          sizeof(my->g_ordstat_inputs));
    memcpy(my->g_delivery_inputs,
delivery_inputs_init_value,
          sizeof(my->g_delivery_inputs));
    memcpy(my->g_stocklev_inputs,
stocklev_inputs_init_value,
          sizeof(my->g_stocklev_inputs));
    memcpy(my->g_wd_inputs,
wd_inputs_init_value, sizeof(my-
>g_wd_inputs));

    memcpy(my->g_Forms, Forms_init_value,
sizeof(my->g_Forms));

    my->g_Forms[0].input_elems = my-
>g_neworder_inputs;
    my->g_Forms[1].input_elems = my-
>g_payment_inputs;
    my->g_Forms[2].input_elems = my-
>g_ordstat_inputs;
    my->g_Forms[3].input_elems = my-
>g_delivery_inputs;
    my->g_Forms[4].input_elems = my-
>g_stocklev_inputs;
    my->g_Forms[5].input_elems = my-
>g_wd_inputs;
}

void setraw()
{
       /** put screen in raw mode **/

    struct termio tbuf;
    int status;
    if (ioctl(tty_in, TCGETA, &tbuf) == -1)
       return;
    tbufsave = tbuf;
    tbuf.c_iflag &= ~(INLCR | ICRNL | IUCLC |
ISTRIP | IXON | BRKINT);
    tbuf.c_oflag &= ~OPOST;
    tbuf.c_lflag &= ~(ICANON | ISIG | ECHO);
    tbuf.c_cc[VMIN] = LEAVE_SCREEN_MIN;
    tbuf.c_cc[VTIME] = LEAVE_SCREEN_TIMEOUT;

    if (ioctl(tty_out, TCSETAF, &tbuf) == -2)
       syserr("ioctl_ERROR#2 - setting raw
mode for STDIN error");
}

/** restore terminal flags **/
void restore_terminal()
{
    struct termio tbuf;
    int status;

    if (ioctl(tty_out, TCSETAF, &tbufsave) == -
1) {
       Clog("ioctl_ERROR#3 - restoring origi-
nal input terminal settings\n");
    }

    tbuf = tbufsave;
    if (ioctl(tty_out, TCSETAF, &tbuf) == -1) {
       Clog("ioctl_ERROR#4 - Forcing the orig-
inal settings back for STDIN\n");
    }
}

/*
 * Read one character from the terminal.
 * If it is a valid menu selection, return it.
 * Otherwise return junk.
 *
 * MODIFIED SWD 4/25/01:
 *     Cleaned up.
 *     Modified to loop until a valid char is
input.
 *     Modified to return the transaction
type, not (txn_type+1) (!!)
 */
int sel_trans()
{
    int sel;
    char c;
    global_data_t *G;
```

```
    G = MY_GLOBAL_DATA;/* so we can debug core
dump */
    assert(G != NULL);

    for (;;) {
       if (read(G->g_tty_in, &c, 1) != 1) {
           Clog("sel_trans: TTY lost connec-
tion\n");
           return 9;
       }

       switch (c) {
       case 'n':
           return NEWORDER;
       case 'p':
           return PAYMENT;
       case 'o':
           return ORDSTAT;
       case 'd':
           return DELIVERY;
       case 's':
           return STOCKLEV;
       case 'e':
       case QUIT:
           return 9;
       }
    }
    /* NOT REACHED */
}

/*
 * INPUT SECTION
 */

/* Validation Function Table */
int newo_val(int *);
int paym_val(int *);
int ords_val(int *);
int del_val(int *);
int stock_val(int *);
int wd_val(int *);

int (*p_check_function[]) () = {
&newo_val, &paym_val, &ords_val, &del_val,
&stock_val, &wd_val};

/*
 * Get the inputs.
 * Validate using one of the above functions.
 * Loop until a valid transaction has been
entered.
 */
int get_inputs(int txn_type)
{
    int done = FALSE;
    int i, returned_key;
    io_elem *ioptr;
    int last_input;
    float float_h_amount = 0.0;

    memset(tuxibuf, '\0', sizeof(info_t));
    int_h_amount = 0;

    last_input =
Forms[txn_type].num_input_elems - 1;
    i = 0;
    while (done == FALSE) {
       ioptr =
&Forms[txn_type].input_elems[i];

       if (txn_type == PAYMENT) {
           if (i == 5)
              payment_input = TRUE;
           else
              payment_input = FALSE;
       }
       returned_key =
           (ioptr->fptr) (ioptr->x, ioptr->y,
ioptr->len,
                          ioptr->flags, ioptr-
>dptr);


       switch (returned_key) {
       case BACKTAB:
           if (i == 0)
              i = last_input;
           else
              i--;
           break;
       case TAB:
           if (i == last_input)
              i = 0;
           else
              i++;
           break;
       case QUIT:
           done = TRUE;
           break;
       case SUBMIT:
       case LF:
           if (screen_bufindex) {
              PAINTSCRLEN(screen_buf,
screen_bufindex);
              screen_bufindex = 0;
           }
           payment_input = FALSE;
          done = (p_check_function[txn_type])
(&i);
```

```
            break;
        }
    }
    return returned_key;
}

int newo_val(int *pos)
{
    int done = FALSE;
    struct no_itm_struct *ol_ptr, *ol_ptr2;
    int blank_line = 0, i, j;
    int blank_array[MAX_OL];
    char *bufp;
    iNO->w_id = global_w_id;

    for (i = 0; i < MAX_OL; i++)
       blank_array[i] = 0;
    if (iNO->d_id <= 0) {
       *pos = 0;
       message = TRUE;
       PAINTSCR(MANDATORY_MSG);



    } else if (iNO->c_id <= 0) {
       *pos = 1;
       message = TRUE;
       PAINTSCR(MANDATORY_MSG);

    } else {

       ol_ptr = iNO->o_ol;

       for (i = 0; i < MAX_OL; i++, ol_ptr++) {
           if (ol_ptr->ol_i_id || ol_ptr-
>ol_supply_w_id
               || ol_ptr->ol_quantity) {
               /* and is that data complete */
               if (ol_ptr->ol_i_id
                   && ol_ptr->ol_supply_w_id &&
ol_ptr->ol_quantity) {
                   iNO->o_ol_cnt++;
                   if (blank_line != 0) {
                      ol_ptr2 = iNO->o_ol;
                      for (j = 0; j < i; j++) {
                          if (blank_array[j]) {
                             blank_array[j] = 0;
                             break;
                          }
                          ol_ptr2++;
                      }
                      ol_ptr2->ol_i_id = ol_ptr-
>ol_i_id;
                      ol_ptr2->ol_supply_w_id =
ol_ptr->ol_supply_w_id;
                      ol_ptr2->ol_quantity =
ol_ptr->ol_quantity;
                      ol_ptr->ol_i_id = 0;
                      ol_ptr->ol_supply_w_id =
0;
                      ol_ptr->ol_quantity = 0;
                      blank_array[i] = 1;
                      bufp = output_screen;
                      bufp +=
                          DISPLAY_INT
                          (bufp, 5, 3,
                           j + FIRST_OL_ROW,
ol_ptr2->ol_supply_w_id);
                      bufp +=
                          DISPLAY_INT
                          (bufp, 5, 11, j +
FIRST_OL_ROW,
                           ol_ptr2->ol_i_id);
                      bufp +=
                          DISPLAY_INT(bufp, 2,
45, j + FIRST_OL_ROW,
                                      ol_ptr2-
>ol_quantity);
                      bufp += DISPLAY(bufp, 3, i
+ FIRST_OL_ROW, “    “);
                      bufp +=
                          DISPLAY(bufp, 11, i +
FIRST_OL_ROW, “     “);
                      bufp += DISPLAY(bufp, 45,
i + FIRST_OL_ROW, “  “);
                      *bufp++ = ‘\0’;
                      PAINT-
SCRLEN(output_screen, bufp - output_screen);
                   }
               } else {
                   *pos = 2 + 3 * i;
                   PAINTSCR(INCOMPLINE_MSG);
                   message = TRUE;
                   iNO->o_ol_cnt = 0;
                   return FALSE;
               }

           } else {
              blank_line++;
              blank_array[i] = 1;
           }
       }

       if (!iNO->o_ol_cnt) {
           *pos = 2;
           PAINTSCR(MANDATORY_MSG);
           message = TRUE;
           iNO->o_ol_cnt = 0;
           return FALSE;
       }
```

```
       done = TRUE;
    }
    return done;
}

int paym_val(int *pos)
{
    int done = FALSE;
    iPT->w_id = global_w_id;
    if (iPT->d_id <= 0) {
       *pos = 0;
       message = TRUE;
       PAINTSCR(MANDATORY_MSG);
    } else if (iPT->c_w_id <= 0) {
       *pos = 2;
       message = TRUE;
       PAINTSCR(MANDATORY_MSG);
    } else if (iPT->c_d_id <= 0) {
       *pos = 3;
       message = TRUE;
       PAINTSCR(MANDATORY_MSG);
    } else if (int_h_amount <= 0) {
       *pos = 5;
       message = TRUE;
       PAINTSCR(MANDATORY_MSG);
    } else if (iPT->c_id <= 0) {
       if (iPT->c_last[0] == '\0') {
           message = TRUE;
           PAINTSCR(ID_OR_LAST_MSG);
           *pos = 1;
       } else {
           done = TRUE;
       }
    } else
       done = TRUE;
    iPT->h_amount = ((float) int_h_amount) /
100.0;
    return done;
}

int ords_val(int *pos)
{
    int done = FALSE;
    iOS->w_id = global_w_id;
    if (iOS->d_id <= 0) {
       *pos = 0;


       message = TRUE;
       PAINTSCR(MANDATORY_MSG);
    } else if (iOS->c_id <= 0) {
       if (iOS->c_last[0] == '\0') {
           message = TRUE;
           PAINTSCR(ID_OR_LAST_MSG);
           *pos = 1;
       } else {
           done = TRUE;
       }
    } else
       done = TRUE;
    return done;
}

int del_val(int *pos)
{
    int done = FALSE;
    iDY->w_id = global_w_id;
    if (iDY->o_carrier_id <= 0) {
       message = TRUE;
       PAINTSCR(MANDATORY_MSG);
    } else {
       struct tms tmsbuf;
       iDY->queue_time = times(&tmsbuf);
       done = TRUE;
    }
    return done;
}

int stock_val(int *pos)
{
    int done = FALSE;
    iSL->w_id = global_w_id;
    iSL->d_id = global_d_id;
    if (iSL->threshold <= 0) {
       message = TRUE;
       PAINTSCR(MANDATORY_MSG);
    } else
       done = TRUE;
    return done;
}

int wd_val(int *pos)
{
    int done = FALSE;

    if (iWD->w_id == 0 || iWD->d_id == 0) {
       message = TRUE;
       PAINTSCR(MANDATORY_MSG);
    } else {
       global_w_id = iWD->w_id;
       global_d_id = iWD->d_id;
       done = TRUE;
    }
    return done;
}

void setup_wd()
{
    io_elem *p;
    char buf[128];
```

```
    void setup_io_elems();
    setraw();
    setup_screen_buffer(&Forms[5], 5);

    p = Forms[WD].input_elems;
    p++->dptr = &iWD->w_id;
    p++->dptr = &iWD->d_id;

   /*SBL:I think this may actually be sneaking
a read? */
    if (Debug) {
       Clog("SBL:in setup_wd where w_id=%d and
d_id=%d\n",
           iWD->w_id, iWD->d_id);
    }

    CLRSCN(buf);
    PAINTSCR(buf);
}

void set_display()
{
    int i;
    char buf[128];
    void setup_io_elems();
    for (i = 0; i < MAX_FORMS; i++)
       setup_screen_buffer(&Forms[i], i);

    setup_io_elems();

    CLRSCN(buf);
    PAINTSCR(buf);
}

void display_screen(int screen_num)
{
    if (PAINT-
SCRLEN(Forms[screen_num].blank_form,

Forms[screen_num].blank_formlen) == -1)
       syserr("Can't write out form");
}

void Send_Menu()
{
    if (PAINTSCRLEN(menu_buf, menu_buflen) ==
-1) {
       syserr("Can't send menu");
    }
}

void setup_io_elems()
{
    io_elem *p;
    int i;

    p = Forms[NEWORDER].input_elems;
    p++->dptr = &iNO->d_id;
    p++->dptr = &iNO->c_id;
    for (i = 0; i < 15; i++) {
       p++->dptr = &iNO-
>o_ol[i].ol_supply_w_id;
       p++->dptr = &iNO->o_ol[i].ol_i_id;
       p++->dptr = &iNO->o_ol[i].ol_quantity;
    }
    p = Forms[PAYMENT].input_elems;
    p++->dptr = &iPT->d_id;
    p++->dptr = &iPT->c_id;


    p++->dptr = &iPT->c_w_id;
    p++->dptr = &iPT->c_d_id;
    p++->dptr = (int *) &iPT->c_last[0];
    p->dptr = &int_h_amount;
    p = Forms[ORDSTAT].input_elems;
    p++->dptr = &iOS->d_id;
    p++->dptr = &iOS->c_id;
    p->dptr = (int *) &iOS->c_last[0];
    p = Forms[DELIVERY].input_elems;
    p->dptr = &iDY->o_carrier_id;
    p = Forms[STOCKLEV].input_elems;
    p->dptr = &iSL->threshold;
}

int setup_screen_buffer(struct form_info
*form_ptr, int txn_type)
{
    FILE *ifile;
    const text_elem *tbuf;
    char *bufp;
    int ct;
    char input_display_buf[64];
    io_elem *io_ptr;

    bufp = screen_buf;
    bufp += CLRSCN(bufp);
    tbuf = form_ptr->tp;
    while (tbuf->text) {
       bufp += DISPLAY(bufp, tbuf->y, tbuf->x,
tbuf->text);
       tbuf++;
    }
    bufp += SWITCH_TO_UNDERL(bufp);

    ct = 0;
    for (io_ptr = form_ptr->input_elems;
io_ptr->y != 999; io_ptr++) {

       strncpy(input_display_buf, blanks,
io_ptr->len);
       input_display_buf[io_ptr->len] = '\0';
```

```
      bufp += DISPLAY(bufp, io_ptr->x,
io_ptr->y, input_display_buf);
      ct++;

    }

    form_ptr->num_input_elems = ct;
    bufp += SWITCH_TO_NORMAL(bufp);

    if (txn_type == PAYMENT)
       /* shishir - changed for 11k wid
          *bufp += DISPLAY_INT(bufp, 4, 12, 4,
global_w_id);
        */
       bufp += DISPLAY_INT(bufp, 5, 12, 4,
global_w_id);
    else if (txn_type != 5)
       /* shishir - changed for 11k wid
          *bufp += DISPLAY_INT(bufp, 4, 12, 2,
global_w_id);
        */
       bufp += DISPLAY_INT(bufp, 5, 12, 2,
global_w_id);
    if (txn_type == STOCKLEV)
       bufp += DISPLAY_INT(bufp, 2, 29, 2,
global_d_id);
    bufp += SWITCH_TO_UNDERL(bufp);
    *bufp++ = '\1';
    *bufp = '\0';
    form_ptr->blank_formlen = bufp -
screen_buf + 1;
    if (!form_ptr->blank_form &&
       ((form_ptr->blank_form = mal-
loc(form_ptr->blank_formlen)) == NULL)) {
       Clog("setup_screen_buffer: malloc
failed\n");
       exit(1);
    }
    memcpy(form_ptr->blank_form, screen_buf,
form_ptr->blank_formlen);
    memset(screen_buf, '\0', form_ptr-
>blank_formlen);

}

int read_integer(col, row, size, flags, data)
int col, row, size, flags, *data;
{
    int exit_read_function = FALSE,
previous_data_exists = FALSE;
   int return_status = TAB, bytes_read = 0, i
= 0, j = 0, k = 0;
    int size1 = 0, cur_col = col;
    char *bufp, temp[50];
    double q;
    char erase_field[20];
    global_data_t*G;

    G = MY_GLOBAL_DATA;
    assert(G != NULL);

    strncpy(temp, " ", 1);
    bufp = G->g_screen_buf + G-
>g_screen_bufindex;
    /* Position cursor at start of field */

   if (G->g_curbuf_read == G->g_read_count ||
G->g_curbuf_read == 0) {
       G->g_screen_buf[0] = '\0';
       bufp += GOTOXY(bufp, col + size - 1,
row);
       PAINTSCRLEN(G->g_screen_buf, bufp - G-
>g_screen_buf);
       bufp = G->g_screen_buf;
    }
    size1 = size;

    if (*data > 0)
       previous_data_exists = TRUE;
    while (exit_read_function == FALSE) {

       /*
          * Below we read from standard input
into the array curbuf.
          * curbuf_read is the pointer to the
array curbuf indicating
          * the position upto which the curbuf
has been parsed.
          * curbuf_consumed is the number of
elements in the buffer
          * temp that holds the array that is
to be displayed.
          * Elements of curbuf_consumed is
selectively copied from
         * curbuf Note:read_count is the total
number of characters
          * in the buffer curbuf. curbuf_read
is always less than or
          * equal to read_count.
        */

       if (G->g_curbuf_read == G->g_read_count
|| G->g_curbuf_read == 0) {

           G->g_curbuf_read = 0;

           G->g_read_count = read(G->g_tty_in,
G->g_curbuf,
                                  sizeof(G-
>g_curbuf));
```

```
          if (Debug) {
             Clog("SBL:in read %d into cur-
buf:\n%s\n",
                    G->g_read_count, G-
>g_curbuf);
          }
          if (G->g_read_count == 0)
             syserr("read_integer: TTY lost
connection");
       }

       /*
        * int message prevents unnecessary dis-
play of warning messages
        */
       if (message == TRUE) {
           bufp += DISPLAY(bufp, MESSAGE_COL,
MESSAGE_ROW, ERASE_MSG);
           message = FALSE;
       }
       if (previous_data_exists == TRUE) {

          if (G->g_curbuf[G->g_curbuf_read] ==
DELETE) {
             previous_data_exists = FALSE;
             strncpy(erase_field, blanks,
size);
             erase_field[size] = '\0';
             bufp += DISPLAY(bufp, col, row,
erase_field);
             bufp += GOTOXY(bufp, col + size -
1, row);
          } else {
             if (G->g_curbuf[G-
>g_curbuf_read] < '0' || G->g_curbuf[G-
>g_curbuf_read] > '9') {
                exit_read_function = TRUE;
                previous_data_exists = FALSE;
                return_status = G-
>g_curbuf[G->g_curbuf_read];
               G->g_curbuf[G->g_curbuf_read]
= '\0';
             } else {
                previous_data_exists = FALSE;
                strncpy(erase_field, blanks,
size);
                erase_field[size] = '\0';
                bufp += DISPLAY(bufp, col,
row, erase_field);
                /*
                   * bufp = G->g_screen_buf;
                 */
             }
          }
       }                  /* if
previous_data_exists */
       while ((G->g_curbuf_read < G-
>g_read_count) && (exit_read_function ==
FALSE)) {
          /*
            * intermediate variable size1 for
cases when
             * floating point field whose size
is less than
              * actual size by 1 because of
decimal.
            */
           /* omit the size1 decrement (PDH);
this causes payment amount
              to fail if the full 7-char input
is used (e.g. 4180.11).
               It's not clear why this ever
worked. */
            /* if (payment_input == TRUE)
                size1 = size - 1; */
          /*
             * Test for integer
           */
          if (
             (G->g_curbuf[G->g_curbuf_read]
>= '0'
              && G->g_curbuf[G-
>g_curbuf_read] <= '9')
             || (G->g_curbuf[G-
>g_curbuf_read] == '.')) {
             /*
                * Consume all integers in
buffer
              */
             for (; G->g_curbuf_read < G-
>g_read_count &&
                  ((G->g_curbuf[G-
>g_curbuf_read] >= '0'
                    && G->g_curbuf[G-
>g_curbuf_read] <= '9')
                   || G->g_curbuf[G-
>g_curbuf_read] == '.'); G->g_curbuf_read++) {
                /*
                   * below we fill up temp
making sure
                   * the size limit is not
exceeded
                 */
                if (G->g_curbuf_consumed <
size1) {
                   temp[G->g_curbuf_consumed]
=
                         G->g_curbuf[G-
>g_curbuf_read];
                   G->g_curbuf_consumed++;
```

```
                }
                /*
                 * number of elements typed
in is
                 * more than the size of the
field
                 */
                else
                   OVERFLOW = TRUE;
                /*
                 * ensure the character is
removed after it is read
                 */
               G->g_curbuf[G->g_curbuf_read]
= '\0';
             }            /* end of for cur-
buf is legitimate number */

             temp[G->g_curbuf_consumed] =
'\0';/* terminate temp string */
             if (payment_input == TRUE) {/*
floating point field */
                /* convert the ascii to float
*/
                q = (atof(temp));
                bufp +=
                   DISPLAY_FLOAT(bufp, 2,
(col + size - 4), row, q);
                /*
                  if (G->g_curbuf_consumed <
3)
                   else
                   bufp +=
DISPLAY_FLOAT(bufp, 2, (col + size - G-
>g_curbuf_consumed - 1), row, q);
                 */
             } else {
                if (G->g_curbuf_consumed <
size + 1)
                   bufp +=
                       DISPLAY(bufp,
                              (col + size - G-
>g_curbuf_consumed), row,
                               temp);
                return_status = G-
>g_curbuf[G->g_curbuf_read];
                cur_col++;
             }
          }
          /* if curbuf[] between "0" and "9"
*/
          /*
             * if not integer, then test for
movement character
           */
          else if (G->g_curbuf[G-
>g_curbuf_read] == TAB
                  || G->g_curbuf[G-
>g_curbuf_read] == LF
                  || G->g_curbuf[G-
>g_curbuf_read] == BACKTAB
                  || G->g_curbuf[G-
>g_curbuf_read] == SUBMIT) {
             if (message == TRUE) {
                bufp +=
                   DISPLAY(bufp, MESSAGE_COL,
MESSAGE_ROW, ERASE_MSG);
                message = FALSE;
             }
             temp[G->g_curbuf_consumed] =
'\0';
             if (payment_input == TRUE) {
                q = atof(temp);
                *data = q * 100. + .125;
             }


             else {
                *data = atoi(temp);
             }
             exit_read_function = TRUE;
             return_status = G->g_curbuf[G-
>g_curbuf_read];
             G->g_curbuf[G->g_curbuf_read] =
'\0';
             G->g_curbuf_read++;
             G->g_curbuf_consumed = 0;
          }
          /* if curbuf[] a movement character
*/
          /*
             * if not integer of movement,
test for DELETE
           */
          else if (G->g_curbuf[G-
>g_curbuf_read] == DELETE) {
             if (payment_input == TRUE) {/*
for floating point
                                          *
field */
                if (G->g_curbuf_consumed !=
0)
                   G->g_curbuf_consumed--;
                if (message == TRUE) {
                   bufp +=
                       DISPLAY(bufp,
MESSAGE_COL, MESSAGE_ROW,
                               ERASE_MSG);
                   message = FALSE;
                }
```

```
                OVERFLOW = FALSE;
                PAINTSCR(G->g_screen_buf);
                temp[G->g_curbuf_consumed] =
'\0';
                q = atof(temp);
               G->g_curbuf[G->g_curbuf_read]
= '\0';
                strncpy(erase_field, blanks,
size);
                erase_field[size] = '\0';
                bufp = G->g_screen_buf;
                G->g_screen_bufindex = 0;
                bufp += DISPLAY(bufp, col,
row, erase_field);
                if (G->g_curbuf_consumed < 3)
                   bufp +=
                      DISPLAY_FLOAT(bufp, 2,
(col + size - 4), row,
                                    q);
                else
                   bufp +=
                       DISPLAY_FLOAT(bufp, 2,
                                  (col + size
- G->g_curbuf_consumed -
                                   1), row,
q);
                if (cur_col != 0)
                   cur_col--;
                if (G->g_curbuf_read < 40)
                   G->g_curbuf_read++;/*
pressed key overflow
                                    * situa-
tions */
                bufp += GOTOXY(bufp, col +
size, row);
             } else {
                if (G->g_curbuf_consumed !=
0)
                   G->g_curbuf_consumed--;
               G->g_curbuf[G->g_curbuf_read]
= '\0';
                G->g_curbuf_read++;
                if (message == TRUE) {
                   bufp +=
                       DISPLAY(bufp,
MESSAGE_COL, MESSAGE_ROW,
                               ERASE_MSG);
                   message = FALSE;
                }
                OVERFLOW = FALSE;
                PAINTSCR(G->g_screen_buf);
                temp[G->g_curbuf_consumed] =
'\0';
                strncpy(erase_field, blanks,
size);
                erase_field[size] = '\0';
                bufp = G->g_screen_buf;
                G->g_screen_bufindex = 0;
                bufp += DISPLAY(bufp, col,
row, erase_field);
                bufp +=
                   DISPLAY(bufp, (col + size
- G->g_curbuf_consumed), row,
                           temp);
                if (cur_col != 0)
                   cur_col--;
                bufp += GOTOXY(bufp, col +
size, row);
             }
          }
          /* end of if DELETE */
          /* could be a ^C */
          else if (G->g_curbuf[G-
>g_curbuf_read] == QUIT) {
             temp[0] = '\0';
             return_status = QUIT;
             G->g_curbuf[G->g_curbuf_read] =
'\0';
             exit_read_function = TRUE;
          } else {
                           /** Any other char-
acter entered at the keyboard ... **/
             if (message == FALSE) {
                bufp +=
                   DISPLAY(bufp, MESSAGE_COL,
MESSAGE_ROW,
                           INVALID_MSG);
                Clog("SBL:invalid message in
curbuf:\n%s\n", G->g_curbuf);
                bufp += GOTOXY(bufp, col +
size, row);
                PAINTSCR(G->g_screen_buf);
                bufp = G->g_screen_buf;
                G->g_screen_bufindex = 0;
                message = TRUE;
             }
             G->g_curbuf_read++;
          }
       }
             /** End of the WHILE loop **/
       if (OVERFLOW == TRUE &&
exit_read_function == FALSE) {
          /*
             * if number of characters are
exceeding the field
             * limit beep and warning message
is necessary
            */
          if (message == FALSE) {
             bufp += DISPLAY(bufp,
```

```
MESSAGE_COL,
                            MESSAGE_ROW,
EXC_FLD_LIM_MSG);
              PAINTSCR(G->g_screen_buf);
              bufp = G->g_screen_buf;
              G->g_screen_bufindex = 0;
              message = TRUE;
            }
            *data = atoi(temp);
            return_status = G->g_curbuf[G-
>g_curbuf_read];
            G->g_curbuf[G->g_curbuf_read] =
'\0';
            G->g_curbuf_read = 0;
            OVERFLOW = FALSE;
        } else {
            G->g_screen_bufindex = bufp - G-
>g_screen_buf;
            if ((G->g_curbuf_read == G-
>g_read_count)
              || (G->g_curbuf_read == 0)
              || (G->g_screen_bufindex >
SCRBUF_LEN - CURBUFLEN)) {


              PAINTSCRLEN(G->g_screen_buf, G-
>g_screen_bufindex);
              G->g_screen_bufindex = 0;
              bufp = G->g_screen_buf;
            }
        }
    }
    /* ensuring unnecessary warning messages
are removed */
    if (message == TRUE) {
       bufp += DISPLAY(bufp, MESSAGE_COL,
MESSAGE_ROW, ERASE_MSG);
       message = FALSE;
       PAINTSCR(G->g_screen_buf);
       bufp = G->g_screen_buf;
       G->g_screen_bufindex = 0;
    }
    return (return_status);
}

int read_string(col, row, size, flags, data)
int col, row, size, flags;
char *data;
{
    int exit_read_function = FALSE,
previous_data_exists =
       FALSE, data_full = FALSE;
   int return_status = TAB, bytes_read = 0, i
= 0, j = 0, size_tot = 0;
    char *bufp, temp[80];
    char erase_field[20];

    strncpy(temp, "\0", 1);
    curbuf_consumed = 0;
    bufp = screen_buf + screen_bufindex;
    /* Position cursor at start of field */

    if (curbuf_read == read_count ||
curbuf_read == 0) {
       screen_buf[0] = '\0';
       bufp += GOTOXY(bufp, col, row);/* Goto
input area */
       PAINTSCRLEN(screen_buf, bufp -
screen_buf);
       bufp = screen_buf;
    }
    if ((*(char *) data) != '\0')
       previous_data_exists = TRUE;
    while (exit_read_function == FALSE) {
       /*
          * Below we read from standard input
into the array curbuf.
          * curbuf_read is the pointer to the
array curbuf indicating
          * the position upto which the curbuf
has been parsed.
          * curbuf_consumed is the number of
elements in the buffer
          * temp that holds the array that is
to be displayed.
          * Elements of curbuf_consumed is
selectively copied from
         * curbuf Note:read_count is the total
number of characters
          * in the buffer curbuf. curbuf_read
is always less than or
          * equal to read_count.
        */
       if (curbuf_read == read_count) {
           curbuf_read = 0;
           read_count = read(tty_in, curbuf,
size - size_tot);
           if (Debug) {
              Clog("SBL:in read %d into cur-
buf:\n%s\n", read_count, curbuf);
           }
           if (read_count == 0)
              syserr("read_string: TTY lost
connection");
       }
       if (message == TRUE) {
           bufp += DISPLAY(bufp, MESSAGE_COL,
MESSAGE_ROW, ERASE_MSG);
           message = FALSE;
       }
```

```
      if (previous_data_exists == TRUE) {
          if (curbuf[curbuf_read] == DELETE) {
             previous_data_exists = FALSE;
             strncpy(erase_field, blanks,
size);
             erase_field[size] = '\0';
             bufp += DISPLAY(bufp, col, row,
erase_field);
             bufp += GOTOXY(bufp, col, row);
           } else {
             if (curbuf[curbuf_read] < ' ' ||
curbuf[curbuf_read] > '~') {
                 exit_read_function = TRUE;
                 previous_data_exists = FALSE;
                 return_status = cur-
buf[curbuf_read];
                 curbuf[curbuf_read] = '\0';

             } else {
                 previous_data_exists = FALSE;
                 strncpy(erase_field, blanks,
size);
                 erase_field[size] = '\0';
                 bufp += DISPLAY(bufp, col,
row, erase_field);
                 bufp += GOTOXY(bufp, col,
row);
             }
           }
      }
      while ((curbuf_read < read_count) &&
(exit_read_function == FALSE)) {
          if (curbuf[curbuf_read] >= ' ' &&
curbuf[curbuf_read] <= '~') {       /** if
between ASCII
                       space (040) through ~
(0176) **/
             for (; curbuf[curbuf_read] >= ' '
                  && curbuf[curbuf_read] <=
'~'; curbuf_read++) {
                 /*
                    * ensuring the
curbuf_consumed is
                    * not more than field size
                  */
                 if (curbuf_consumed < size) {
                    temp[curbuf_consumed] =
curbuf[curbuf_read];
                    curbuf_consumed++;
                 }
                 /* else overflow condition */
                 else
                    OVERFLOW = TRUE;
                 curbuf[curbuf_read] = '\0';
/* erasing characters
                                          *
already read from the
                                          *
buffer */
             }
             temp[curbuf_consumed] = '\0';/*
terminate temp string */
             bufp += DISPLAY(bufp, col, row,
temp);
             return_status = cur-
buf[curbuf_read];
           }
             else if (curbuf[curbuf_read] ==
TAB
                    || curbuf[curbuf_read] ==
LF
                    || curbuf[curbuf_read] ==
BACKTAB
                    || curbuf[curbuf_read] ==
SUBMIT) {
             if (curbuf_consumed > 0) {
                 if (message == TRUE) {
                    bufp +=
                        DISPLAY(bufp,
MESSAGE_COL, MESSAGE_ROW,
                                ERASE_MSG);
                    message = FALSE;
                 }
                 temp[curbuf_consumed] = '\0';
                 strcpy(data, temp);
                 exit_read_function = TRUE;
                 return_status = cur-
buf[curbuf_read];
                 curbuf[curbuf_read] = '\0';
                 curbuf_read++;
                 curbuf_consumed = 0;
             } else {
                 if (message == TRUE) {
                    bufp +=
                        DISPLAY(bufp,
MESSAGE_COL, MESSAGE_ROW,
                                ERASE_MSG);
                    message = FALSE;
                 }
                 temp[curbuf_consumed] = '\0';
                 strcpy(data, temp);
                 exit_read_function = TRUE;
                 return_status = cur-
buf[curbuf_read];
                 curbuf[curbuf_read] = '\0';
                 curbuf_read++;
             }
           } else if (curbuf[curbuf_read] ==
DELETE) {
             for (curbuf_read = curbuf_read;
```

```
                 curbuf[curbuf_read] ==
DELETE; curbuf_read++) {
                curbuf[curbuf_read] = '\0';
                temp[curbuf_consumed - 1] =
'\0';

                if (curbuf_consumed != 0)
                   curbuf_consumed--;
             }
             if (curbuf_consumed >= 0) {
                bufp += BLANK_UNDERLINE(bufp,
col, row, " ");
                bufp += DISPLAY(bufp, col,
row, temp);
                PAINTSCR(screen_buf);
                bufp = screen_buf;
                screen_bufindex = 0;
             } else {
                if (message == FALSE) {
                   bufp +=
                       DISPLAY
                       (bufp, MESSAGE_COL,
MESSAGE_ROW,
                        EXC_FLD_LIM_MSG);
                   bufp += BEEP(bufp);
                   PAINTSCR(screen_buf);
                   bufp = screen_buf;
                   screen_bufindex = 0;
                   message = TRUE;
                }
                curbuf[curbuf_read] = '\0';
                curbuf_read = 0;
             }
          } else if (curbuf[curbuf_read] ==
QUIT) {
             temp[0] = '\0';
             return_status = QUIT;
             curbuf[curbuf_read] = '\0';
             exit_read_function = TRUE;
          } else {  /** Any other character
entered at the keyboard ... **/
             if (message == FALSE) {
                bufp +=
                   DISPLAY(bufp, MESSAGE_COL,
MESSAGE_ROW,
                           INVALID_MSG);
                Clog("SBL:invalid message in
curbuf:\n%s\n", curbuf);
                bufp += GOTOXY(bufp, col,
row);
                message = TRUE;
             }
             curbuf_read++;
          }
       }
```

```
              /** End of the WHILE loop **/
       if (OVERFLOW == TRUE &&
exit_read_function == FALSE)
             /** If read enough to fill the
size already **/
       {
           if (message == FALSE) {
              bufp += DISPLAY(bufp,
MESSAGE_COL,
                              MESSAGE_ROW,
EXC_FLD_LIM_MSG);
              PAINTSCR(screen_buf);
              bufp = screen_buf;
              screen_bufindex = 0;
              message = TRUE;
           }
           OVERFLOW = FALSE;
           temp[curbuf_consumed] = '\0';
           strcpy(data, temp);
           curbuf_consumed--;
           return_status = curbuf[curbuf_read];
       } else {
           screen_bufindex = bufp - screen_buf;
           if ((curbuf_read == read_count)
              || (curbuf_read == 0)
              || (screen_bufindex > SCRBUF_LEN
- CURBUFLEN)) {
              PAINTSCRLEN(screen_buf,
screen_bufindex);
              screen_bufindex = 0;
              bufp = screen_buf;
           }
       }
    }
    if (message == TRUE) {
       bufp += DISPLAY(bufp, MESSAGE_COL,
MESSAGE_ROW, ERASE_MSG);
       message = FALSE;
       PAINTSCR(screen_buf);
       screen_bufindex = 0;
    }
    return (return_status);
}

void display_newo();
void display_paym();
void display_ords();
void display_del();
void display_stock();
void (*p_print_function[]) () = {


    &display_newo, &display_paym,
&display_ords, &display_del,
       &display_stock};
```

```
display_output(int txn_type)
{
    char c;

    (p_print_function[txn_type]) ();
    read(tty_in, &c, 1);
}

void display_newo()
{
    struct no_itm_struct *ol_ptr, *ool;

    char *bufp;
    int i, r;

    bufp = output_screen;

    if (oNO->status[0] == '\0') {

       PAINTSCR(EXECUTION_STATUS_MSG);
       return;

    } else {

       bufp += SWITCH_TO_NORMAL(bufp);
       bufp += DISPLAY(bufp, 61, 2, oNO-
>o_entry_d);
       bufp += DISPLAY(bufp, 25, 3, oNO-
>c_last);
       bufp += DISPLAY(bufp, 52, 3, oNO-
>c_credit);
       bufp += DISPLAY_FLOAT(bufp, 5, 64, 3,
oNO->c_discount);
       bufp += DISPLAY_INT(bufp, 8, 15, 4, oNO-
>o_id);
       bufp += DISPLAY_INT(bufp, 2, 42, 4, oNO-
>o_ol_cnt);
       bufp += DISPLAY_FLOAT(bufp, 5, 59, 4,
oNO->w_tax);
       bufp += DISPLAY_FLOAT(bufp, 5, 74, 4,
oNO->d_tax);
       ol_ptr = iNO->o_ol;
       ool = oNO->o_ol;

       for (i = 0, r = FIRST_OL_ROW; i < iNO-
>o_ol_cnt;
            r++, i++, ol_ptr++, ool++) {
           bufp += DISPLAY(bufp, 19, r, ool-
>i_name);
          bufp += DISPLAY_INT(bufp, 3, 51, r,
ool->s_quantity);
           bufp += DISPLAY(bufp, 58, r, ool-
>brand);
           bufp += DISPLAY_MONEY(bufp, 6, 62,
r, ool->i_price);
           bufp += DISPLAY_MONEY(bufp, 7, 71,
r, ool->ol_amount);
       }

       bufp += DISPLAY_MONEY(bufp, 8, 70, 22,
oNO->total);
       bufp += DISPLAY(bufp, 19, 22, oNO->sta-
tus);
       bufp += DISPLAY(bufp, 23, 75, "**((");
       *bufp++ = '\0';
       PAINTSCRLEN(output_screen, bufp -
output_screen);
    }
#ifdef DEBUG
    Clog("DBG: Screen output chars = %d\n",
(bufp - &output_screen[0]));
#endif
}

void display_paym()
{
    char *bufp, temp[51], tempbuf2[201];
    char *make_phone(char *), *make_zip(char
*);
    bufp = output_screen;
    bufp += SWITCH_TO_NORMAL(bufp);/* jr */

    if (oPT->c_id == 0) {
       PAINTSCR(EXECUTION_STATUS_MSG);
    } else {
    bufp += DISPLAY(bufp, 7, 2, oPT->h_date);
    bufp += DISPLAY(bufp, 1, 5, oPT-
>w_street_1);
    bufp += DISPLAY(bufp, 1, 6, oPT-
>w_street_2);
    bufp += DISPLAY(bufp, 1, 7, oPT->w_city);
    bufp += DISPLAY(bufp, 22, 7, oPT->w_state);
    bufp += DISPLAY(bufp, 25, 7, make_zip(oPT-
>w_zip));
    bufp += DISPLAY(bufp, 42, 5, oPT-
>d_street_1);
    bufp += DISPLAY(bufp, 42, 6, oPT-
>d_street_2);
    bufp += DISPLAY(bufp, 42, 7, oPT->d_city);
    bufp += DISPLAY(bufp, 63, 7, oPT->d_state);
    bufp += DISPLAY(bufp, 66, 7, make_zip(oPT-
>d_zip));
    bufp += DISPLAY_INT(bufp, 4, 11, 9, oPT-
>c_id);
    bufp += DISPLAY(bufp, 29, 10, oPT->c_last);
    bufp += DISPLAY(bufp, 9, 10, oPT->c_first);
    bufp += DISPLAY(bufp, 26, 10, oPT-
>c_middle);
    bufp += DISPLAY(bufp, 9, 11, oPT-
>c_street_1);
```

```
    bufp += DISPLAY(bufp, 9, 12, oPT-
>c_street_2);
    bufp += DISPLAY(bufp, 9, 13, oPT->c_city);
    bufp += DISPLAY(bufp, 30, 13, oPT-
>c_state);
    bufp += DISPLAY(bufp, 33, 13, make_zip(oPT-
>c_zip));
    bufp += DISPLAY(bufp, 58, 10, oPT-
>c_since);
    bufp += DISPLAY(bufp, 58, 11, oPT-
>c_credit);
    bufp += DISPLAY_FLOAT(bufp, 5, 58, 12, oPT-
>c_discount);
    bufp += DISPLAY(bufp, 58, 13,
make_phone(oPT->c_phone));
    bufp += DISPLAY_MONEY(bufp, 14, 55, 15,
oPT->c_balance);
    bufp += DISPLAY_MONEY(bufp, 13, 17, 16,
oPT->c_credit_lim);

    if (oPT->c_data_1[0] != ' ' || oPT-
>c_data_1[0] != '\0') {
       bufp += DISPLAY50(bufp, 12, 18, oPT-
>c_data_1);
       bufp += DISPLAY50(bufp, 12, 19, oPT-
>c_data_2);
       bufp += DISPLAY50(bufp, 12, 20, oPT-
>c_data_3);
       bufp += DISPLAY50(bufp, 12, 21, oPT-
>c_data_4);
    }
    if (!oPT->h_date)
       bufp += DISPLAY(bufp, MESSAGE_COL,
MESSAGE_ROW - 2, BAD_INPUTS);
    bufp += DISPLAY(bufp, 23, 75, "**((");
    } /* end of if (false) else */
    *bufp++ = '\0';
    PAINTSCRLEN(output_screen, bufp -
output_screen);
#ifdef DEBUG
    Clog("DBG: Screen output chars = %d\n",
(bufp - &output_screen[0]));
#endif
}

void display_ords()
{
    struct ord_itm_struct *sol;


    char *bufp;
    int i = 0, r = 8;

    if (oOS->c_id == 0) {
       PAINTSCR(EXECUTION_STATUS_MSG);
       return;

    } else {

    bufp = output_screen;
    bufp += SWITCH_TO_NORMAL(bufp);
    bufp += DISPLAY_INT(bufp, 4, 11, 3, oOS-
>c_id);
    bufp += DISPLAY(bufp, 44, 3, oOS->c_last);
    bufp += DISPLAY(bufp, 24, 3, oOS->c_first);
    bufp += DISPLAY(bufp, 41, 3, oOS-
>c_middle);
    bufp += DISPLAY_MONEY(bufp, 9, 15, 4, oOS-
>c_balance);
    bufp += DISPLAY_INT(bufp, 8, 15, 6, oOS-
>o_id);
    bufp += DISPLAY(bufp, 38, 6, oOS-
>o_entry_d);
    bufp += DISPLAY_INT(bufp, 2, 76, 6, oOS-
>o_carrier_id);


    for (i = 0; i < oOS->ol_cnt; i++) {

       sol = &oOS->s_ol[i];

       if (sol->ol_supply_w_id > 0) {
           bufp += DISPLAY_INT(bufp, 4, 3, r,
sol->ol_supply_w_id);
           bufp += DISPLAY_INT(bufp, 6, 14, r,
sol->ol_i_id);
           bufp += DISPLAY_INT(bufp, 2, 25, r,
sol->ol_quantity);
           bufp += DISPLAY_MONEY(bufp, 8, 32,
r, sol->ol_amount);
           bufp += DISPLAY(bufp, 47, r, sol-
>ol_delivery_d);
           r++;
       }
    }
    if (!oOS->ol_cnt)
       bufp += DISPLAY(bufp, MESSAGE_COL,
MESSAGE_ROW - 2, BAD_INPUTS);

    bufp += DISPLAY(bufp, 23, 75, "**((");
    *bufp++ = '\0';
    PAINTSCRLEN(output_screen, bufp -
output_screen);
#ifdef DEBUG
    Clog("DBG: Screen output chars = %d\n",
(bufp - &output_screen[0]));
#endif
   }
}
```

```
void display_del()
{
    char *bufp;

    bufp = output_screen;
    /*PAINTSCR(DELIVERY_QUEUED_MSG); */
    bufp += sprintf(bufp, "%s",
DELIVERY_QUEUED_MSG);
    bufp += DISPLAY(bufp, 23, 75, "**((");
    *bufp++ = '\0';
    PAINTSCRLEN(output_screen, bufp -
output_screen);
#ifdef DEBUG
    Clog("DBG: Screen output chars = %d\n",
(bufp - &output_screen[0]));
#endif
}

void display_stock()
{
    char *bufp;
    bufp = output_screen;
    bufp += SWITCH_TO_NORMAL(bufp);/* jr */

    if (oSL->low_stock == -1) {

       PAINTSCR(EXECUTION_STATUS_MSG);
       return;

    } else {

    bufp += DISPLAY_INT(bufp, 3, 12, 6, oSL-
>low_stock);
    bufp += DISPLAY(bufp, 23, 75, "**((");
    *bufp++ = '\0';
    PAINTSCRLEN(output_screen, bufp -
output_screen);
#ifdef DEBUG
    Clog("DBG: low stock:%d\n", oSL-
>low_stock);
    Clog("DBG: Screen output chars = %d\n",
(bufp - &output_screen[0]));
#endif
    }
}

char *make_phone(char *data)
{
    static char tempphone[20];
    strncpy(tempphone, data, 6);
    tempphone[6] = '-';
    strncpy(&tempphone[7], &data[6], 3);
    tempphone[10] = '-';
    strncpy(&tempphone[11], &data[9], 3);
    tempphone[14] = '-';
    strncpy(&tempphone[15], &data[12], 4);
    tempphone[19] = '\0';
    return tempphone;
}

char *make_zip(char *data)
{
    static char temp[10];

    strncpy(temp, data, 5);
    temp[5] = '-';
    strncpy(&temp[6], &data[5], 4);
    temp[10] = '\0';
    return temp;
}
```

tpcc_forms.h

```
/*********************************************
*************************
tpcc_forms.h
*********************************************
*************************/

#include <termio.h>

#define MAX_FORMS 6
#define MESSAGE_ROW 24
#define MESSAGE_COL 1
#define RTE_SYNCH_CHARACTER '\1'
#define SCRBUF_LEN 1536
#define FIRST_OL_ROW 7

#define CLRSCN(buf)
sprintf(buf,"\033[H\033[2J")
#define DISPLAY_INT(buf,wid,x,y,ip)
sprintf(buf,"\033[%d;%dH%*.1d", y, x,wid,ip)
#define DISPLAY_MONEY(buf,wid,x,y,fp)
sprintf(buf,"\033[%d;%dH$%#*.2f",y,x,wid,fp)
#define DISPLAY_FLOAT(buf,wid,x,y,fp)
sprintf(buf,"\033[%d;%dH%#*.2f", y, x,wid,fp)
#define DISPLAY(buf,x,y,txt)
sprintf(buf,"\033[%d;%dH%s", y, x,txt)
#define DISPLAY50(buf,x,y,txt)
sprintf(buf,"\033[%d;%dH%50.50s", y, x,txt)
#define PAINTSCR(buf)
write(tty_out,buf,strlen(buf))
#define PAINTSCRLEN(buf, len)
write(tty_out,buf,len)
#define SWITCH_TO_NORMAL(buf)
sprintf(buf,"\033[m")
#define SWITCH_TO_UNDERL(buf)
sprintf(buf,"\033[4m")
```

```
#define GOTOXY(buf,x,y)
sprintf(buf,"\033[%d;%dH", y, x)
#define BEEP(buf) sprintf(buf,"\007")
#define BLANK_UNDERLINE(buf,x,y,txt)
sprintf(buf,"\033[4m;\033[%d;%dH%s",y,x,txt);

  /** Possible status values returned by read
functions **/

#define CANCELLED 3
#define PREVIOUS_FIELD 4

  /** 52 Possible key strokes read in by the
read functions. Some are also
          returned as status from the read
functions.  **/

#define BACKTAB 2/** Decided to use the CTRL B
for now **/
#define DELETE 8
#define ESCAPE 27
#define LF 10
#define QUIT 3/** CNTRL-C Key stroke to quit
for now **/
#define SPACE 32
#define SUBMIT 13/** Done with screen and sub-
mit key: CR **/

#define TAB 9
#define UNDERLINE 95

#define LEAVE_SCREEN_MIN ((unsigned char)300)
/* Min # chars to leave screen */
#define LEAVE_SCREEN_TIMEOUT 2 /* Min time to
leave screen, 10=1sec */

#define CURBUFLEN 300

extern void syserr ();
void tpcc_forms_init();
void Init_Screen ();
void display_screen_array (int);
void Send_Menu ();
int Get_Menu_Input ();

int read_integer (int, int, int, int, int *);
int read_money (int, int, int, int, float *);
int read_string (int, int, int, int, char *);

         /**
         The following is the struct type used
to define the I/O element
          y is the row position on the screen.
          x is the column position on the
screen.
          len is the size of the data field in
bytes.
          flags is the indicator of manadatory
data fields. '1' means required.
          dptr is the pointer to the data.
          fptr is the pointer to the read fun-
tion for the type of data dptr
          points to.
          **/
typedef struct
{
  int y;
  int x;
  int len;
  int flags;
  int *dptr;
  int (*fptr) ();
} io_elem;

typedef struct
{
  int x;
  int y;
  char *text;
} text_elem;

struct form_info
{
  const text_elem *tp;
  char *blank_form;
  int blank_formlen;
  io_elem *input_elems;
  int num_input_elems;
};


#define NEWORDER_INPUTS_LEN 48
#define PAYMENT_INPUTS_LEN  7
#define ORDSTAT_INPUTS_LEN 4
#define DELIVERY_INPUTS_LEN 2
#define STOCKLEV_INPUTS_LEN 2
#define WD_INPUTS_LEN  3

#define menu_buflen (sizeof (menu_buf))
```

tpcc_global.c

```
/*
 * We gather all the global data here in order
to multi-thread
 * the tuxedo client.
 */
```

```
#include <pthread.h>
#include “tpcc_client.h”
#include “tpcc_forms.h”
#include “tpcc_tux.h”
#include “tpcc_global.h”

int          Debug = 0;
pthread_key_tThreadKey;/* key for thread spe-
cific data */
int          ClientNum;/* which process are
we?    */
```

tpcc_global.h

```
/*
 * We gather all the global data here in order
to multi-thread
 * the tuxedo client.
 */

extern int   Debug;
extern pthread_key_tThreadKey;/* key for
thread specific data */
extern int   ClientNum;/* which process are
we?    */

typedef struct global_data {

       /** used by tpcc_forms.c **/
       int    g_screen_bufindex;
       char    g_screen_buf[SCRBUF_LEN];
       int    g_w_id;
       int    g_d_id;
       int    g_tty_in;    /* input and out-
put tty fds */
       int    g_tty_out;
       int    g_curbuf_consumed;
       int    g_curbuf_read;
       int    g_read_count;
       char   g_curbuf[CURBUFLEN];
       BOOLEAN g_OVERFLOW;
       BOOLEAN g_message;  /* for suppressing
warning messages */
       BOOLEAN g_payment_input; /* for set-
ting money condition in read_int */
       struct termio g_tbufsave;
       int    g_int_h_amount;
       char   g_output_screen[SCRBUF_LEN];

       io_elem
g_neworder_inputs[NEWORDER_INPUTS_LEN];
       io_elem
g_payment_inputs[PAYMENT_INPUTS_LEN];
       io_elem
g_ordstat_inputs[ORDSTAT_INPUTS_LEN];
       io_elem
g_delivery_inputs[DELIVERY_INPUTS_LEN];
       io_elem
g_stocklev_inputs[STOCKLEV_INPUTS_LEN];
       io_elemg_wd_inputs[WD_INPUTS_LEN];

       struct form_info g_Forms[MAX_FORMS];

       /** used by tpcc_tux.c **/
       /*
        * Data is returned in tuxibuf. Some
#defines are set up in tpcc_tux.h
        * for easy access/naming of the input
and output areas
        */
       char           *g_tuxibuf;
       char           *g_tuxobuf;

       /* general */
       pthread_tg_my_thread_id;

} global_data_t;

/*
 * Each thread gets its own copy of this global
data.
 * Note that the parent doesn’t have one!
 */
#define MY_GLOBAL_DATA  ((global_data_t
*)pthread_getspecific(ThreadKey))

#define screen_bufindex(MY_GLOBAL_DATA-
>g_screen_bufindex)
#define screen_buf(MY_GLOBAL_DATA-
>g_screen_buf)
#define global_w_id(MY_GLOBAL_DATA->g_w_id)
#define global_d_id(MY_GLOBAL_DATA->g_d_id)
#define tty_in      (MY_GLOBAL_DATA->g_tty_in)
#define tty_out     (MY_GLOBAL_DATA-
>g_tty_out)
#define curbuf_consumed(MY_GLOBAL_DATA-
>g_curbuf_consumed)
#define curbuf_read(MY_GLOBAL_DATA-
>g_curbuf_read)
#define read_count(MY_GLOBAL_DATA-
>g_read_count)
#define curbuf      (MY_GLOBAL_DATA->g_curbuf)
#define OVERFLOW(MY_GLOBAL_DATA->g_OVERFLOW)
#define message     (MY_GLOBAL_DATA-
>g_message)
#define payment_input(MY_GLOBAL_DATA-
>g_payment_input)
#define tbufsave(MY_GLOBAL_DATA->g_tbufsave)
```

```
#define int_h_amount(MY_GLOBAL_DATA-
>g_int_h_amount)
#define output_screen(MY_GLOBAL_DATA-
>g_output_screen)
#define neworder_inputs(MY_GLOBAL_DATA-
>g_neworder_inputs)
#define payment_inputs(MY_GLOBAL_DATA-
>g_payment_inputs)
#define ordstat_inputs(MY_GLOBAL_DATA-
>g_ordstat_inputs)
#define delivery_inputs(MY_GLOBAL_DATA-
>g_delivery_inputs)
#define stocklev_inputs(MY_GLOBAL_DATA-
>g_stocklev_inputs)
#define wd_inputs(MY_GLOBAL_DATA->g_wd_inputs)
#define Forms       (MY_GLOBAL_DATA->g_Forms)
#define tuxibuf     (MY_GLOBAL_DATA-
>g_tuxibuf)
#define tuxobuf     (MY_GLOBAL_DATA-
>g_tuxobuf)
#define my_thread_id(MY_GLOBAL_DATA-
>g_my_thread_id)
```

tpcc_login.c

```
/*
 * Login user/rte.
 * Prompt for username/password and verify the
user.
 */

#include <stdio.h>
#include <fcntl.h>
#include <unistd.h>
#include <errno.h>

static const char Myname[] =
“tpcc_login_user”;
int                 debug = 0;

/*
 * Conduct the login dialog
 *
 * Return -1 on failure.
 */
int
tpcc_login_user(int inFd, int outFd, char
*userName)
{
       int ret;
       int loginOK;
       int failcount;
       char *s1 = “login: \r\n”;
       char *s2 = “Password: \r\n”;
       char *s3 = “Login incorrect\r\n\r\n”;
       char *p;
       char buf[1024];
       char *mygetline(int fd, char *buf, int
buflen);

       for (failcount = 0; failcount < 5; fail-
count++) {
              loginOK = 0;
              ret = write(outFd, s1,
strlen(s1));
              if (ret == -1) {
                     Clog(“%s: launch: cannot
write to fd %d, errno=%d\n”,
                            Myname, outFd,
errno);
                     perror(“write”);
                     return(-1);
              }

              p = mygetline(inFd, buf,
sizeof(buf));
              if ((p != NULL) && strcmp(p,
userName) == 0)
                     loginOK = 1;

              ret = write(outFd, s2,
strlen(s2));
              if (ret == -1) {
                     Clog(“%s: launch: cannot
write to fd %d, errno=%d\n”,
                            Myname, outFd,
errno);
                     perror(“write”);
                     return(-1);
              }

              p = mygetline(inFd, buf,
sizeof(buf));
              if (loginOK) {
                     break;
              }
              write(outFd, s3, strlen(s3));
       }
       if (! loginOK) {
              Clog(“%s: login failed too many
times.  Dying.\n”, Myname);
              return(-1);
       }

       return(0);
}

char *
mygetline(int fd, char *buf, int buflen)
```

```
{
       int ret;
       int nchar;
       char *p;

       nchar = 0;
       p = buf;

       while (nchar < buflen) {
              ret = read(fd, buf + nchar,
buflen - nchar);
              if (ret == -1) {
                     Clog("%s: read error on fd
%d, errno=%d\n",
                            Myname, fd, errno);
                     perror("read");
                     return(NULL);
              } else if (ret == 0) {
                     Clog("%s: EOF on read of
fd %d, errno=%d\n",
                            Myname, fd, errno);
                     return(NULL);
              }

              nchar += ret;

              for (;p - buf < nchar; p++)
                     if (*p == '\n' || *p ==
'\r') {
                            *p = '\0';
                            return buf;
                     }
       }
       Clog("%s: no EOL in %d characters. Giv-
ing up.\n", Myname, buflen);
       return(NULL);
}
```

tpcc_tux.c

```
/***************** monitor.c ***************/
/* ** monitor.c -- All functions for Tuxedo
call and return */
#include <stdio.h>
#include <stdarg.h>
#include <atmi.h>
#include <pthread.h>
#include "tpcc_client.h"
#include "tpcc_tux.h"
#include "tpcc_forms.h"
#include "tpcc_global.h"

const char *svc_names[] = {
       "NEWO", "PAYM", "ORDS", "DEL", "STOCK"
};

int
Snd_Txn_To_Monitor_i(int txn_type)
{
       long olen = sizeof(struct io_tpcc);

#if DEBUG
       Clog("DBG: In Snd_Txn_To_Monitor\n");
       print_input_data(txn_type);
#endif

       tperrno = 0;
       if (txn_type == DELIVERY) {
              if ( tpacall((char
*)svc_names[txn_type],
                            tuxibuf, ilen,
TPNOREPLY) == -1)
                     return(-100);
       } else {
              if (tpcall((char
*)svc_names[txn_type],
                     (char *)tuxibuf, ilen,
                     &tuxobuf, &olen, 0) == -1)
                        return(-100);
       }
       return(0);
}

/* log a detailed error message */
void
log_detailed_error()
{
       int detail;

       detail = tperrordetail(0);

       Clog("DETAILS: tperrno(%d):%s \n\ttper-
rdetail=(%d):%s\n",
              tperrno,
              tpstrerror(tperrno),
              detail,
              tpstrerrordetail(detail, 0));
}

#defineI_DLY 50
#defineN_DLY 15

int
Snd_Txn_To_Monitor(int txn_type)
{
       int i;
       int delay;

       delay = I_DLY;
```

```
	for (i = 0; i < N_DLY; i++) {
		if
(Snd_Txn_To_Monitor_i(txn_type) == 0)
			break;
		poll(0,0,delay);
		delay = delay * 3 / 2;
	}
	if (i > 0 && i < N_DLY) {
		Clog("WARN: Retried %s %d
times\n",
			(char
*)svc_names[txn_type], i);
		log_detailed_error();
	} else if (i >= N_DLY) {
		Clog("ERR: Retried %s %d times,
failed\n",
			(char
*)svc_names[txn_type], i);
		log_detailed_error();
		return(-100);
	}

	return 0;
}

/*
 * If we are running one context per process,
then
 * do the tpinit(NULL) here. Otherwise, do the
tpinit(MULTICONTEXT)
 * in Init_Monitor_Per_Context().
 */
int Init_Monitor_Once()
{

#ifdef SINGLE_CTX
	if (tpinit(NULL) == -1) {
		tpmerror("tpinit", tperrno);
		return -1;
	}
#endif
	return(0);
}

/*
 * Create a new context, set it as the current
context, and return it.
 */
int
Init_Monitor_Per_Ctx(TPCONTEXT_T *ctx)
{
	static TPINIT *tpinitbuf = NULL;

	if (tpinitbuf == NULL) {/* first time
*/
		tpinitbuf = (TPINIT *)tpal-
loc("TPINIT", NULL, TPINITNEED(0));
		if (tpinitbuf == NULL) {
			Clog("tpalloc failed,
%s\n", tpstrerror(tperrno));
			return(-1);
		}
		tpinitbuf->flags = TPMULTICON-
TEXTS;
	}
	if (tpinit(tpinitbuf) == -1) {
		Clog("tpinit failed, %s\n",
tpstrerror(tperrno));
		return(-1);
	}
	if (tpgetctxt(ctx, 0) == -1) {
		Clog("Init_Monitor_Per_Ctx:
tpgetctx failed, %s\n",
			tpstrerror(tperrno));
		return(-1);
	}
	userlog("tpcc client process: ini-
tialzed new context\n");
	return(0);
}

/*
 * Each thread in a process calls this.
 * Alloc per thread data. Set thread's con-
text.
 */
int
Init_Monitor_Per_Thread(TPCONTEXT_T *ctx)
{
	if (tpsetctxt(*ctx, 0) == -1) {
		Clog("tpsetctxt failed, %s\n",
tpstrerror(tperrno));
		return(-1);
	}

	if ((tuxibuf = tpalloc("CARRAY", NULL,
ilen)) == NULL) {
		Clog("tpalloc failed, %s\n",
tpstrerror(tperrno));
		tpmerror("tpalloc", tperrno);
		return (-1);
	}

	if ((tuxobuf = tpalloc("CARRAY", NULL,
ilen)) == NULL) {
		Clog("tpalloc failed, %s\n",
tpstrerror(tperrno));
		tpmerror("tpalloc", tperrno);
		return (-1);
```

```
	}
	return(0);
}

Rundown_Monitor()
{
	int status;

	if (tpterm() == -1) {
		Clog("tpterm failed, %s\n",
tpstrerror(tperrno));
	}
	else {
		Clog("tpterm OK\n");
	}
}

tpmerror(char *service_called, int errnum)
{
	char errmsg[256];

	fprintf(stderr, "\033[24;1H\033[mTUX-
EDO: Failed %s with error: %s\n",
	     service_called, tpstrerror(err-
num));
	fprintf(stderr, "\n");
}

#ifdef DEBUG
print_input_data(int type)
{
	int i;
	time_t the_time;
	the_time = time(&the_time);
	Clog("DBG:=============TIME: %s == ==
== == == == ==\n",ctime(&the_time));

	switch (type) {
	case NEWORDER:
		Clog("DBG: NEWORDER INPUTS at
%s\n", ctime(&the_time));
		Clog("DBG: w_id: %d, d_id:
%d, c_id: %d o_ol_cnt: %d\n",
		iNO->w_id, iNO->d_id, iNO-
>c_id, iNO->o_ol_cnt);
		for (i = 0; i < iNO-
>o_ol_cnt; i++)
		Clog("DBG: ol_i_id: %d,
ol_supply_w_id: %d, ol_quantity: %d \n ",\
		iNO->o_ol[i].ol_i_id, iNO-
>o_ol[i].ol_supply_w_id,iNO-
>o_ol[i].ol_quantity);
		break;
	    case PAYMENT:
	       Clog("DBG: PAYMENT INPUTS at %s
\n ",ctime(&the_time));
		Clog("DBG: w_id: %d, d_id:
%d\n", iPT->w_id, iPT->d_id);
		Clog("DBG: c_last: %s ", iPT-
>c_last);
		Clog(" c_id: %d", iPT->c_id);
		Clog(" c_w_id: %d, c_d_id:
%d\n", iPT->c_w_id, iPT->c_d_id);
		Clog("DBG: h_amount: %f\n",
iPT->h_amount);
		break;
	    case ORDSTAT:
	       Clog("DBG: ORDER STATUS INPUTS at
%s \n ",ctime(&the_time));
		Clog("DBG: w_id: %d, d_id:
%d\n", iOS->w_id, iOS->d_id);
		Clog("DBG: c_id: %d, c_last:
%s\n",
		iOS->c_id, iOS->c_last);
		break;
	    case DELIVERY:
	       Clog("DBG: DELIVERY INPUTS at
%s\n", ctime(&the_time));
		Clog("DBG: w_id: %d,
o_carrier_id: %d\n", iDY->w_id, iDY -
>o_carrier_id);
		break;
	    case STOCKLEV:
	       Clog("DBG: STOCK LEVEL INPUTS at
%s \n ",ctime(&the_time));
		Clog("DBG: w_id: %d, d_id:
%d, threshold: %d\n", iSL ->w_id, iSL-
>d_id,iSL->threshold);
		break;
other:
		Clog("DBG: Txn_type = %d is
illegal at %s \n",type,ctime(&the_time));
	}
	return;
}
#endif /* ifdef DEBUG */
```

tpcc_tux.h

```
/********** monitor.h ********

/* ** monitor.h -- All Tuxedo definitions and
storage ** */

#define ilen(sizeof(struct io_tpcc))

/*
 * Data is returned in tuxibuf. Some #defines
are set up here for easy
```

```
 * access/naming of the input and output areas
 */

#define oNO (&((info_t *) tuxobuf)->neworder)
#define oPT (&((info_t *) tuxobuf)->payment)
#define oOS (&((info_t *) tuxobuf)->ordstat)
#define oDY (&((info_t *) tuxobuf)->delivery)
#define oSL (&((info_t *) tuxobuf)->stocklev)
#define iNO (&((info_t *) tuxibuf)->neworder)
#define iPT (&((info_t *) tuxibuf)->payment)
#define iOS (&((info_t *) tuxibuf)->ordstat)
#define iDY (&((info_t *) tuxibuf)->delivery)
#define iSL (&((info_t *) tuxibuf)->stocklev)
#define iWD (&((info_t *) tuxibuf)->wd)

extern void     Clog(char *,...);
```

.tpclogin.c

```
/*
 * Initialize tpcc client process. Login
user/rte.
 *
 * Originally was a main, run with no arguments
from inetd, run from
 * command line with -d. Now called from multi-
threadded client.
 */

#include <stdio.h>
#include <fcntl.h>
#include <pwd.h>
#include <unistd.h>

/*
 * Run-time parameters, set by invocation
arguments.
 */
char *myname;
int debug;
char *client_user;
char *error_output;
char *environment;

void set_defaults();
void usage();

/*
 * local procedures
 */
void inetd_setup();
void launch();

/*
 * MAIN
 *
 * Crack the arguments and make the top-level
calls.
 */
int
setp_environment(int argc, char *argv[])
{
       /* getopt linkage */
       int c;
       int nargs;
       extern int optind;
       extern char * optarg;

       myname = *argv;

       set_defaults();

       while ((c = getopt(argc, argv,
“E:c:e:dh”)) != EOF)
       switch (c) {
       case ‘E’:
              environment = optarg;
              break;
       case ‘c’:
              client_user = optarg;
              break;
       case ‘e’:
              error_output = optarg;
              break;
       case ‘d’:
              debug++;
              break;
       case ‘h’:
       case ‘?’:
       default:
              usage();
       }

       /*
        * Set up, includes permissions, redi-
rections, etc.
        */
       if (debug)
              fprintf(stderr, “%s: running in
debug mode from a terminal\n”,
                     myname);
       else
              inetd_setup();

       char **load_environment();

       /* Load the environment from a file */
```

```
	envp = load_environment();

	return(0);
}


void
set_defaults()
{
	debug = 0;
	error_output = “tpcc.stderr”;
	client_user = “tpcc”;
	environment = “tpclogin_profile”;
}

void
usage()
{
	set_defaults();
	fprintf(stderr, “Usage: %s
<options>\n”, myname);
	fprintf(stderr, “Options:\n”);
	fprintf(stderr, “\t-d (set debug
mode)\n”);
	fprintf(stderr, “\t-c <client user id
(%s)>\n”, client_user);
	fprintf(stderr, “\t-e <error log file
(%s)>\n”, error_output);
	fprintf(stderr, “\t-E <environment file
(%s)>\n”, environment);
	exit(1);
}

/*
 * If we are launched from inetd (or from our
own listner),
 * we have some work to do to create the envi-
ronment.
 *
 * Some of these calls will fail if we are not
root.
 */
void
inetd_setup()
{
	int i;
	int ret;
	struct passwd *p;

	/*
	 * Clean up the environment a little
	 */
	for (i = 3; i <= 10; i++)
		close(i);
	alarm(0);

	/*
	 * Look up the user in the /etc/passwd
file.
	 */
	for (;;) {
		p = getpwent();
		if (p == NULL) {
			fprintf(stderr, “%s: can-
not find user %s in the passwd file.\n”,
				myname,
client_user);
			exit(1);
		}
		if (strcmp(p->pw_name,
client_user) == 0)
			break;
	}
	endpwent();

	/*
	 * Go to the working directory.
	 */
	ret = chdir(p->pw_dir);
	if (ret == -1) {
		fprintf(stderr, “%s: cannot chdir
to %s\n”,
			myname, p->pw_dir);
		perror(“chdir”);
		exit(1);
	}

	/*
	 * Connect stderr to a file here.
	 */
	ret = open(error_output, O_WRONLY |
O_CREAT | O_APPEND, 0666);
	if (ret == -1) {
		fprintf(stderr, “%s: cannot open
%s for “,
			myname, error_output);
		fprintf(stderr, “write-only,
creat, append\n”);
		perror(“open”);
		exit(1);
	}
	ret = dup2(ret, 2);
	if (ret == -1) {
		fprintf(stderr, “%s: cannot dup2
into fd 2\n”, myname);
		perror(“dup2”);
		exit(1);
	}
	setlinebuf(stderr);
```

```
	/*
	 * set the IDs.  No error checking /
ignore errors.
	 */
	setgid(p->pw_gid);
	setuid(p->pw_uid);

}

/*
 * LAUNCH
 *
 * Conduct the login dialog
 */
void
tpcc_login_user(int inFd, int outFd)
{
	int ret;
	int loginOK;
	int failcount;
	char *s1 = "login: \r\n";
	char *s2 = "Password: \r\n";
	char *s3 = "Login incorrect\r\n\r\n";
	char *p;
	char **envp;
	char *argv[3];
	char *mygetline(int fd);

	for (failcount = 0; failcount < 5; fail-
count++) {
		loginOK = 0;
		ret = write(outFd, s1,
strlen(s1));
		if (ret == -1) {
			fprintf(stderr, "%s:
launch: cannot write to fd %d\n",
				myname, outFd);
			perror("write");
			exit(2);
		}

		p = mygetline(inFd);
		if (strcmp(p, client_user) == 0)
			loginOK = 1;

		ret = write(outFd, s2,
strlen(s2));
		if (ret == -1) {
			fprintf(stderr, "%s:
launch: cannot write to fd %d\n",
				myname, outFd);
			perror("write");
			exit(2);
		}
		p = mygetline(inFd);
		if (loginOK) {
			break;
		}
		write(outFd, s3, strlen(s3));
	}
	if (! loginOK) {
		fprintf(stderr, "%s: login failed
too many times.  Dying.\n",
			myname);
		exit(3);
	}
}

char *
mygetline(int fd)
{
	int ret;
	int nchar;
	char *p;
	static char buf[1024];

	nchar = 0;
	p = buf;

	while (nchar < sizeof buf) {
		ret = read(fd, buf + nchar,
(sizeof buf) - nchar);
		if (ret == -1) {
			fprintf(stderr, "%s: read
error on fd %d\n", myname,fd);
			perror("read");
			exit(2);
		} else if (ret == 0) {
			fprintf(stderr, "%s: EOF
on read of fd %d\n",myname,fd);
			exit(2);
		}

		nchar += ret;

		for (;p - buf < nchar; p++)
			if (*p == '\n' || *p ==
'\r') {
				*p = '\0';
				return buf;
			}
	}
	fprintf(stderr, "%s: no EOL in %d char-
acters.  Giving up.\n",
		myname, sizeof buf);
	exit(3);
	/* NOT REACHED */
}
```

```
/*
 * Load the environment up from the environ-
ment file.
 *
 * Two pass load.  First pass counts the lines,
second loads them.
 *
 * The file contains strings of the form
var=val\n, no whitespace.
 * Lines begining with '#' are ignored.
 * Empty lines are ignored.
 *
 */

static char *null_env[] = { (char *) 0, };

char **
load_environment()
{
       int n;
       int count;
       FILE *envf;
       char *q;
       char **envp, **p;
       char lbuf[128];
       char *malloc();

       /*
        * Open the file.  If we can't assume
its missing
        * and use a null environment.
        */
       envf = fopen(environment, "r");
       if (envf == NULL)
              return null_env;

       /*
        * First pass, count the lines.
        */
       count = 0;
       while (fgets(lbuf, sizeof lbuf, envf)
!= NULL) {
              n = strlen(lbuf) - 1;
              if (lbuf[n] != '\n') {
                     fprintf(stderr, "%s: envi-
ronment string longer than",
                                   myname);
                     fprintf(stderr, " %d
chars, ignored: %s\n",
                                   sizeof lbuf - 1,
lbuf);
              } else if (lbuf[0] != '#' && n >
0)
                     count++;
       }
       rewind(envf);

       /*
        * Allocate the array of pointers.
        */
       envp = (char **)malloc(++count * sizeof
(char *));
       if (envp == (char **)-1) {
              fprintf(stderr, "%s: malloc
failed.\n", myname);
              exit(4);
       }

       /*
        * Second pass, read the strings, trim-
ming the trailing '\n's.
        * For each string, allocate space for
it, and copy it in.
        * Then link it into the pointer array.
        */
       p = envp;
       while (fgets(lbuf, sizeof lbuf, envf)
!= NULL) {
              n = strlen(lbuf) - 1;
              if (n > 0 && lbuf[n] == '\n' &&
lbuf[0] != '#') {
                     lbuf[n] = '\0';
                     q = malloc(n + 1);
                     if (q == (char *)-1) {
                            fprintf(stderr,
"%s: malloc failed.\n",
                                          myname);
                            exit(4);
                     }
                     strcpy(q, lbuf);
                     *p++ = q;
              }
       }
       /*
        * Null terminate the array, close the
file, return.
        */
       *p++ = (char *)0;
       fclose(envf);
       return envp;
}
/*
 * Debugging routine to print out the environ-
ment.
 */
env_print(FILE *f, char **ep)
{
       fprintf(f, "Environment:\n");
       while (ep && *ep)
              fprintf(f, "\t%s\n", *ep++);
```

```
        fflush(f);
}
```

Appendix B. Server source code

affine.sh

```
#
# Affine sybase engines to processors.
# We create num_proc_sets processor sets if
the -c flag is specified.
# We affine each sybase engine that is run-
ning.
#
# Must be run as root.
#
num_proc_sets=15
USAGE=”Usage: $0 [-n] [-c] [-s
processors_sets]
where
       -n: Just echo commands, don’t run them.
       -c: Create the processor sets before
affining the engines.
       -s: Number of processor sets to create.
           Default is $num_proc_sets.
“

engine_name=dataserv
do_create=0
cmd=””

while getopts “cns:” arg
do
       case $arg in

       c) do_create=1 ;;
       n) cmd=echo ;;
       s) num_proc_sets=$OPTARG ;;
       \?) echo “$USAGE”
           exit 1
           ;;
       esac
done

id | grep “uid=0(root)” >/dev/null
if [ $? -ne 0 ] ; then
       echo “You must be root to run this.”
       exit 1
fi

# Create the processor sets
if [ “$do_create” -eq 1 ] ; then
       num_created=0
       proc_id=0
       while [ $num_created -lt $num_proc_sets
]
       do
                     $cmd psrset -c $proc_id
                     num_created=‘expr
$num_created + 1‘
                     proc_id=‘expr $proc_id +
1‘
       done
fi

# Get the pids of the sybase engine processes
pids=‘ps -e | grep $engine_name | cut -c1-7‘
if [ -z “$pids” ] ; then
       echo “Sybase engines ($engine_name) are
not running”
       exit 1
fi

# Bind the pids to the processor sets
set_num=1
for pid in $pids
do
       $cmd psrset -b $set_num $pid
       set_num=‘expr $set_num + 1‘
done

$cmd psradm -i 0 1 2 3 4 5 6 8 9 10 11 12 13 15
$cmd psrset -i
$cmd psrinfo -v 0
$cmd psrinfo -v 7
$cmd psrinfo -v 14
$cmd psrinfo -v 15
```

SYB_driver.h

```
#ifndef SYB_DRIVER_H
#define SYB_DRIVER_H

#define SERVER  NULL
#define DATABASE “tpcc”
#define USER “sa”
#define MAX_ERROR       1

/* XXX: Basically we don’t need this file in
dbbench.  We only need the
        above 4 lines.   */

/* define argv’s *//* XXX: We won’t need these
*/
#defineDBNAME       1       /* XXX */
#defineFUNC_NAME2       /* XXX */
#defineNUSERS       3       /* XXX */
#defineRAMP_UP      4       /* XXX */
#defineSTDYSTATE5       /* XXX */
#defineRAMP_DOWN6       /* XXX */
#defineMAX_WAREHOUSE7       /* XXX */
#defineROLLBACK_PCT8       /* XXX */
#defineDELTA        9       /* XXX */
```

```
#defineINTERVAL25   /* XXX: Total interval of
buckets, in sec */
#defineUNIT         .5     /* XXX: Time period
of each bucket */
#defineHIST_MAX50   /* XXX: Num of histogram
buckets = INTERVAL/UNIT */
#defineBUCKET       500    /* XXX: Division
factor for response time */
#defineMATCH        0      /* XXX: used as =’s
with string ops */
#defineMILLI        1000   /* XXX: Conversion
from seconds to milliseconds */
typedef struct  CNTRL           /* XXX: We
won’t need these */
{
int tran_count;
int deadlock_cnt;
int res_time;
double tot_time;
int min_res;
int max_res;
 int not_done;
double  tran_sqr;
int tran_2sec;
} CONTROL;

extern LOGINREC *login;
extern DBPROCESS *dbproc;

extern int scale, nusers; /* XXX */
extern DBINT run_id; /* XXX */
extern char* db_name; /* XXX */
extern int rampup, stdystate, rampdown; /* XXX
*/
extern int end_rampup, end_stdystate,
end_rampdown; /* XXX */
extern char func_name[32]; /* XXX */
extern CONTROL status[11]; /* XXX */
extern unsigned long avg_delay;
extern unsigned long last_resp;
extern int delta;

double drand();
void sel1(); /* Function to run “select 1”
test */
/* int err_handler();  XXX: moved to
SYB_tpcc.h */
/* int msg_handler();  XXX: moved to
SYB_tpcc.h */
void     init_time();
unsigned long delay();

#endif SYB_DRIVER_H
```

SYB_error.c

```
#include <stdio.h>
#include <sybfront.h>
#include <sybdb.h>
#include <syberror.h>
#include “SYB_tpcc.h”

/* message numbers that we don’t want to deal
with */
#defineCONTEXT_SET5701
#defineLANGUAGE_SET5703
#defineCHARACTER_SET5704
#defineABORT_ERROR6104

/* Tuxedo include files */
#include “atmi.h”
#include “userlog.h”

int
err_handler(dbproc, severity, errno, oserr)
DBPROCESS *dbproc;
int severity;
int errno;
int oserr;
{ userlog(“DB-LIBRARY Error %d:”, errno);
display_xction(dberrstr(errno));
if (oserr != DBNOERR)
{
userlog(“O/S Error: “);
display_xction(dboserrstr(oserr));
}

/* exit on any error */
exit(-100);
}



int
msg_handler(dbproc,msgno,msgstate,sever-
ity,msgtext,servername,procname,line)
DBPROCESS*dbproc;
int msgno;
int msgstate;
int severity;
char *msgtext;
char *servername;
char *procname;
int  line;
{
/* changing database messages */
if (msgno == CONTEXT_SET ||
msgno == LANGUAGE_SET ||
msgno == CHARACTER_SET)
```

```
return(SUCCEED);

if (msgno == ABORT_ERROR)
return(SUCCEED);

/* Is this a deadlock message */
if (msgno == 1205)
{/* Set the deadlock indicator */
/* *((DBBOOL *) dbgetuserdata(dbproc)) = TRUE;
*/
display_xction(msgtext);
deadlock = 1;
return(SUCCEED);
}
else {
userlog("msg no %d - %s\n", msgno, msgtext);
userlog("xact_type: %d deadlock= %d\n",
xact_type, deadlock);
if (msgno == 0)
return(SUCCEED);
else
return(FAIL);
}
}
```

SYB_rpc.c

```
#include <stdio.h>
#include <sys/types.h>
#include <sys/time.h>
#include <time.h>
#include <sybfront.h>
#include <sybdb.h>

#include "SYB_tpcc.h"
#include "SYB_driver.h"
#include "SYB_rpc_var.c"

/* Tuxedo include files */
#include "atmi.h"
#include "userlog.h"
#include "fml.h"


#include "Usysflds.h"

#include "tpcc_tux_forms.h"
#include "tpcc_tux_forms_var.c"

/* Date conversion routines */
DBDATETIME syb_datetime;
DBDATETIME syb_date;

/* Added to count invalid transactions for V
3.3 */
int invalid_xact;

/* Added to switch between real and dummy
stored procs */
extern int dummy_sp;

void
sybdate2datetime (DBDATETIME * sybdate, char
*datetime)
{
  DBDATEREC daterec;
  dbdatecrack (NULL, &daterec, sybdate);

  if (daterec.dateyear < 1900)
       strcpy(datetime, "                  ");
  else

  sprintf (datetime, "%02d-%02d-%04d
%02d:%02d:%02d",
           daterec.datedmonth,
           daterec.datemonth + 1,
           daterec.dateyear,
           daterec.datehour, daterec.datem-
inute, daterec.datesecond);
}

void
sybdate2date (DBDATETIME * sybdate, char
*date)
{
  DBDATEREC daterec;

  dbdatecrack (NULL, &daterec, sybdate);

  if (daterec.dateyear < 1900)
       strcpy(date, "          ");
  else

  sprintf (date, "%02d-%02d-%04d",
           daterec.datedmonth, daterec.date-
month + 1, daterec.dateyear);
}

void
new_order_rpc ()
{
  int try;

  for (try = 0; try < MaxTries; try++)
    {
      if (try > 0)
       display_xction ("Repeating NO");

      deadlock = 0; invalid_xact = 0;
      if (new_order_body () != TRUE)
```

```
       break;;
      dbcancel (dbproc);
      sleep_before_retry ();
    }
  if (try >= MaxTries)
    display_xction (“Failed”);
}


int
new_order_body ()
{
  int i, j;
  DBINT retcode;
  struct items_inf *cur_ip;/* Pointer to cur-
rent item */

  deadlock = 0;
  if (o_all_local)
    {
      if (!dummy_sp)
       dbrpcinit (dbproc, “neworder_local”,
0);
      else
       dbrpcinit (dbproc, “dummy_no_local”,
0);
    }
  else
    {
      if (!dummy_sp)
       dbrpcinit (dbproc, “neworder_remote”,
0);
      else
       dbrpcinit (dbproc, “dummy_no_remote”,
0);
    }
  dbrpcparam (dbproc, NULL, 0, SYBINT2, -1, -
1, (BYTE *)&global_w_id);
  dbrpcparam (dbproc, NULL, 0, SYBINT1, -1, -
1, (BYTE *)&global_d_id);
  dbrpcparam (dbproc, NULL, 0, SYBINT4, -1, -
1, (BYTE *)&c_id);
  dbrpcparam (dbproc, NULL, 0, SYBINT2, -1, -
1, (BYTE *)&o_ol_cnt);

  for (i = 0; i < (int) o_ol_cnt; i++)
    {
     dbrpcparam (dbproc, NULL, 0, SYBINT4, -1,
-1, (BYTE *)&ol[i].i_id);
      if (!o_all_local) {
       dbrpcparam (dbproc, NULL, 0, SYBINT2, -
1, -1,
              (BYTE *)&ol[i].supply_w_id);
      }
     dbrpcparam (dbproc, NULL, 0, SYBINT2, -1,
-1, (BYTE *)&ol[i].quantity);

    }

  if (dbrpcsend (dbproc) != SUCCEED)
    return TRUE;
  if (dbsqlok (dbproc) != SUCCEED)
    return TRUE;

  total_amount = 0;
  for (i = 0; i < (int) o_ol_cnt; i++)
    {


      if (dbresults (dbproc) != SUCCEED ||
deadlock)
       return TRUE;
      else
       {
         dbbind (dbproc, 1, NTBSTRINGBIND,
sizeof (i_name), (BYTE *)i_name);
        dbbind (dbproc, 2, SMALLBIND, 0, (BYTE
*)&s_quantity);
         dbbind (dbproc, 3, FLT8BIND, 0, (BYTE
*)&i_price);
         dbbind (dbproc, 4, NTBSTRINGBIND,
sizeof (b_g), (BYTE *)b_g);
         if (dbnextrow (dbproc) != REG_ROW)
           return TRUE;

          if (dbretstatus (dbproc) == -6)
          {
            display_xction (“invalid_xact”);
                          neworder->status[0]
= ‘\0’;
                             invalid_xact++;
           return FALSE;
          }

           i_price /= 100.0;          /* cents
=> dollars */

         if (*i_name == ‘\0’)
           {
             /*display_xction(“Invalid item
in”); */
             /*bad_items++; */

             strcpy (neworder->status, “Item
number is not valid”);
             commit_flag = FALSE;
           }
         if (dbcanquery (dbproc) != SUCCEED ||
deadlock)
           return TRUE;
```

```
      }
      if (dbhasretstat (dbproc))
      {
        if ((retcode = dbretstatus (dbproc))
== -3)
          {
            deadlock = 1;
            display_xction (“Deadlock vic-
tim:”);
          }
        else if (retcode < 0)
          {
            userlog (“Unknown return status
%d:”, retcode);
            display_xction (“”);
          }
        return TRUE;
      }

      cur_ip = &neworder->n_items[i];
      strcpy (cur_ip->i_name, i_name);
      cur_ip->i_price = i_price;
      cur_ip->s_quantity = s_quantity;
      strcpy (cur_ip->brand, b_g);
      ol_amount = i_price * ol[i].quantity;
      cur_ip->ol_amount = ol_amount;

      total_amount += ol_amount;
    }


  if (dbresults (dbproc) != SUCCEED || dead-
lock)
    {
      /*
         userlog(“NEWORDER failed\n”);
       */
      return TRUE;
    }

  dbbind (dbproc, 1, REALBIND, 0, (BYTE
*)&w_tax);
  dbbind (dbproc, 2, REALBIND, 0, (BYTE
*)&d_tax);
  dbbind (dbproc, 3, INTBIND, 0, (BYTE
*)&o_id);
  dbbind (dbproc, 4, NTBSTRINGBIND, sizeof
(c_last), (BYTE *)c_last);
  dbbind (dbproc, 5, REALBIND, 0, (BYTE
*)&c_discount);
  dbbind (dbproc, 6, NTBSTRINGBIND, sizeof
(c_credit), (BYTE *)c_credit);
  dbbind (dbproc, 7, DATETIMEBIND, 0, (BYTE
*)&syb_datetime);
  if (dbnextrow (dbproc) != REG_ROW || dead-
lock)
    return TRUE;
  if (dbcanquery (dbproc) != SUCCEED || dead-
lock)
    return TRUE;
  sybdate2datetime (&syb_datetime, o_entry_d);

  neworder->w_tax = w_tax * 100.;/* fraction
to % */
  neworder->d_tax = d_tax * 100.;/* fraction
to % */
  neworder->o_id = o_id;
  strcpy (neworder->c_last, c_last);
  neworder->c_discount = c_discount * 100.;/*
fraction to % */
  strcpy (neworder->c_credit, c_credit);
  strcpy (neworder->o_entry_d, o_entry_d);
  total_amount *= (1.-c_discount) *
(1+w_tax+d_tax);


  /*
     userlog(“finishing\n”);
   */


  /* TPC-C V3.3 Error Checking for invalid
input data */


  if (dbretstatus (dbproc) == -6)
    {
      display_xction (“invalid_xact”);
      neworder->status[0] = ‘\0’;
      invalid_xact++;
    }



  return FALSE;
}

void
payment_byid_rpc ()
{
  int try;

  for (try = 0; try < MaxTries; try++)
    {
      if (try > 0)
       display_xction (“Repeating”);

      if (payment_byid_begin () == TRUE)
       {
         dbcancel (dbproc);
```

```
        sleep_before_retry ();
        continue;
       }
      if (payment_end () == TRUE)
       {
        dbcancel (dbproc);
        sleep_before_retry ();
        continue;
       }
      break;
    }

  if (try >= MaxTries)
    {
      display_xction (“MaxTries Failed”);
      tpreturn (TPFAIL, 0, (char *) paym_rqst-
>data, paym_rqst->len, 0);
    }
}

int
payment_byid_begin ()
{
  double      h_amount_cents = h_amount *
100.0;      /* dollars => cents */
  deadlock = 0; invalid_xact = 0;
  if (!dummy_sp)
    dbrpcinit (dbproc, “payment_byid”, 0);
  else
    dbrpcinit (dbproc, “dummy_pay_byid”, 0);
  dbrpcparam (dbproc, NULL, 0, SYBINT2, -1, -
1, (BYTE *)&global_w_id);
  dbrpcparam (dbproc, NULL, 0, SYBINT2, -1, -
1, (BYTE *)&c_w_id);
  dbrpcparam (dbproc, NULL, 0, SYBFLT8, -1, -
1, (BYTE *)&h_amount_cents);
  dbrpcparam (dbproc, NULL, 0, SYBINT1, -1, -
1, (BYTE *)&global_d_id);
  dbrpcparam (dbproc, NULL, 0, SYBINT1, -1, -
1, (BYTE *)&c_d_id);
  dbrpcparam (dbproc, NULL, 0, SYBINT4, -1, -
1, (BYTE *)&c_id);
  return (dbrpcsend (dbproc) == SUCCEED ?
FALSE : TRUE);
}

void
payment_byname_rpc ()
{
  int try;
  for (try = 0; try < MaxTries; try++)
    {
      if (try > 0)
       display_xction (“Repeating”);
      if (payment_byname_begin () == TRUE)
       {
        dbcancel (dbproc);
        sleep_before_retry ();
        continue;
       }
      if (payment_end () == TRUE)
       {
        dbcancel (dbproc);
        sleep_before_retry ();
        continue;
       }
      break;
    }
  if (try >= MaxTries)
    {
      display_xction (“MaxTries Failed”);
      tpreturn (TPFAIL, 0, (char *) paym_rqst-
>data, paym_rqst->len, 0);
    }
}

int
payment_byname_begin ()
{
  double      h_amount_cents = h_amount *
100.0;      /* dollars => cents */
  deadlock = 0; invalid_xact = 0;
  if (!dummy_sp)
    dbrpcinit (dbproc, “payment_byname”, 0);
  else
    dbrpcinit (dbproc, “dummy_pay_byname”, 0);
  dbrpcparam (dbproc, NULL, 0, SYBINT2, -1, -
1, (BYTE *)&global_w_id);
  dbrpcparam (dbproc, NULL, 0, SYBINT2, -1, -
1, (BYTE *)&c_w_id);
  dbrpcparam (dbproc, NULL, 0, SYBFLT8, -1, -
1, (BYTE *)&h_amount_cents);
  dbrpcparam (dbproc, NULL, 0, SYBINT1, -1, -
1, (BYTE *)&global_d_id);
  dbrpcparam (dbproc, NULL, 0, SYBINT1, -1, -
1, (BYTE *)&c_d_id);
  dbrpcparam (dbproc, NULL, 0, SYBCHAR, -1,
strlen (c_last), (BYTE *)c_last);
  return (dbrpcsend (dbproc) == SUCCEED ?
FALSE : TRUE);
}

int
payment_end ()
{
  if (dbsqlok (dbproc) != SUCCEED)
    return TRUE;
  if (dbresults (dbproc) != SUCCEED || dead-
lock)
    return TRUE;
```

```
  else


    {


      dbbind (dbproc, 1, INTBIND, 0, (BYTE
*)&c_id);
      dbbind (dbproc, 2, NTBSTRINGBIND, sizeof
(c_last), (BYTE *)c_last);
     dbbind (dbproc, 3, DATETIMEBIND, 0, (BYTE
*)&syb_datetime);
      dbbind (dbproc, 4, NTBSTRINGBIND, sizeof
(w_street_1), (BYTE *)w_street_1);
      dbbind (dbproc, 5, NTBSTRINGBIND, sizeof
(w_street_2), (BYTE *)w_street_2);
      dbbind (dbproc, 6, NTBSTRINGBIND, sizeof
(w_city), (BYTE *)w_city);
      dbbind (dbproc, 7, NTBSTRINGBIND, sizeof
(w_state), (BYTE *)w_state);
      dbbind (dbproc, 8, NTBSTRINGBIND, sizeof
(w_zip), (BYTE *)w_zip);

      dbbind (dbproc, 9, NTBSTRINGBIND, sizeof
(d_street_1), (BYTE *)d_street_1);
     dbbind (dbproc, 10, NTBSTRINGBIND, sizeof
(d_street_2), (BYTE *)d_street_2);
     dbbind (dbproc, 11, NTBSTRINGBIND, sizeof
(d_city), (BYTE *)d_city);
     dbbind (dbproc, 12, NTBSTRINGBIND, sizeof
(d_state), (BYTE *)d_state);
     dbbind (dbproc, 13, NTBSTRINGBIND, sizeof
(d_zip), (BYTE *)d_zip);
     dbbind (dbproc, 14, NTBSTRINGBIND, sizeof
(c_first), (BYTE *)c_first);
     dbbind (dbproc, 15, NTBSTRINGBIND, sizeof
(c_middle), (BYTE *)c_middle);
     dbbind (dbproc, 16, NTBSTRINGBIND, sizeof
(c_street_1), (BYTE *)c_street_1);
     dbbind (dbproc, 17, NTBSTRINGBIND, sizeof
(c_street_2), (BYTE *)c_street_2);
     dbbind (dbproc, 18, NTBSTRINGBIND, sizeof
(c_city), (BYTE *)c_city);
     dbbind (dbproc, 19, NTBSTRINGBIND, sizeof
(c_state), (BYTE *)c_state);
     dbbind (dbproc, 20, NTBSTRINGBIND, sizeof
(c_zip), (BYTE *)c_zip);
     dbbind (dbproc, 21, NTBSTRINGBIND, sizeof
(c_phone), (BYTE *)c_phone);
      dbbind (dbproc, 22, DATETIMEBIND, 0,
(BYTE *)&syb_date);
     dbbind (dbproc, 23, NTBSTRINGBIND, sizeof
(c_credit), (BYTE *)c_credit);
      dbbind (dbproc, 24, FLT8BIND, 0, (BYTE
*)&c_credit_lim);
      dbbind (dbproc, 25, REALBIND, 0, (BYTE
*)&c_discount);
      dbbind (dbproc, 26, FLT8BIND, 0, (BYTE
*)&c_balance);
     dbbind (dbproc, 27, NTBSTRINGBIND, sizeof
(c_data), (BYTE *)c_data);
      if (dbnextrow (dbproc) != REG_ROW ||
deadlock)
       return TRUE;
      if (dbcanquery (dbproc) != SUCCEED ||
deadlock)
       return TRUE;
      sybdate2datetime (&syb_datetime,
h_date);
      sybdate2date (&syb_date, c_since);
      c_credit_lim /= 100.0;          /* cents
=> dollars */
      c_balance /= 100.0;             /* cents
=> dollars */
      c_discount *= 100.0;      /* fraction to
% */

      /*
         if (*c_last == '\0')
         {
        strcpy(c_data, "   Execution Status :
Invalid Transaction......   ");
         tpreturn(TPFAIL, 0, (char *)paym_rqst-
>data, paym_rqst->len, 0);

         }
       */
    }
  /* TPC-C V3.3 Error Checking for invalid
input data */
  if (dbretstatus (dbproc) == -6)
    {
      display_xction ("invalid_xact");
      invalid_xact++;
    }


  return FALSE;
}

void
order_status_byid_rpc ()
{
  int try;

  for (try = 0; try < MaxTries; try++)
    {
      if (try > 0)
       display_xction ("Repeating");
      if (order_status_byid_begin () == TRUE)
```

```
      {
        dbcancel (dbproc);
        if (invalid_xact)
           break;
        sleep_before_retry ();
        continue;
      }
     if (order_status_end () == TRUE)
      {
        dbcancel (dbproc);
        if (invalid_xact)
           break;
        sleep_before_retry ();
        continue;
      }
     break;
    }

  if (try >= MaxTries)
    {
      display_xction (“MaxTries Failed”);
      tpreturn (TPFAIL, 0, (char *) ords_rqst-
>data, ords_rqst->len, 0);
    }
}

int
order_status_byid_begin ()
{
  deadlock = 0; invalid_xact = 0;

  dbrpcinit (dbproc, “order_status_byid”, 0);
  dbrpcparam (dbproc, NULL, 0, SYBINT2, -1, -
1, (BYTE *)&c_w_id);
  dbrpcparam (dbproc, NULL, 0, SYBINT1, -1, -
1, (BYTE *)&c_d_id);
  dbrpcparam (dbproc, NULL, 0, SYBINT4, -1, -
1, (BYTE *)&c_id);
  return (dbrpcsend (dbproc) == SUCCEED ?
FALSE : TRUE);
}

void
order_status_byname_rpc ()
{
  int try;

  for (try = 0; try < MaxTries; try++)
    {
      if (try > 0)
       display_xction (“Repeating”);

     if (order_status_byname_begin () == TRUE)
      {
        dbcancel (dbproc);
        if (invalid_xact)
           break;
        sleep_before_retry ();
        continue;
       }
      if (order_status_end () == TRUE)
       {
        dbcancel (dbproc);
        if (invalid_xact)
           break;
        sleep_before_retry ();
        continue;
       }
      break;
    }

  if (try >= MaxTries)
    {
      display_xction (“MaxTries Failed”);
      tpreturn (TPFAIL, 0, (char *) ords_rqst-
>data, ords_rqst->len, 0);
    }
}

int
order_status_byname_begin ()
{
  deadlock = 0; invalid_xact = 0;
  dbrpcinit (dbproc, “order_status_byname”,
0);
  dbrpcparam (dbproc, NULL, 0, SYBINT2, -1, -
1, (BYTE *)&c_w_id);
  dbrpcparam (dbproc, NULL, 0, SYBINT1, -1, -
1, (BYTE *)&c_d_id);
  dbrpcparam (dbproc, NULL, 0, SYBCHAR, -1,
strlen (c_last), (BYTE *)c_last);
  return (dbrpcsend (dbproc) == SUCCEED ?
FALSE : TRUE);
}

int
order_status_end ()
{
  int count;
  int rc;

  if (dbsqlok (dbproc) != SUCCEED) {
    return TRUE;
  }


 if (((rc = dbresults (dbproc)) != SUCCEED) ||
deadlock) {
    userlog(“%d: dbresults rc=%d, dead-
lock=%d\n”, __LINE__, rc, deadlock);
```

```
    return TRUE;
  }
  else
    {

      /* V 3.3 Error Checking
         ** checks whether current command
returned any rows
       */
      if (dbrows (dbproc) != SUCCEED)
       {
      display_xction (“invalid_xact”);
        invalid_xact++;
       }

      dbbind (dbproc, 1, SMALLBIND, 0, (BYTE
*)&ol_supply_w_id);
      dbbind (dbproc, 2, INTBIND, 0, (BYTE
*)&ol_i_id);
      dbbind (dbproc, 3, SMALLBIND, 0, (BYTE
*)&ol_quantity);
      dbbind (dbproc, 4, FLT8BIND, 0, (BYTE
*)&ol_amount);
     dbbind (dbproc, 5, DATETIMEBIND, 0, (BYTE
*)&syb_date);

      count = 0;
      while ((code = dbnextrow (dbproc)) ==
REG_ROW && !deadlock)
       {
         ol_amount /= 100.0;         /* cents
=> dollars */
         /*
            ** Print order_line information on
RTE
          */
         ordstat-
>o_items[count].ol_supply_w_id =
ol_supply_w_id;
         ordstat->o_items[count].ol_i_id =
ol_i_id;
        ordstat->o_items[count].ol_quantity =
ol_quantity;
         ordstat->o_items[count].ol_amount =
ol_amount;


         sybdate2date (&syb_date,
ol_delivery_d);
         strcpy (ordstat-
>o_items[count].ol_delivery_d, ol_delivery_d);
         count++;
       }
      ordstat->item_cnt = count;

      /*
         userlog(“count= %d \n”, count);
       */

      if (code != NO_MORE_ROWS || deadlock)
       return TRUE;
    }

  if (dbresults (dbproc) != SUCCEED || dead-
lock)
    return TRUE;
  else
    {

      /* V 3.3 Error Checking
         ** checks whether current command
returned any rows
       */
      if (dbrows (dbproc) != SUCCEED)
       {
      display_xction (“invalid_xact”);
        invalid_xact++;
       }


      dbbind (dbproc, 1, INTBIND, 0, (BYTE
*)&c_id);
      dbbind (dbproc, 2, NTBSTRINGBIND, sizeof
(c_last), (BYTE *)c_last);
      dbbind (dbproc, 3, NTBSTRINGBIND, sizeof
(c_first), (BYTE *)c_first);
      dbbind (dbproc, 4, NTBSTRINGBIND, sizeof
(c_middle), (BYTE *)c_middle);
      dbbind (dbproc, 5, FLT8BIND, 0, (BYTE
*)&c_balance);
      dbbind (dbproc, 6, INTBIND, 0, (BYTE
*)&o_id);
     dbbind (dbproc, 7, DATETIMEBIND, 0, (BYTE
*)&syb_datetime);
      dbbind (dbproc, 8, SMALLBIND, 0, (BYTE
*)&o_carrier_id);
      if (((rc = dbnextrow (dbproc)) !=
REG_ROW) || deadlock) {
        userlog(“%d: dbnextrow rc=%d, dead-
lock=%d\n”, __LINE__, rc, deadlock);
       return TRUE;
      }
      if (((rc = dbcanquery (dbproc)) != SUC-
CEED) || deadlock) {
        userlog(“%d: rc=%d, deadlock=%d\n”,
__LINE__, rc, deadlock);
       return TRUE;
      }
      sybdate2datetime (&syb_datetime,
o_entry_d);
```

```
      strcpy (ordstat->c_first, c_first);
      strcpy (ordstat->c_middle, c_middle);
      strcpy (ordstat->c_last, c_last);
      ordstat->c_balance = c_balance / 100.;
      ordstat->o_id = (int) o_id;
      strcpy (ordstat->o_entry_d, o_entry_d);
      ordstat->o_carrier_id = o_carrier_id;

    }

  return FALSE;
}

void
delivery_rpc ()
{
  /*
     Called by delivery processes.
   */
  int try;


  global_d_id = 1;
  for (try = 0; try < MaxTries; try++)
    {
      if (try > 0)
       display_xction (“Repeating”);

      if (delivery_body () == TRUE)
       {
         dbcancel (dbproc);
         sleep_before_retry ();
         continue;
       }
      break;
    }
  if (try >= MaxTries)
    {
      display_xction (“MaxTries Failed”);

      fwrite (outbuf, strlen (outbuf), 1,
delfile);
      fflush (delfile);
      tpreturn (TPFAIL, 0, (char *) del_rqst-
>data, del_rqst->len, 0);
    }
}

int
delivery_body ()
{
  extern int Delivery_skipped;/* buddy */

  deadlock = 0; invalid_xact = 0;
  if (!dummy_sp)
    dbrpcinit (dbproc, “delivery”, 0);
  else
    dbrpcinit (dbproc, “dummy_del”, 0);
  dbrpcparam (dbproc, NULL, 0, SYBINT2, -1, -
1, (BYTE *)&global_w_id);
  dbrpcparam (dbproc, NULL, 0, SYBINT2, -1, -
1, (BYTE *)&o_carrier_id);
  dbrpcparam (dbproc, NULL, 0, SYBINT1, -1, -
1, (BYTE *)&global_d_id);
  if (dbrpcsend (dbproc) != SUCCEED)
    return TRUE;
  if (dbsqlok (dbproc) != SUCCEED)
    return TRUE;

  for (; global_d_id <= 10; global_d_id++)
    {
      if (dbresults (dbproc) != SUCCEED ||
deadlock)
       return TRUE;

      dbbind (dbproc, 1, INTBIND, 0, (BYTE
*)&o_id);
      if (dbnextrow (dbproc) != REG_ROW ||
deadlock)
       return TRUE;

      if (o_id == NULL) {
       Delivery_skipped = 1 ;
      }
      else {
       Delivery_skipped = 0 ;
      }


      if (dbcanquery (dbproc) != SUCCEED ||
deadlock)
       return TRUE;
     if (dbhasretstat (dbproc) && dbretstatus
(dbproc) != 0)
       return TRUE;
    }
  return FALSE;
}

void
stock_level_rpc ()
{
  int try;


  for (try = 0; try < MaxTries; try++)
    {
      if (try > 0)
       display_xction (“Repeating”);
```

```
      if (stock_level_body () == TRUE)
       {
         dbcancel (dbproc);
         if (invalid_xact)
            break;
         sleep_before_retry ();
         continue;
       }
      break;
    }

  if (try >= MaxTries)
    {
      display_xction (“MaxTries Failed”);
     tpreturn (TPFAIL, 0, (char *) stock_rqst-
>data, stock_rqst->len, 0);
    }
}

int
stock_level_body ()
{
  int found, iid, uniq[500];
  int i, j, count;

  deadlock = 0; invalid_xact = 0;
  if (!dummy_sp)
    dbrpcinit (dbproc, “stock_level”, 0);
  else
    dbrpcinit (dbproc, “dummy_stock”, 0);
  dbrpcparam (dbproc, NULL, 0, SYBINT2, -1, -
1, (BYTE *)&global_w_id);
  dbrpcparam (dbproc, NULL, 0, SYBINT1, -1, -
1, (BYTE *)&global_d_id);
  dbrpcparam (dbproc, NULL, 0, SYBINT2, -1, -
1, (BYTE *)&threshold);
  if (dbrpcsend (dbproc) != SUCCEED)
    return TRUE;
  if (dbsqlok (dbproc) != SUCCEED)
    return TRUE;

  if (dbresults (dbproc) != SUCCEED || dead-
lock)
    return TRUE;
  dbbind (dbproc, 1, INTBIND, 0, (BYTE
*)&iid);


  /* sort for distinct(s_i_id) */
  count = 0;
  while (dbnextrow (dbproc) == REG_ROW &&
!deadlock)
    {
      found = 0;
      for (j = 0; j < count; j++)
       {
         if (iid == uniq[j])
           {
             found = 1;
             break;
           }
       }

      if (found == 0)
       {
         if (count >= 500)
           display_xction (“Too many rows
returned by”);
         else
           uniq[count++] = iid;
       }
    }
  if (deadlock)
    return TRUE;
  if (dbcanquery (dbproc) != SUCCEED || dead-
lock)
    return TRUE;

  low_count = count;
  stocklevel->low_stock = low_count;

  /* TPC-C V3.3 Error Checking for invalid
input data */


  if (dbretstatus (dbproc) == -6)
    {
      display_xction (“invalid_xact”);
      stocklevel->low_stock = -1;
      invalid_xact++;
    }


  return FALSE;
}

void
ins_rpc ()
{
  dbfcmd (dbproc, “insert into foo values(%d,
‘kjhkjhkjhkjhkjh’)”,
          global_w_id);
  dbsqlexec (dbproc);
  dbresults (dbproc);
}

void
sleep_before_retry ()
{
  sleep (1);
```

```
}

void
display_xction (msg)
     char *msg;
{
  int i;
  userlog (“%s %s “, msg,
func_array[xact_type].name);

  switch (xact_type)
    {

    case XACT_NEWO:/* new_order */
      userlog (“w=%d, d=%d, c=%d, %d lines:
\n[“,
              global_w_id, global_d_id, c_id,
o_ol_cnt);
      for (i = 0; i < (int) o_ol_cnt; i++)
       userlog (“ %d”, ol[i].i_id);
      userlog (“]\n”);
      break;
    case XACT_PAYM_ID:/* payment_byid */
      userlog (“w=%d/%d, d=%d/%d, c=%d\n”,
            global_w_id, c_w_id, global_d_id,
c_d_id, c_id);
      break;

    case XACT_PAYM_NAME:/* payment_byname */
      userlog (“w=%d/%d, d=%d/%d, l=%s\n”,
            global_w_id, c_w_id, global_d_id,
c_d_id, c_last);
      break;

    case XACT_ORDS_ID:/* order_status_byid */
     userlog (“cw=%d, cd=%d, c=%d\n”, c_w_id,
c_d_id, c_id);
      break;

    case XACT_ORDS_NAME:/* order_status_byname
*/
     userlog (“cw=%d, cd=%d, l=%s\n”, c_w_id,
c_d_id, c_last);
      break;

    case XACT_DEL:/* delivery_qu */
      userlog (“w=%d, carrier=%d\n”,
global_w_id, o_carrier_id);
      break;

    case XACT_STOCK:/* stock level */
      userlog (“w=%d, d=%d, th=%d\n”,
global_w_id, global_d_id, threshold);
      break;

    case XACT_BKEND:/* delivery */
      userlog (“w=%d, d=%d, carrier=%d,
tx_count=%d\n”,
             global_w_id, global_d_id,
o_carrier_id, tx_count);
      break;

    default:
      userlog (“Unknown xact_type = %d\n”,
xact_type);
    }
}
```

SYB_rpc_var.c

```
/*
**  This file contains the declaration of the
shared variables in rpc.c.
**  Shared variables are used in passing argu-
ments to TPC-C transactions.
*/

XCTION func_array[XCTION_COUNT+1] =
    {
{“new_order”},
{“payment_byid”},
{“payment_byname”},
{“order_status_byid”},
{“order_status_byname”},
{“delivery_qu”},
{“stock_level”},
{“delivery”},
{“NULL”}
    };
int rollback_pct;
int lines_per_call = 15;
char b_g[2];
DBFLT8 total_amount;
DBTINYINT commit_flag;
int xact_type, prev_xact_type = -9999;
int deadlock;
int bad_items;
int max_ware;
char *db_name = “tpcc”;
RETCODE code;
DBSMALLINT global_w_id;
DBTINYINT global_d_id;
ORDER_LINE ol[15];/* XXX: should be 16, or 15
?? */

/*
** Variables for the customer table
*/
```

```
DBINT c_id;
DBTINYINT c_d_id;
DBSMALLINT c_w_id;
char c_first[17];
char c_middle[3];
char c_last[17];
char c_street_1[21];
char c_street_2[21];
char c_city[21];
char c_state[3];
char c_zip[10];
char c_phone[17];
char c_since[31];
char c_credit[3];
DBFLT8 c_credit_lim;
DBREAL c_discount;
DBFLT8 c_balance;
char c_data[201];

/*
** Variables for warehouse
*/

char w_name[11];
char w_street_1[21];
char w_street_2[21];
char w_city[21];
char w_state[3];
char w_zip[10];
DBREAL w_tax;

/*
** Variables for district
*/

DBTINYINT       d_id;
DBSMALLINT d_w_id;
char    d_name[11];
char    d_street_1[21];
char    d_street_2[21];
char    d_city[21];
char    d_state[3];
char    d_zip[10];
DBREAL  d_tax;

/*
** Variables for item table
*/

int i_id;
DBFLT8 i_price;
char i_name[25];

/*
** Variables for the stock table
*/

DBSMALLINT s_quantity;
DBSMALLINT threshold;
DBINT low_count;
char s_dist[25];

/*
** Variables for order table
*/

int o_id;


     DBTINYINT o_d_id;
     DBSMALLINT o_w_id;
     DBSMALLINT o_c_id;
     char o_entry_d[31];
     DBSMALLINT o_carrier_id;
     DBSMALLINT o_ol_cnt, o_ol_now, o_ol_done;
     DBTINYINT o_all_local;
     /*
     ** Variables for order_line
     */
     int ol_o_id;
     DBTINYINT ol_d_id;
     DBSMALLINT ol_w_id;
     DBSMALLINT ol_number;
     DBINT ol_i_id;
     DBSMALLINT ol_supply_w_id;
     char ol_delivery_d[31];
     DBSMALLINT ol_quantity;
     DBFLT8 ol_amount;

     /*
     ** Variables for new_order tble
     */
     int no_o_id;
     DBTINYINT no_d_id;
     DBSMALLINT no_w_id;

     /*
     ** Variables for history table
     */

     DBFLT8 h_amount;
     char h_date[20];
```

SYB_tpcc.h

```
#ifndef SYB_TPCC_H
#define SYB_TPCC_H

#defineMAXDIST 10
```

```
#define MaxTries 5
#define smaller(x,y) (x<y ? x : y)
#define XCTION_COUNT 8

/*
 * XXX: For error handlers
 */
int     err_handler();
int    msg_handler();

#define XACT_NEWO0
#define XACT_PAYM_ID1
#define XACT_PAYM_NAME2
#define XACT_ORDS_ID3
#define XACT_ORDS_NAME4
#define XACT_DEL5
#define XACT_STOCK6
#define XACT_BKEND7
/*
** Structure for each line of an order
*/
typedef struct Order_Line {
int     i_id;
DBSMALLINT      supply_w_id;
DBSMALLINT  quantity;
DBSMALLINT  s_quantity;
DBFLT8  i_price;
DBFLT8  ol_amount;
char    i_name[26];
int increment;
} ORDER_LINE;

/*
**  Define the TPC-C functions
*/




          void gen_new_order();void
new_order_rpc();
          void gen_payment_byid();void
payment_byid_rpc();
          void gen_payment_byname();void
payment_bynamme_rpc();
          void gen_order_status_byid(); void
order_status_byid_rpc();
         void gen_order_status_byname();void
order_status_byname_rpc();
          void gen_delivery();void
delivery_qu_add();
          void gen_stock_level();void
stock_level_rpc();
          void delivery_qu_del();void
delivery_rpc();
          void delivery_qu_connect();
          void display_xction();
          void sleep_before_retry ();
          void display_xction();
          void pick_xact_type();

          typedef struct Xction {
          char name[30];
          } XCTION;

          extern XCTION
func_array[XCTION_COUNT+1];

          extern int rollback_pct;
          extern int lines_per_call;
          extern charb_g[2];
          extern DBFLT8 total_amount;
          extern DBTINYINT commit_flag;
          extern int xact_type,
prev_xact_type;
          extern int deadlock;
          extern int bad_items;
          extern int max_ware;
          extern RETCODE code;

         /* set to global so that wont clash
with tpcc_client.h */

          extern DBSMALLINT global_w_id;
          extern DBTINYINT global_d_id;
          extern ORDER_LINE ol[15];  /* XXX:
should be 16, or 15 ??? */

          /*
         ** Variables for the customer table
          */

          extern DBINT c_id;
          extern DBTINYINT c_d_id;
          extern DBSMALLINT c_w_id;
          extern char c_first[17];
          extern char c_middle[3];
          extern char c_last[17];
          extern char c_street_1[21];
          extern char c_street_2[21];
          extern char c_city[21];
          extern char c_state[3];
          extern char c_zip[10];
          extern char c_phone[17];
          extern char c_since[31];
          extern char c_credit[3];
          extern DBFLT8 c_credit_lim;
          extern DBREAL c_discount;
          extern DBFLT8 c_balance;
          extern char c_data[201];
```

```
        /*
        ** Variables for warehouse
        */
        extern char w_name[11];
        extern char w_street_1[21];
        extern char w_street_2[21];
        extern char w_city[21];
        extern char w_state[3];
        extern char w_zip[10];
        extern DBREAL w_tax;
        /*
        ** Variables for district
        */

        extern DBTINYINT global_d_id;
        extern DBSMALLINT d_w_id;
        extern char    d_name[11];
        extern char    d_street_1[21];
        extern char    d_street_2[21];
        extern char    d_city[21];
        extern char    d_state[3];
        extern char    d_zip[10];
        extern DBREAL  d_tax;

        /*
        ** Variables for item table
        */

        extern int i_id;
        extern DBFLT8 i_price;
        extern char i_name[25];
        /*
        ** Variables for the stock table
        */
        extern DBSMALLINT s_quantity;
        extern DBSMALLINT threshold;
        extern DBINT low_count;
        extern char s_dist[25];

        /*
        ** Variables for order table
        */

        extern int o_id;
        extern DBTINYINT o_d_id;
        extern DBSMALLINT o_w_id;
        extern DBSMALLINT o_c_id;
        extern char o_entry_d[31];
        extern DBSMALLINT o_carrier_id;


extern DBSMALLINT o_ol_cnt, o_ol_now,
o_ol_done;
extern DBTINYINT o_all_local;

/*
** Variables for order_line
*/

extern int ol_o_id;
extern DBTINYINT ol_d_id;
extern DBSMALLINT ol_w_id;
extern DBSMALLINT ol_number;
extern DBINT ol_i_id;
extern DBSMALLINT ol_supply_w_id;
extern char ol_delivery_d[31];
extern DBSMALLINT ol_quantity;
extern DBFLT8 ol_amount;

/*
** Variables for new_order tble
*/

extern int no_o_id;
extern DBTINYINT no_d_id;
extern DBSMALLINT no_w_id;

/*
** Variables for history table
*/

extern DBFLT8 h_amount;
extern char h_date[20];

#endif SYB_TPCC_H

/*
** Added by SBL to switch between real to dummy
stored procedures
*/
extern int dummy_sp;
```

tpcc_srv_newordpay.c

```
#include <stdio.h>
#include <sys/types.h>
#include <sys/time.h>
#include <time.h>
#include <signal.h>
#include <math.h>
#include <string.h>
#include <stdlib.h>

             /* Sybase header files */
#include “sybfront.h”
#include “sybdb.h”
#include “SYB_tpcc.h”
#include “SYB_driver.h”
```

```
/* #include “SYB_rpc_var.c”  */
             /* Tuxedo includes */
#include “atmi.h”
#include “userlog.h”
#include “tpcc_tux_forms.h”
             /* Lists of items on an order */
             /* These structures should match
the struct definitions for item_struct and
no_struct
              * defined in client.h exactly.
             * Any change to those, should be
reflected here
              */

struct newo_inf *neworder;/* Neworder field
structure */
struct ord_inf *ordstat;/* Input structure to
ordstat_tx */
char blank_mesg[25] = “
“;

             /* List of fields in payment */
             /* This structure should be
EXACTLY identical to the one declared in cli-
ent.h */
#ifdef NOT_REQ
struct pay_inf
{
  int w_id;
  int d_id;
  int c_id;
  int c_w_id;
  int c_d_id;
  double h_amount;
  double c_credit_lim;
  double c_balance;
  double c_discount;
  char h_date[20];
  char w_street_1[21];
  char w_street_2[21];
  char w_city[21];
  char w_state[3];
  char w_zip[11];
  char d_street_1[21];
  char d_street_2[21];
  char d_city[21];
  char d_state[3];
  char d_zip[11];
  char c_first[17];
  char c_middle[3];
  char c_last[17];
  char c_street_1[21];
  char c_street_2[21];
  char c_city[21];
  char c_state[3];
  char c_zip[11];
  char c_phone[17];
  char c_since[11];
  char c_credit[3];


  char c_data_1[51];
  char c_data_2[51];
  char c_data_3[51];
  char c_data_4[51];
};
#endif
struct pay_inf *payp;/* Input structure to
payment_tx */

DBPROCESS *dbproc;
LOGINREC *login;

/* By default, do not use dummy stored proce-
dures */
int dummy_sp = 0;

/*
 * Initialize the neworder transaction
 */
int
init_all_tx ()
{                           /* Install the
error and message handler */
  /* userlog (“before dberrhandle \n”); */
  dberrhandle (err_handler);
  dbmsghandle (msg_handler);

  /* Initialize global variable for error han-
dling */
  deadlock = 0;

  /* userlog (“before dblogin \n”); */
  login = dblogin ();
  /* userlog (“before DBSETLUSER \n”); */
  DBSETLUSER (login, USER);
  /* userlog (“before DBSETLPACKET \n”); */
  DBSETLPACKET (login, 4096);

  /* userlog (“before DBSETLCHARSET \n”); */
  DBSETLCHARSET (login, getenv (“CHARSET”));

  /* Open a dbproc */
  /* userlog (“before dbopen \n”); */
  if ((dbproc = dbopen (login, (char *)
SERVER)) == NULL)
    {
      initerr (“Fatal dbopen: Could not open
connection\n”);
      return (-1);
```

```
    }

  /* Use the the right database */
  /* userlog ("before dbuse \n"); */
  if (dbuse (dbproc, (char *) DATABASE) != SUC-
CEED)
    {
      initerr ("Fatal dbuse: Could not use
DATABASE\n");
      return (-1);
    }

  /* Done with initialization */
  userlog ("leaving tpsvrinit \n");
  return (0);

}

/*
 * This function executes the neworder trans-
action
 */
neworder_tx (rqst)
     TPSVCINFO *rqst;
{

  int i;
  int rollback = 0;
  int linecnt;
  int ret;
  struct items_inf *cur_ip;/* Pointer to cur-
rent item */


  neworder = (struct newo_inf *) (rqst->data);
  linecnt = neworder->o_ol_cnt;

/*
dtcurrent(&ord_date);
dttofmtasc(&ord_date, neworder->o_entry_d,
sizeof(neworder->o_entry_d), "%d-%m-%Y
%H:%M:%S");
*/


/*        datetime(&neworder->o_entry_d); */
  strncpy (neworder->status, blank_mesg, 24);
/* read warehouse, customer */
again:
  neworder->total = 0;

  global_w_id = neworder->w_id;
  global_d_id = neworder->d_id;
  c_id = neworder->c_id;
  o_ol_cnt = neworder->o_ol_cnt;

  o_all_local = 1;/* Assume all local, then
check.
                                Ideally, this
flag can be passed from
                                the FORMs pack-
age */
/*
userlog("after o_all_local ... \n");
*/

  for (i = 0; i < (int) o_ol_cnt; i++)
    {
      cur_ip = &neworder->n_items[i];



      ol[i].i_id = cur_ip->ol_i_id;
      ol[i].supply_w_id = cur_ip-
>ol_supply_w_id;
      ol[i].quantity = cur_ip->ol_quantity;

      if (ol[i].supply_w_id != global_w_id)
       o_all_local = 0;/* non-local order */
    }

  /*
     userlog("before calling new_order_rpc
\n");
     userlog("w_id=%d, d_id=%d, c_id=%d,
o_ol_cnt=%d \n", global_w_id, global_d_id,
c_id,
     o_ol_cnt);
   */
  new_order_rpc ();

  /* pick up total amount */
  neworder->total = total_amount;



  /*
     userlog("after calling new_order_rpc
\n");
     userlog("w_id=%d, d_id=%d, c_id=%d,
o_ol_cnt=%d \n", neworder->w_id, neworder-
>d_id,
     neworder->c_id, neworder->o_ol_cnt);
   */
  tpreturn (TPSUCCESS, 0, rqst->data, sizeof
(struct newo_inf), 0);
}

       /* Start of Tuxedo code */
int
```

```
tpsvrinit (argc, argv)
     int argc;
     char **argv;
{
  int c;

  /* If  Tuxedo server started with -- -d in
CLOPT,
   * then we need to call dummy stored proce-
dures instead of
   * real ones */
  while ((c = getopt (argc, argv, “d”)) != EOF)
    {
      switch (c)
       {
       case ‘d’:
         dummy_sp = 1;
         userlog (“tpsvrinit got arg to use
dummy sp\n”);
         break;
       default:
         break;
       }
    }

  return (init_all_tx ());/* Prepare transac-
tion */
}

void
tpsvrdone ()
{
  dbexit ();
}

NEWO (rqst)
     TPSVCINFO *rqst;
{
  xact_type = XACT_NEWO;
  neworder_tx (rqst);
}


initerr (str)
     char *str;
{
  userlog (“init_all_tx ERROR during %s\n”,
str);
}

ordstat_tx (rqst)
     TPSVCINFO *rqst;
{
  int byid;
  extern int invalid_xact;

  ordstat = (struct ord_inf *) (rqst->data);

  if (ordstat->c_id == 0)
    {                      /* Customer
selected by name */
      byid = FALSE;
      xact_type = XACT_ORDS_NAME;
    }
  else
    {
      byid = TRUE;
      xact_type = XACT_ORDS_ID;
    }

  c_w_id = ordstat->w_id;
  c_d_id = ordstat->d_id;
  if (!byid)
    {
      strcpy (c_last, ordstat->c_last);
      order_status_byname_rpc ();
      ordstat->c_id = c_id;
    }
  else
    {
      c_id = ordstat->c_id;
      order_status_byid_rpc ();
      strcpy (ordstat->c_last, c_last);
    }
  if (invalid_xact)
    ordstat->c_id = 0;

  tpreturn (TPSUCCESS, 0, rqst->data, sizeof
(struct ord_inf), 0);
}

ORDS (rqst)
     TPSVCINFO *rqst;
{
  ordstat_tx (rqst);
}

payment_tx (rqst)
     TPSVCINFO *rqst;
{


  int byid;
  extern int invalid_xact;

  payp = (struct pay_inf *) (rqst->data);

  global_w_id = payp->w_id;
  c_w_id = payp->c_w_id;
  h_amount = payp->h_amount;
```

```
  global_d_id = payp->d_id;
  c_d_id = payp->c_d_id;
  if (payp->c_id == 0)
    {                       /* Customer
selected by name */
      byid = FALSE;
      xact_type = XACT_PAYM_NAME;
    }
  else
    {
      byid = TRUE;
      xact_type = XACT_PAYM_ID;
    }
  if (byid)
    {                       /* Customer
selected by id */
      c_id = payp->c_id;
      payment_byid_rpc ();
    }
  else
    {
      strcpy (c_last, payp->c_last);
      payment_byname_rpc ();
      payp->c_id = c_id;
    }
  if (invalid_xact)
      payp->c_id = 0;/* failed */
  else {

  strcpy (payp->h_date, h_date);
  strcpy (payp->w_street_1, w_street_1);
  strcpy (payp->w_street_2, w_street_2);
  strcpy (payp->w_city, w_city);
  strcpy (payp->w_state, w_state);
  strcpy (payp->w_zip, w_zip);
  strcpy (payp->d_street_1, d_street_1);
  strcpy (payp->d_street_2, d_street_2);
  strcpy (payp->d_city, d_city);
  strcpy (payp->d_state, d_state);
  strcpy (payp->d_zip, d_zip);

  strcpy (payp->c_first, c_first);
  strcpy (payp->c_middle, c_middle);
  strcpy (payp->c_last, c_last);
  strcpy (payp->c_street_1, c_street_1);
  strcpy (payp->c_street_2, c_street_2);
  strcpy (payp->c_city, c_city);
  strcpy (payp->c_state, c_state);
  strcpy (payp->c_zip, c_zip);
  strcpy (payp->c_phone, c_phone);
  strcpy (payp->c_since, c_since);
  strcpy (payp->c_credit, c_credit);

  payp->c_credit_lim = c_credit_lim;
  payp->c_discount = c_discount;
  payp->c_balance = c_balance;

  if (c_data == 0)
    {
      payp->c_data_1[0] =
       payp->c_data_2[0] = payp->c_data_3[0] =
payp->c_data_4[0] = 0;
    }
  else
    {
      strncpy (payp->c_data_1, c_data, 50);
      strncpy (payp->c_data_2, c_data + 50,
50);
      strncpy (payp->c_data_3, c_data + 100,
50);
      strncpy (payp->c_data_4, c_data + 150,
50);
    }

  } /* end if (failed)  else  */

  tpreturn (TPSUCCESS, 0, rqst->data, sizeof
(struct pay_inf), 0);
}

PAYM (rqst)
     TPSVCINFO *rqst;
{
  payment_tx (rqst);
}
```

tpcc_srv_stockdel.c

```
/*
 * File: tpcc_srv_stockdel.ec
 * Delivery and Stock transaction code for
Sybase Tuxedo
 * This program is different from the other
servers, in that it
 * records transaction info in a results file.
 *
 */

/* #include “tpcc_client.h” */
#include <stdlib.h>
#include <sys/signal.h>
#include <sys/utsname.h>
#include <errno.h>
#include <stdio.h>


#include <sys/types.h>
#include <sys/time.h>
```

```
#include <time.h>
#include <sys/times.h>/* buddy */

#include “tpcc_tux_forms.h”

       /* Tuxedo */
#include “atmi.h”
#include “userlog.h”

       /* Sybase header files */
#include <sybfront.h>
#include <sybdb.h>
#include “SYB_tpcc.h”
#include “SYB_driver.h”
       /*  #include “SYB_rpc_var.c”  XXX: Don’t
need this line.  */


static struct req_struct *delp;/* Transaction
message */
extern char outbuf[];
extern int tx_count;/* Transaction counter */
extern FILE *delfile;

       /* List of fields in stock */
       /* This structure should be EXACTLY
identical to the one declared in client.h */
       /* List of fields in stock-level */
       /* struct stock_inf {
          int     w_id;
          int d_id;
          int threshold;
          int low_stock;
          }; */

struct stock_inf *stocklevel;/* Input to
stocklevel transaction */


DBPROCESS *dbproc;
LOGINREC *login;

       /* By default, do not use dummy stored
procedures */
int dummy_sp = 0;

/* times() value for start of this run. - Buddy
*/
long Start_of_run = -1;

cleanup ()
{
  fclose (delfile);
}
```

```
int
init_stockdel_tx ()
{
  /* Prepare delivery transaction */

  /* Install the error and message handler */
  /* userlog (“before dberrhandle \n”); */
  dberrhandle (err_handler);
  dbmsghandle (msg_handler);

  /* initialize global variable for deadlock
and error handling */
  deadlock = 0;


  /* userlog (“before dblogin \n”); */
  login = dblogin ();
  /* userlog (“before DBSETLUSER \n”); */
  DBSETLUSER (login, USER);

  /*
   *      always use large packet ...
   cp = getenv(“CHARSET”);
   userlog(“result of getenv CHARSET. result=
%s \n”, cp);
   cp = getenv(“PACKET”);
   userlog(“result of getenv PACKET. result=
%s \n”, cp);
   if (strcmp(cp,”LARGE”) == 0) {
   */

  /* userlog (“before DBSETLPACKET \n”); */

  DBSETLPACKET (login, 4096);

  /*
     }
   */
  /* userlog (“before DBSETLCHARSET \n”); */
  DBSETLCHARSET (login, getenv (“CHARSET”));

  /* Open a dbproc */
  /* userlog (“before dbopen \n”); */
  if ((dbproc = dbopen (login, (char *)
SERVER)) == NULL)
    {
      initerr (“Fatal dbopen: Could not open
connection\n”);
      return (-1);
    }

  /* Use the the right database */
  /* userlog (“before dbuse \n”); */
  if (dbuse (dbproc, (char *) DATABASE) != SUC-
CEED)
```

```
    {
      initerr ("Fatal dbuse: Could not use
DATABASE\n");
      return (-1);
    }

  /* Done with initialization */
  userlog ("leaving tpsvrinit \n");
  return (0);
}


delivery_tx (rqst)
     TPSVCINFO *rqst;
{
  extern int Delivery_skipped;/* buddy */
  struct tms tmsbuf ;/* buddy */

  /*
   * Read in master times file to get time for
start of run.
   * Needed because report tool expects times-
tamps to be relative to
   * start of run.
   * We must do this at the first transaction,
rather than during tpsrvinit -
   * because the master times file is not cre-
ated until the run starts.
   * (And Tuxedo boots before the run starts.)
   * - buddy
   */
   if (Start_of_run == -1) {
     if (! read_master_times() ) {
      userlog ("WARN: Cannot read master times
file. Times for delivery statistics will not
be synchronized for report.\n");
       Start_of_run = -2;  /* only try this
once */
    }
  }

  delp = (struct req_struct *) (rqst->data);
  global_w_id = delp->w_id;
  o_carrier_id = delp->o_carrier_id;
  tx_count++;

  delivery_rpc ();/* XXX: use Sybase's
SYB_rpc.c version  */

  sprintf (outbuf, "word %d %ld %ld\n",
         Delivery_skipped,
         delp->qtime - Start_of_run,
         times (&tmsbuf) - Start_of_run);

  fwrite (outbuf, strlen (outbuf), 1,
delfile);
  fflush (delfile);
  tpreturn (TPSUCCESS, 0, rqst->data, sizeof
(struct req_struct), 0);
}


/* If errors occur during initialization, exit
*/
initerr (str)
     char *str;
{
  userlog ("init_stockdel_tx ERROR %\n", str);
}


/*
 * Read in master times file to get time for
start of run.
 * Needed because report tool expects times-
tamps to be relative to
 * start of run.
 *
 * Filename is $TMPDIR/<hostname>
 * It is created by writefile - an rte program.
 *
 * - buddy 8/6/01
 */
int
read_master_times()
{
  FILE *fp;
  int  len;
  char *dir;
  char  filename[512];
  struct utsname hostname;
  long   start_rampup, start_runtime,
start_rampdown,
         end_measurement, start_of_run_secs;

  if ((dir = getenv ("TMPDIR")) == (char *)
NULL) {
     userlog ("TMPDIR environment variable not
set\n");
      exit (1);
  }

  uname (&hostname);

  strcpy (filename, dir);
  sprintf (filename, "%s/%s", dir, host-
name.nodename);

  if ((fp = fopen(filename, "r")) == NULL) {
      userlog("Open %s for times failed.
```

```
Errno=%d", filename, errno);
       return(0);
  }

  if (fscanf(fp, "%ld %ld %ld %ld %ld %ld",
              &start_rampup, &start_runtime,
&start_rampdown,
             &end_measurement, &Start_of_run,
&start_of_run_secs) != 6) {

       userlog("WARN: Read of %s for times
failed. Errno=%d", filename, errno);
       return(0);
  }

  userlog("mastertimes: %ld %ld %ld %ld %ld
%ld\n",
         start_rampup, start_runtime,
start_rampdown,
         end_measurement, Start_of_run,
start_of_run_secs);

  return(1);
}

/* Tuxedo */
tpsvrinit (argc, argv)
     int argc;
     char **argv;
{
  char *p, ident[20];
  char filename[200];
  int proc_no, count;
  struct utsname name;


  int c;

/* If  Tuxedo server started with -- -d in
CLOPT,
 * then we need to call dummy stored procedures
instead of
 * real ones */
  while ((c = getopt (argc, argv, "d")) != EOF)
    {
      switch (c)
       {
       case 'd':
         dummy_sp = 1;
         userlog ("tpsvrinit got arg to use
dummy sp\n");
         break;
       default:
         break;
       }
    }


  if ((p = getenv ("TMPDIR")) == (char *) NULL)
    {
     userlog ("TMPDIR environment variable not
set\n");
      exit (1);
    }

/* proc_no = atoi(argv[optind]); *//* Needs
argument which is the proc no */

  proc_no = (int) getpid ();
/* Get hostname of our machine  and create
results file */
  uname (&name);
  strcpy (filename, p);
  sprintf (filename + strlen (filename),
"/%s.del%d", name.nodename, proc_no);
  userlog ("filename = %s \n", filename);
  delfile = fopen (filename, "w");
  if (delfile == NULL)
    {
     userlog ("Cannot create file %s\n", file-
name);
    }

  return (init_stockdel_tx ());/* Prepare
transaction */
}

void
tpsvrdone ()
{
  cleanup ();       /* Close results file */
  dbexit ();
}

DEL (rqst)
     TPSVCINFO *rqst;
{
  xact_type = XACT_BKEND;/* ??? is is OK to
set it here ???? */
  delivery_tx (rqst);
}



/*
 * Function: do stocklevel transaction
 * Input is the stocklevel structure. Output
is low_stock field
 */
stocklevel_tx (rqst)
```

```
     TPSVCINFO *rqst;
{

  stocklevel = (struct stock_inf *) (rqst-
>data);

  global_w_id = stocklevel->w_id;
  global_d_id = stocklevel->d_id;
  threshold = stocklevel->threshold;
  stock_level_rpc ();

  tpreturn (TPSUCCESS, 0, rqst->data, sizeof
(struct stock_inf), 0);
}

STOCK (rqst)
     TPSVCINFO *rqst;
{
  xact_type = XACT_STOCK;/* ??? is is OK to
set it here ???? */
  stocklevel_tx (rqst);
}
```

tpcc_tux_forms.h

```
#ifndef TPCC_TUXFORMS_H
#define TPCC_TUXFORMS_H

     /*  This file contains the definition of
the data structures used
      *  by the tpcc_srv_xxx.c files to commu-
nicate with the Tuxforms.
      *
      *  It should be placed AFTER the follow-
ing include files:
      * #include “atmi.h”
      * #include “userlog.h”
      *
      */

     /* For NEWO:     */

     struct items_inf {
     int ol_supply_w_id;
     int ol_i_id;
     char i_name[25];
     int     ol_quantity;
     int s_quantity;
     char brand[2];
     double i_price;
     double ol_amount;
     } ;

     struct newo_inf {
             int     w_id;
             int     d_id;
             int     c_id;
             int     o_id;
             int     o_ol_cnt;
             double  c_discount;
             double  w_tax;
             double  d_tax;
             char    o_entry_d[20];
             char    c_credit[3];
             char    c_last[17];
     struct items_inf n_items[15];
             char    status[25];
             double  total;
     };


     /* For BKEND:*/

#define DEL_SUCCESS     0
#define DEL_FAIL        1
#define DEL_RETRY       2

     /* Structure used to queue delivery trans-
action */
     struct req_struct {
     int w_id;
     int o_carrier_id;
     time_t qtime;          /* Time transaction
was queued */
     };


     /* For ORDS:*/

     struct ord_itm_inf {
     int ol_supply_w_id;
     int ol_i_id;
     int     ol_quantity;
     double ol_amount;
     char ol_delivery_d[11];
     } ;




struct ord_inf {  /* This structure is copied
from ORDS_VIEW in TPCC.h */
        int     item_cnt;
        int     w_id;
        int     d_id;
        int     c_id;
```

```
        int     o_id;
        int     o_carrier_id;
        double  c_balance;
        char    c_first[17];
        char    c_middle[3];
        char    c_last[17];
        char    o_entry_d[20];
struct ord_itm_inf o_items[15];
};


/* For PAYM:    */

/* List of fields in payment */
/* This structure should be EXACTLY identical
to the one
 * declared in client.h
 */
struct pay_inf {
int     w_id;
int     d_id;
int     c_id;
int     c_w_id;
int     c_d_id;
double h_amount;
double c_credit_lim;
double c_balance;
double c_discount;
char h_date[20];
char w_street_1[21];
char w_street_2[21];
char w_city[21];
char w_state[3];
char w_zip[11];
char d_street_1[21];
char d_street_2[21];
char d_city[21];
char d_state[3];
char d_zip[11];
char c_first[17];
char c_middle[3];
char c_last[17];
char c_street_1[21];
char c_street_2[21];
char c_city[21];
char c_state[3];
char c_zip[11];
char c_phone[17];
char c_since[11];
char c_credit[3];
char c_data_1[51];
char c_data_2[51];
char c_data_3[51];
char c_data_4[51];
};
```

```
/* For STOCK:*/

/* List of fields in stock */
/* This structure should be EXACTLY identical
to the one declared in cli
List of fields in stock-level */
struct stock_inf {
int w_id;
int d_id;
int threshold;
int low_stock;
};
#endif TPCC_TUXFORMS_H
```

tpcc_tux_forms_var.c

```
/* This file declares the share variables
between tpcc_srv_xxx.c and
 * SYB_rpc.c files.
 *
 * The idea is that this file will only be
included in SYB_rpc.c placed
 * after the #include “tpcc_tux_forms.h” line.
The tpcc_srv_xxx.c files
 * will declare those that they need as extern.
 *
 */


/* For NEWO:     */

struct newo_inf newosp;
struct newo_inf *neworder;
FBFR *newo_fbfr;
TPSVCINFO *newo_rqst;
int newolen;

/* For BKEND:   */

struct req_struct *delp;/* Transaction message
*/
char outbuf[1024];/* Buffer for results file
*/
int tx_count = 0;/* Transaction counter */
int Delivery_skipped;/* Added by Buddy  8/6/01
*/


FILE *delfile;
TPSVCINFO *del_rqst;

              /* For ORDS:    */
```

```
struct ord_inf ordsp, *ordstat = &ordsp;
FBFR *ordsbuf;      /* FML buffer for output
carray */
int ordslen;        /* Size of FML buffer */
TPSVCINFO *ords_rqst;


             /* For PAYM: */

struct pay_inf *payment;/* Input structure to
payment_tx */
struct pay_inf paymsp;/* Payment structure */
TPSVCINFO *paym_rqst;


             /* For STOCK:   */

struct stock_inf *stocklevel;/* Input to
stocklevel transaction */
struct stock_inf stocksp;
TPSVCINFO *stock_rqst;
```

Appendix C. Database Layout

Main shell scripts

bld_system

```
#!/bin/sh
locktype=${1:-allpages};export locktype
partition=16;export partition
echo “!!! Using $locktype tables !!!”
echo “!!! you have 5 seconds to abort !!!”
sleep 5

setupscript=./bm_setup

. $setupscript

# verstr=$version; export verstr
# bmbinary=$SQL_RELEASE/bin/buildmaster.$ver-
str; export bmbinary
bmbinary=$SQL_RELEASE/bin/dataserver; export
bmbinary
tpcc_scripts=$TPCC_DIR/tpcc_scripts; export
tpcc_scripts
pagesize=$pagesize; export pagesize

#      RCP FROM EARL RELEASE AREA BUILDMAS-
TER,INSTALLMASTER,DATASERVER

trap “” 22
echo ‘date‘ “Started bld_system”

# Build the device.
‘devcreate.sh buildmaster $bmbinary $verstr
$pagesize < devices ‘
sleep 60
# Boot server, run installmaster
run_server - -c./run.cfg -T1623
sleep 360
isql -Usa -P < $SQL_RELEASE/scripts/install-
master > $$_im.log
sleep 120
msv_rpc_attach.sh

# Build devices, database, and segments
echo ‘date‘ “Creating devices, databases and
segments”
devcreate.sh sql System10 < devices | isql -e
-Usa -P
# sleep 120
# echo ‘date‘ “ Finished building database”

# Extend tempdb
echo “extending tempdb”
$tpcc_scripts/tpcc_ext_tempdb.sh
echo ‘date‘ “expanding log devices “
$tpcc_scripts/tpcc_ext_log.sh
echo ‘date‘ “done expanding log devices “

# Create tables, some indexes, and administra-
tive procs.
$tpcc_scripts/tpcc_tables_parallel.sh $lock-
type $partition
sleep 60
$tpcc_scripts/tpcc_admin.sh
generic_procs.sh

# Truncate log, checkpoint, and shutdown
dumptran_server.sh master
dumptran_server.sh tpcc
shutdown_server.sh
sleep 180
# Start server with no logging and 2 engines
for load.
# Load the data; shutdown again.
run_server  - -c./load.cfg -T699
sleep 360
$tpcc_scripts/tpcc_largeio.sh
echo ‘date‘ “ Started loading data”
$tpcc_scripts/tpcc_load_parallel.sh 1
$DB_SCALE $partition
# partition history table
# $tpcc_scripts/tpcc_alter_history_tbl.sh
echo ‘date‘ “ Finished loading data”
shutdown_server.sh
sleep 120
#
# Reboot, create indexes, config trips, load
stored procs,
# Shutdown.
run_server - -c./run.cfg
sleep 360

echo ‘date‘ -- Started building indexes
$tpcc_scripts/tpcc_indexes_parallel.sh
echo ‘date‘ -- Finished building indexes

$tpcc_scripts/tpcc_proc.sh
$tpcc_scripts/spads_admin.sh
shutdown_server.sh
sleep 60

# Reboot, bind tables and indexes to named
caches, shutdown
run_server - -c./run.cfg
sleep 360
$tpcc_scripts/tpcc_cache_bind.sh
$tpcc_scripts/des_bind.sh
sleep 60
shutdown_server.sh
echo ‘date‘ “Finished bld_system”
```

```
des_bind.sh
#!/usr/bin/sh -f
#
# script to bind the hot objects and proce-
dures to their own des
#
isql -Usa -P$PASSWORD << EOF
declare @dbid   int
select @dbid = db_id("tpcc")
dbcc tune(log_prealloc, @dbid,"on")
dbcc tune(des_bind, @dbid, warehouse)
dbcc tune(des_bind, @dbid, district)
dbcc tune(des_bind, @dbid, item)
dbcc tune(des_bind, @dbid, stock)
dbcc tune(des_bind, @dbid, order_line)
dbcc tune(des_bind, @dbid, orders)
dbcc tune(des_bind, @dbid, new_order)
dbcc tune(des_bind, @dbid, customer)
dbcc tune(des_bind, @dbid, history)
dbcc tune(des_bind, @dbid, neworder_local)
dbcc tune(des_bind, @dbid, neworder_remote)
dbcc tune(des_bind, @dbid, payment_byid)
dbcc tune(des_bind, @dbid, payment_byname)
dbcc tune(des_bind, @dbid, order_status_byid)
dbcc tune(des_bind, @dbid,
order_status_byname)
dbcc tune(des_bind, @dbid, delivery)
dbcc tune(des_bind, @dbid, stock_level)
dbcc tune(freelock_xfr_bsize, 150)
dbcc tune(max_eng_freelocks, 50)
dbcc tune(deviochar, -1, "8")
dbcc tune("doneinproc",0)
dbcc tune("cleanup",0)
go
use master
go
sp_dboption tpcc,"trunc log on chkpt",false
go
sp_dboption tpcc,"no free space acctg",true
go
use tpcc
go
checkpoint
go
EOF
```

devcreate.sh

```
#!/bin/sh
# @(#) devcreate.sh 1.1 06/07/95
#
#      scripts/devcreate.sh
#
#      Read a device file from stdin in the
format given in format/devices
#      and ouput on stdout the SQL statements
to create the devices,
#      databases and segments defined by the
input device file.
#
#      The SQL is output in the following order
#
#      1) Disk inits and disk mirrors
#      2) Create databases
#      3) sp_addsegments and sp_extendsegments
#

if [ "$1" = "buildmaster" ]
then
       bm=y
       bmbinary=$2
       verstr=$3
       pagesize=$4
elif [ "$1" = "sql" ]
then
       bm=n
       release=$2
else
       echo "Usage : $0 [builmaster
buildmaster_binary |sql] < device_file" >&2
       echo "buildmaster - generate buildmas-
ter command" >&2
       echo "sql [release] - generate SQL com-
mands" >&2
       exit 1
fi

in_device=n
in_db=n

logical_name=
physical_name=
device_size=

vdevno=0

sql_file=/tmp/dvsql$$
db_file=/tmp/dvdb$$
db_s_file=/tmp/dvdb_s$$
seg_file=/tmp/dvseg$$
export sql_file db_file seg_file
```

```
grep -v '^#' | tr -s '\011 ' '\012\012' | while
read token garbage
do
       case $token in
              DEVICE)# A new device
                     # clear the fields for the
next device
                     logical_name=
                     physical_name=
                     device_size=
                     vstart_offset=
                     mirror=

                     in_device=y
                     in_db=n
                     ;;

              db=*)  # database name
                     if [ "$in_db" = "y" ]
                     then
                            # Store info about
db
                            echo
                     fi

                     # Start the new database
                     db_name=`echo $token | sed
's/db=//''
                     db_log=
                     db_size=0

                     in_db=y
                     ;;


              log)   # This disk is the log
disk for the current db
                     if [ "$in_db" = "y" ]
                     then
                            db_log="log on"
                     fi
                     ;;

              vstart=*)
                     vstart_offset=`echo $token
| sed 's/vstart=//''
                     ;;

              mirror=*)
                     # store info about log-
mirror
                     mirror=`echo $token | sed
's/mirror=//''
                     ;;

              size=*) # the size of the current
db on this disk
                     if [ "$in_db" = "y" ]
                     then
                            # Store info about
size
                            db_size=`echo
$token | sed 's/size=//''
                     fi
                     ;;

              segment=*)
                     if [ "$in_db" = "y" ]
                     then
                            segment=`echo
$token | sed 's/segment=//''
                            echo "$db_name
$segment $logical_name" >> $seg_file
                     fi
                     ;;

              DEVICE_END)# Complete the device

                     # Save any database frag-
ment information
                     if [ "$in_db" = "y" ]
                     then
                            echo "$db_name
$logical_name $db_size $db_log" >> $db_file
                     fi

                     if [ "$bm" = "y" -a
"$in_device" = "y" -a "$logical_name" = "mas-
ter" ]
                     then
                            # Convert Mb to 2k
pages.
                            numpages=`expr
$device_size \* 512`

                            if [ "$verstr" =
"12_5" -o "$verstr" = "14_0" ]
                            then
                                   echo "$bmbi-
nary -d$physical_name -z$pagesize -b$numpages
-f "
                            else
                                   echo "$bmbi-
nary -d$physical_name -s$numpages"
                            fi
                     fi

                     # The disk init SQL, but
not for the master device
```

```
                    #
                    if [ “$bm” = “n” -a
“$in_device” = “y” -a “$logical_name” != “mas-
ter” ]
                    then
                          # Convert Mb to 2k
pages.
                          numpages=‘expr
$device_size \* 512‘

                          # echo SQL to cre-
ate the device
                          echo “disk init”
                          echo “  name =
‘$logical_name’,”
                          echo “  physname =
‘$physical_name’,”
                          echo “  vdevno =
$vdevno,”
                          echo “  size =
$numpages”
                          if [
“$vstart_offset” != “” ]
                          then
                                echo “  ,
vstart = $vstart_offset”
                          fi
                          echo go
                    fi


                    # The disk mirror SQL
(including master)
                    if [ “$mirror” = “” -o
“$bm” = “y” ]
                    then
                          false;
                    else
                          # Echo SQL to cre-
ate the disk mirror
                          echo “disk mirror”
                          echo “  name =
‘$logical_name’,”
                          echo “  mirror =
‘$mirror’,”
                          echo “  writes =
noserial”
                          echo go
                    fi

                    ;;

              *)    # could be one of several
                    if [ “$in_device” = “y” ]
                    then
                                if [
“$logical_name” = “” ]
                                then

logical_name=$token
                                      if [
$logical_name != “master” ]
                                      then

vdevno=‘expr $vdevno + 1‘
                                      fi
                                elif [
“$physical_name” = “” ]
                                then

physical_name=$token
                                elif [
“$device_size” = “” ]
                                then

device_size=$token
                                else
                                      echo
                                fi
                          else
                                echo
                          fi
                          ;;
        esac
done

# If we are in buildmaster mode we can just
stop here.
if [ “$bm” = “y” ]
then
        rm $db_file $seg_file
        exit 0
fi

#
# Now we have generated the disk init com-
mands, create the
# create database commands.
#
# The file $db_file will have been created
with the following format
#
# dbname device size [log on]

#sort $db_file > $db_s_file
cat $db_file > $db_s_file
rm $db_file

# Add a dummy line end to the database file
echo “__$$” >> $db_s_file
```

```
current_db=
in_db=n
logdbinfo=
export in_db current_db logdbinfo
cat $db_s_file | while read dbname device size
log
do
       if [ “$dbname” = “$current_db” ]
       then
              if [ -z “$log” ]
              then
                     if [ -z “$dbinfo” ]
                     then
                            dbinfo=”on $device
= $size”
                     else
                            dbinfo=”$dbinfo,
$device = $size”
                     fi
              else
                     logdbinfo=”$log $device =
$size”
              fi
       elif  [ “$in_db” = “y” ]
       then
              echo “create database
$current_db”
              echo $dbinfo
              if [ -n “$logdbinfo” ]
              then
                     echo $logdbinfo
              fi
              echo go
              logdbinfo=

              current_db=$dbname
              dbinfo=”on $device = $size”
              in_db=y
       else
              current_db=$dbname
              if [ -z “$log” ]
              then
                     dbinfo=”on $device =
$size”
              else
                     logdbinfo=”$log $device =
$size”
              fi
              in_db=y
       fi
done
#rm $db_s_file

#
# Now we have the create database commands,
create the segment commands
#
# The file $seg_file will have been created
with the following format
#
# dbname device segment
#

current_db=
current_seg=
seg_db=
export current_seg current_db seg_db

sort $seg_file | while read dbname segment
device garbage
do
       if [ “$dbname” = “$current_db” ]
       then
              false
       else
              echo “use $dbname”
              echo go
              # In System 10 segment procs now
takes db as 2nd arg
              if [ “$release” = “System10” ]
              then
                     seg_db=”$dbname ,”
              fi
       fi

       if [ “$segment” = “system” -o “$segment”
= “default” ]
       then
              false # do nothing
       elif [ “$segment” = “$current_seg” ]
       then
              echo “sp_extendsegment $segment ,
$seg_db  $device”
              echo go
       else
              echo “sp_addsegment $segment ,
$seg_db $device”
              echo go
       fi
       current_seg=$segment
       current_db=$dbname
done

# now sort the segment file in database,
device order
# to enable us to drop the unwanted system and
default segments

in_device=no
```

```
export in_device
sort +0 -1 +2 -3 $seg_file | while read dbname
segment device garbage
do
       if [ “$device” = “$current_dev” ]
       then
              false
       else
              if [ “$in_device” = “yes” ]
              then
                     if [ “$drop_segs” = “yes”
]
                     then
                            echo
“sp_dropsegment ‘default’, $seg_db
$current_dev”
                            echo go
                            echo
“sp_dropsegment ‘system’, $seg_db
$current_dev”
                            echo go
                     fi
              fi

              in_device=yes
              drop_segs=yes
       fi

       if [ “$dbname” = “$current_db” ]
       then
              false
       else
              echo “use $dbname”
              echo go
              # In System 10 segment procs now
takes db as 2nd arg
              if [ “$release” = “System10” ]
              then
                     seg_db=”$dbname ,”
              fi
       fi

       if [ “$segment” = “system” -o “$segment”
= “default” ]
       then
              drop_segs=no
       fi

       current_dev=$device
       current_db=$dbname
done

rm $seg_file

echo   “use master”
echo   go
echo   “checkpoint”
echo   go
```

devices

```
DEVICE master /mnt/tpccvol1/tpcc_master 300
       db=tpcc size=300
       segment=default segment=system
DEVICE_END
DEVICE stock1 /mnt/tpccvol1/tpcc_stock1 6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock2 /mnt/tpccvol2/tpcc_stock2 6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock3 /mnt/tpccvol3/tpcc_stock3 6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock4 /mnt/tpccvol4/tpcc_stock4 6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock5 /mnt/tpccvol5/tpcc_stock5 6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock6 /mnt/tpccvol6/tpcc_stock6 6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock7 /mnt/tpccvol7/tpcc_stock7 6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock8 /mnt/tpccvol8/tpcc_stock8 6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock9 /mnt/tpccvol9/tpcc_stock9 6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock10 /mnt/tpccvol10/tpcc_stock10
6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock11 /mnt/tpccvol11/tpcc_stock11
6965
       db=tpcc size=6965
```

```
       segment=Sstock
DEVICE_END
DEVICE stock12 /mnt/tpccvol12/tpcc_stock12
6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock13 /mnt/tpccvol13/tpcc_stock13
6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock14 /mnt/tpccvol14/tpcc_stock14
6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock15 /mnt/tpccvol15/tpcc_stock15
6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock16 /mnt/tpccvol16/tpcc_stock16
6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock17 /mnt/tpccvol17/tpcc_stock17
6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock18 /mnt/tpccvol18/tpcc_stock18
6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock19 /mnt/tpccvol19/tpcc_stock19
6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock20 /mnt/tpccvol20/tpcc_stock20
6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock21 /mnt/tpccvol21/tpcc_stock21
6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock22 /mnt/tpccvol22/tpcc_stock22
6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock23 /mnt/tpccvol23/tpcc_stock23
6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock24 /mnt/tpccvol24/tpcc_stock24
6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock25 /mnt/tpccvol1/tpcc_stock25 6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock26 /mnt/tpccvol2/tpcc_stock26 6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock27 /mnt/tpccvol3/tpcc_stock27 6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock28 /mnt/tpccvol4/tpcc_stock28 6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock29 /mnt/tpccvol5/tpcc_stock29 6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock30 /mnt/tpccvol6/tpcc_stock30 6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock31 /mnt/tpccvol7/tpcc_stock31 6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock32 /mnt/tpccvol8/tpcc_stock32 6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock33 /mnt/tpccvol9/tpcc_stock33 6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock34 /mnt/tpccvol10/tpcc_stock34
6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock35 /mnt/tpccvol11/tpcc_stock35
6965
```

```
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock36 /mnt/tpccvol12/tpcc_stock36
6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock37 /mnt/tpccvol13/tpcc_stock37
6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock38 /mnt/tpccvol14/tpcc_stock38
6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock39 /mnt/tpccvol15/tpcc_stock39
6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock40 /mnt/tpccvol16/tpcc_stock40
6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock41 /mnt/tpccvol17/tpcc_stock41
6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock42 /mnt/tpccvol18/tpcc_stock42
6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock43 /mnt/tpccvol19/tpcc_stock43
6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock44 /mnt/tpccvol20/tpcc_stock44
6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock45 /mnt/tpccvol21/tpcc_stock45
6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock46 /mnt/tpccvol22/tpcc_stock46
6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock47 /mnt/tpccvol23/tpcc_stock47
6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE stock48 /mnt/tpccvol24/tpcc_stock48
6965
       db=tpcc size=6965
       segment=Sstock
DEVICE_END
DEVICE orderline1 /mnt/tpccvol1/tpcc_ordl1
8900
       db=tpcc size=8900
       segment=Sorder_line
DEVICE_END
DEVICE orderline2 /mnt/tpccvol2/tpcc_ordl2
8900
       db=tpcc size=8900
       segment=Sorder_line
DEVICE_END
DEVICE orderline3 /mnt/tpccvol3/tpcc_ordl3
8900
       db=tpcc size=8900
       segment=Sorder_line
DEVICE_END
DEVICE orderline4 /mnt/tpccvol4/tpcc_ordl4
8900
       db=tpcc size=8900
       segment=Sorder_line
DEVICE_END
DEVICE orderline5 /mnt/tpccvol5/tpcc_ordl5
8900
       db=tpcc size=8900
       segment=Sorder_line
DEVICE_END
DEVICE orderline6 /mnt/tpccvol6/tpcc_ordl6
8900
       db=tpcc size=8900
       segment=Sorder_line
DEVICE_END
DEVICE orderline7 /mnt/tpccvol7/tpcc_ordl7
8900
       db=tpcc size=8900
       segment=Sorder_line
DEVICE_END
DEVICE orderline8 /mnt/tpccvol8/tpcc_ordl8
8900
       db=tpcc size=8900
       segment=Sorder_line
DEVICE_END
DEVICE orderline9 /mnt/tpccvol9/tpcc_ordl9
8900
```

```
       db=tpcc size=8900
       segment=Sorder_line
DEVICE_END
DEVICE orderline10 /mnt/tpccvol10/tpcc_ordl10
8900
       db=tpcc size=8900
       segment=Sorder_line
DEVICE_END
DEVICE orderline11 /mnt/tpccvol11/tpcc_ordl11
8900
       db=tpcc size=8900
       segment=Sorder_line
DEVICE_END
DEVICE orderline12 /mnt/tpccvol12/tpcc_ordl12
8900
       db=tpcc size=8900
       segment=Sorder_line
DEVICE_END
DEVICE orderline13 /mnt/tpccvol13/tpcc_ordl13
8900
       db=tpcc size=8900
       segment=Sorder_line
DEVICE_END
DEVICE orderline14 /mnt/tpccvol14/tpcc_ordl14
8900
       db=tpcc size=8900
       segment=Sorder_line
DEVICE_END
DEVICE orderline15 /mnt/tpccvol15/tpcc_ordl15
8900
       db=tpcc size=8900
       segment=Sorder_line
DEVICE_END
DEVICE orderline16 /mnt/tpccvol16/tpcc_ordl16
8900
       db=tpcc size=8900
       segment=Sorder_line
DEVICE_END
DEVICE orderline17 /mnt/tpccvol17/tpcc_ordl17
8900
       db=tpcc size=8900
       segment=Sorder_line
DEVICE_END
DEVICE orderline18 /mnt/tpccvol18/tpcc_ordl18
8900
       db=tpcc size=8900
       segment=Sorder_line
DEVICE_END
DEVICE orderline19 /mnt/tpccvol19/tpcc_ordl19
8900
       db=tpcc size=8900
       segment=Sorder_line
DEVICE_END
DEVICE orderline20 /mnt/tpccvol20/tpcc_ordl20
8900
       db=tpcc size=8900
       segment=Sorder_line
DEVICE_END
DEVICE orderline21 /mnt/tpccvol21/tpcc_ordl21
8900
       db=tpcc size=8900
       segment=Sorder_line
DEVICE_END
DEVICE orderline22 /mnt/tpccvol22/tpcc_ordl22
8900
       db=tpcc size=8900
       segment=Sorder_line
DEVICE_END
DEVICE orderline23 /mnt/tpccvol23/tpcc_ordl23
8900
       db=tpcc size=8900
       segment=Sorder_line
DEVICE_END
DEVICE orderline24 /mnt/tpccvol24/tpcc_ordl24
8900
       db=tpcc size=8900
       segment=Sorder_line
DEVICE_END
DEVICE tpcc_log1 /mnt/logs/tpcc_log1 32000
       db=tpcc size=32000
       log
DEVICE_END
DEVICE tpcc_log2 /mnt/logs/tpcc_log2 32000
       db=tpcc size=32000
       log
DEVICE_END
DEVICE tpcc_log3 /mnt/logs/tpcc_log3 32000
       db=tpcc size=32000
       log
DEVICE_END
DEVICE customer1 /mnt/tpccvol24/tpcc_cust1
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer2 /mnt/tpccvol23/tpcc_cust2
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer3 /mnt/tpccvol22/tpcc_cust3
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer4 /mnt/tpccvol21/tpcc_cust4
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
```

```
DEVICE customer5 /mnt/tpccvol20/tpcc_cust5
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer6 /mnt/tpccvol19/tpcc_cust6
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer7 /mnt/tpccvol18/tpcc_cust7
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer8 /mnt/tpccvol17/tpcc_cust8
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer9 /mnt/tpccvol16/tpcc_cust9
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer10 /mnt/tpccvol15/tpcc_cust10
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer11 /mnt/tpccvol14/tpcc_cust11
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer12 /mnt/tpccvol13/tpcc_cust12
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer13 /mnt/tpccvol12/tpcc_cust13
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer14 /mnt/tpccvol11/tpcc_cust14
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer15 /mnt/tpccvol10/tpcc_cust15
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer16 /mnt/tpccvol9/tpcc_cust16
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer17 /mnt/tpccvol8/tpcc_cust17
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer18 /mnt/tpccvol7/tpcc_cust18
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer19 /mnt/tpccvol6/tpcc_cust19
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer20 /mnt/tpccvol5/tpcc_cust20
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer21 /mnt/tpccvol4/tpcc_cust21
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer22 /mnt/tpccvol3/tpcc_cust22
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer23 /mnt/tpccvol2/tpcc_cust23
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer24 /mnt/tpccvol1/tpcc_cust24
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer25 /mnt/tpccvol24/tpcc_cust25
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer26 /mnt/tpccvol23/tpcc_cust26
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
```

```
DEVICE customer27 /mnt/tpccvol22/tpcc_cust27
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer28 /mnt/tpccvol21/tpcc_cust28
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer29 /mnt/tpccvol20/tpcc_cust29
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer30 /mnt/tpccvol19/tpcc_cust30
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer31 /mnt/tpccvol18/tpcc_cust31
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer32 /mnt/tpccvol17/tpcc_cust32
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer33 /mnt/tpccvol16/tpcc_cust33
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer34 /mnt/tpccvol15/tpcc_cust34
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer35 /mnt/tpccvol14/tpcc_cust35
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer36 /mnt/tpccvol13/tpcc_cust36
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer37 /mnt/tpccvol12/tpcc_cust37
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
```

```
DEVICE customer38 /mnt/tpccvol11/tpcc_cust38
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer39 /mnt/tpccvol10/tpcc_cust39
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer40 /mnt/tpccvol9/tpcc_cust40
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer41 /mnt/tpccvol8/tpcc_cust41
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer42 /mnt/tpccvol7/tpcc_cust42
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer43 /mnt/tpccvol6/tpcc_cust43
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer44 /mnt/tpccvol5/tpcc_cust44
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer45 /mnt/tpccvol4/tpcc_cust45
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer46 /mnt/tpccvol3/tpcc_cust46
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer47 /mnt/tpccvol2/tpcc_cust47
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
DEVICE customer48 /mnt/tpccvol1/tpcc_cust48
4595
       db=tpcc size=4595
       segment=Scustomer segment=Scidx
DEVICE_END
```

```
DEVICE history1 /mnt/tpccvol1/tpcc_history1
2409
       db=tpcc size=2409
       segment=Shistory
DEVICE_END
DEVICE history2 /mnt/tpccvol2/tpcc_history2
2409
       db=tpcc size=2409
       segment=Shistory
DEVICE_END
DEVICE history3 /mnt/tpccvol3/tpcc_history3
2409
       db=tpcc size=2409
       segment=Shistory
DEVICE_END
DEVICE history4 /mnt/tpccvol4/tpcc_history4
2409
       db=tpcc size=2409
       segment=Shistory
DEVICE_END
DEVICE history5 /mnt/tpccvol5/tpcc_history5
2409
       db=tpcc size=2409
       segment=Shistory
DEVICE_END
DEVICE history6 /mnt/tpccvol6/tpcc_history6
2409
       db=tpcc size=2409
       segment=Shistory
DEVICE_END
DEVICE history7 /mnt/tpccvol7/tpcc_history7
2409
       db=tpcc size=2409
       segment=Shistory
DEVICE_END
DEVICE history8 /mnt/tpccvol8/tpcc_history8
2409
       db=tpcc size=2409
       segment=Shistory
DEVICE_END
DEVICE history10 /mnt/tpccvol10/tpcc_history10
2409
       db=tpcc size=2409
       segment=Shistory
DEVICE_END
DEVICE new_order1 /mnt/tpccvol9/tpcc_norder1
500
       db=tpcc size=500
       segment=Snew_order
DEVICE_END
DEVICE new_order2 /mnt/tpccvol10/tpcc_norder2
500
       db=tpcc size=500
       segment=Snew_order
DEVICE_END
DEVICE new_order3 /mnt/tpccvol11/tpcc_norder3
500
       db=tpcc size=500
       segment=Snew_order
DEVICE_END
DEVICE new_order4 /mnt/tpccvol12/tpcc_norder4
500
       db=tpcc size=500
       segment=Snew_order
DEVICE_END
DEVICE new_order5 /mnt/tpccvol13/tpcc_norder5
500
       db=tpcc size=500
       segment=Snew_order
DEVICE_END
DEVICE new_order6 /mnt/tpccvol14/tpcc_norder6
500
       db=tpcc size=500
       segment=Snew_order
DEVICE_END
DEVICE new_order7 /mnt/tpccvol15/tpcc_norder7
500
       db=tpcc size=500
       segment=Snew_order
DEVICE_END
DEVICE new_order8 /mnt/tpccvol16/tpcc_norder8
500
       db=tpcc size=500
       segment=Snew_order
DEVICE_END
DEVICE orders1 /mnt/tpccvol17/tpcc_orders1
2085
       db=tpcc size=2085
       segment=Sorders
DEVICE_END
DEVICE orders2 /mnt/tpccvol18/tpcc_orders2
2085
       db=tpcc size=2085
       segment=Sorders
DEVICE_END
DEVICE orders3 /mnt/tpccvol19/tpcc_orders3
2085
       db=tpcc size=2085
       segment=Sorders
DEVICE_END
DEVICE orders4 /mnt/tpccvol20/tpcc_orders4
2085
       db=tpcc size=2085
       segment=Sorders
DEVICE_END
DEVICE orders5 /mnt/tpccvol21/tpcc_orders5
2085
       db=tpcc size=2085
       segment=Sorders
DEVICE_END
```

```
DEVICE orders6 /mnt/tpccvol22/tpcc_orders6
2085
       db=tpcc size=2085
       segment=Sorders
DEVICE_END
DEVICE orders7 /mnt/tpccvol23/tpcc_orders7
2085
       db=tpcc size=2085
       segment=Sorders
DEVICE_END
DEVICE orders8 /mnt/tpccvol24/tpcc_orders8
2085
       db=tpcc size=2085
       segment=Sorders
DEVICE_END
DEVICE wdino1 /mnt/tpccvol24/tpcc_awdino1 439
       db=tpcc size=439
         segment=Swarehouse segment=Sdistrict
segment=Sitem
DEVICE_END
DEVICE wdino2 /mnt/tpccvol23/tpcc_awdino2 439
       db=tpcc size=439
         segment=Swarehouse segment=Sdistrict
segment=Sitem
DEVICE_END
DEVICE wdino3 /mnt/tpccvol22/tpcc_awdino3 439
       db=tpcc size=439
         segment=Swarehouse segment=Sdistrict
segment=Sitem
DEVICE_END
DEVICE wdino4 /mnt/tpccvol21/tpcc_awdino4 439
       db=tpcc size=439
         segment=Swarehouse segment=Sdistrict
segment=Sitem
DEVICE_END
DEVICE wdino5 /mnt/tpccvol20/tpcc_awdino5 439
       db=tpcc size=439
         segment=Swarehouse segment=Sdistrict
segment=Sitem
DEVICE_END
DEVICE wdino6 /mnt/tpccvol19/tpcc_awdino6 439
       db=tpcc size=439
         segment=Swarehouse segment=Sdistrict
segment=Sitem
DEVICE_END
DEVICE wdino7 /mnt/tpccvol18/tpcc_awdino7 439
       db=tpcc size=439
         segment=Swarehouse segment=Sdistrict
segment=Sitem
DEVICE_END
DEVICE wdino8 /mnt/tpccvol17/tpcc_awdino8 439
       db=tpcc size=439
         segment=Swarehouse segment=Sdistrict
segment=Sitem
DEVICE_END
```

dumptran_server.sh

```
#!/bin/sh -f
isql -Usa -P$PASSWORD << EOF
dump tran $1 with truncate_only
go
use $1
go
checkpoint
go
EOF
```

filer_vols_16way.sh

```
rsh pike vol create logs -r 14 -d 8a.18 8a.19
8a.20 8a.21 8a.22 8a.23 8a.24 8a.25 8a.26
8a.27 8a.28
rsh pike vol options logs nosnap on
rsh pike vol options logs minra on
rsh pike vol options logs no_atime_update on
rsh pike options dafs.enable on
rsh pike dafs start
rsh pike options dafs.auth_none.enable on
rsh pike dafs export create logs /vol/logs
rsh pike options dafs.dynamic_nreq.enable off
rsh pike options dafs.max_requests 1000
rsh pike options dafs.idle_timeout 28800

rsh pulaski vol create tpccvol1 -r 27 -d 7.16
7.17 7.18 7.19 7.20 7.21 7.22 7.23 7.24 7.25
7.26 7.27 7.28 7.29 7.32 7.33 7.34 7.35 7.36
7.37 7.38 7.39 7.40 7.41 7.42 7.43 7.44
rsh pulaski vol options tpccvol1 nosnap on
rsh pulaski vol options tpccvol1 minra on
rsh pulaski vol options tpccvol1
no_atime_update on
rsh pulaski options dafs.enable on
rsh pulaski dafs start
rsh pulaski options dafs.auth_none.enable on
rsh pulaski dafs export create tpccvol1
/vol/tpccvol1
rsh pulaski options dafs.dynamic_nreq.enable
off
rsh pulaski options dafs.max_requests 1000
rsh pulaski options dafs.idle_timeout 28800

rsh pulaski vol create tpccvol9 -r 27 -d 8.18
8.19 8.20 8.21 8.22 8.23 8.24 8.25 8.26 8.27
8.28 8.29 8.32 8.33 8.34 8.35 8.36 8.37 8.38
8.39 8.40 8.41 8.42 8.43 8.44 8.45 9.16
rsh pulaski vol options tpccvol9 nosnap on
rsh pulaski vol options tpccvol9 minra on
rsh pulaski vol options tpccvol9
no_atime_update on
```

```
rsh pulaski dafs export create tpccvol9
/vol/tpccvol9

rsh pulaski vol create tpccvol17 -r 27 -d 9.17
9.18 9.19 9.20 9.21 9.22 9.23 9.24 9.25 9.26
9.27 9.28 9.29 9.32 9.33 9.34 9.35 9.36 9.37
9.38 9.39 9.40 9.41 9.42 9.43 9.44 9.45
rsh pulaski vol options tpccvol17 nosnap on
rsh pulaski vol options tpccvol17 minra on
rsh pulaski vol options tpccvol17
no_atime_update on
rsh pulaski dafs export create tpccvol17
/vol/tpccvol17

rsh keiko vol create tpccvol2 -r 27 -d 7.16
7.17 7.18 7.19 7.20 7.21 7.22 7.23 7.24 7.25
7.26 7.27 7.28 7.29 7.32 7.33 7.34 7.35 7.36
7.37 7.38 7.39 7.40 7.41 7.42 7.43 7.44
rsh keiko vol options tpccvol2 nosnap on
rsh keiko vol options tpccvol2 minra on
rsh keiko vol options tpccvol2 no_atime_update
on
rsh keiko options dafs.enable on
rsh keiko dafs start
rsh keiko options dafs.auth_none.enable on
rsh keiko dafs export create tpccvol2
/vol/tpccvol2
rsh keiko options dafs.dynamic_nreq.enable off
rsh keiko options dafs.max_requests 1000
rsh keiko options dafs.idle_timeout 28800

rsh keiko vol create tpccvol10 -r 27 -d 8.18
8.19 8.20 8.21 8.22 8.23 8.24 8.25 8.26 8.27
8.28 8.29 8.32 8.33 8.34 8.35 8.36 8.37 8.38
8.39 8.40 8.41 8.42 8.43 8.44 8.45 9.16
rsh keiko vol options tpccvol10 nosnap on
rsh keiko vol options tpccvol10 minra on
rsh keiko vol options tpccvol10
no_atime_update on
rsh keiko dafs export create tpccvol10
/vol/tpccvol10

rsh keiko vol create tpccvol18 -r 27 -d 9.17
9.18 9.19 9.20 9.21 9.22 9.23 9.24 9.25 9.26
9.27 9.28 9.29 9.32 9.33 9.34 9.35 9.36 9.37
9.38 9.39 9.40 9.41 9.42 9.43 9.44 9.45
rsh keiko vol options tpccvol18 nosnap on
rsh keiko vol options tpccvol18 minra on
rsh keiko vol options tpccvol18
no_atime_update on
rsh keiko dafs export create tpccvol18
/vol/tpccvol18

rsh wynn vol create tpccvol3 -r 27 -d 7.16 7.17
7.18 7.19 7.20 7.21 7.22 7.23 7.24 7.25 7.26
7.27 7.28 7.29 7.32 7.33 7.34 7.35 7.36 7.37
7.38 7.39 7.40 7.41 7.42 7.43 7.44
rsh wynn vol options tpccvol3 nosnap on
rsh wynn vol options tpccvol3 minra on
rsh wynn vol options tpccvol3 no_atime_update
on
rsh wynn options dafs.enable on
rsh wynn dafs start
rsh wynn options dafs.auth_none.enable on
rsh wynn dafs export create tpccvol3
/vol/tpccvol3
rsh wynn options dafs.dynamic_nreq.enable off
rsh wynn options dafs.max_requests 1000
rsh wynn options dafs.idle_timeout 28800


rsh wynn vol create tpccvol11 -r 27 -d 8.18
8.19 8.20 8.21 8.22 8.23 8.24 8.25 8.26 8.27
8.28 8.29 8.32 8.33 8.34 8.35 8.36 8.37 8.38
8.39 8.40 8.41 8.42 8.43 8.44 8.45 9.16
rsh wynn vol options tpccvol11 nosnap on
rsh wynn vol options tpccvol11 minra on
rsh wynn vol options tpccvol11 no_atime_update
on
rsh wynn dafs export create tpccvol11
/vol/tpccvol11

rsh wynn vol create tpccvol19 -r 27 -d 9.17
9.18 9.19 9.20 9.21 9.22 9.23 9.24 9.25 9.26
9.27 9.28 9.29 9.32 9.33 9.34 9.35 9.36 9.37
9.38 9.39 9.40 9.41 9.42 9.43 9.44 9.45
rsh wynn vol options tpccvol19 nosnap on
rsh wynn vol options tpccvol19 minra on
rsh wynn vol options tpccvol19 no_atime_update
on
rsh wynn dafs export create tpccvol19
/vol/tpccvol19

rsh betor vol create tpccvol4 -r 27 -d 7.16
7.17 7.18 7.19 7.20 7.21 7.22 7.23 7.24 7.25
7.26 7.27 7.28 7.29 7.32 7.33 7.34 7.35 7.36
7.37 7.38 7.39 7.40 7.41 7.42 7.43 7.44
rsh betor vol options tpccvol4 nosnap on
rsh betor vol options tpccvol4 minra on
rsh betor vol options tpccvol4 no_atime_update
on
rsh betor options dafs.enable on
rsh betor dafs start
rsh betor options dafs.auth_none.enable on
rsh betor dafs export create tpccvol4
/vol/tpccvol4
rsh betor options dafs.dynamic_nreq.enable off
rsh betor options dafs.max_requests 1000
rsh betor options dafs.idle_timeout 28800
```

```
rsh betor vol create tpccvol12 -r 27 -d 8.18
8.19 8.20 8.21 8.22 8.23 8.24 8.25 8.26 8.27
8.28 8.29 8.32 8.33 8.34 8.35 8.36 8.37 8.38
8.39 8.40 8.41 8.42 8.43 8.44 8.45 9.16
rsh betor vol options tpccvol12 nosnap on
rsh betor vol options tpccvol12 minra on
rsh betor vol options tpccvol12
no_atime_update on
rsh betor dafs export create tpccvol12
/vol/tpccvol12

rsh betor vol create tpccvol20 -r 27 -d 9.17
9.18 9.19 9.20 9.21 9.22 9.23 9.24 9.25 9.26
9.27 9.28 9.29 9.32 9.33 9.34 9.35 9.36 9.37
9.38 9.39 9.40 9.41 9.42 9.43 9.44 9.45
rsh betor vol options tpccvol20 nosnap on
rsh betor vol options tpccvol20 minra on
rsh betor vol options tpccvol20
no_atime_update on
rsh betor dafs export create tpccvol20
/vol/tpccvol20

rsh decker vol create tpccvol5 -r 27 -d 7.16
7.17 7.18 7.19 7.20 7.21 7.22 7.23 7.24 7.25
7.26 7.27 7.28 7.29 7.32 7.33 7.34 7.35 7.36
7.37 7.38 7.39 7.40 7.41 7.42 7.43 7.44
rsh decker vol options tpccvol5 nosnap on
rsh decker vol options tpccvol5 minra on
rsh decker vol options tpccvol5
no_atime_update on
rsh decker options dafs.enable on
rsh decker dafs start
rsh decker options dafs.auth_none.enable on
rsh decker dafs export create tpccvol5
/vol/tpccvol5
rsh decker options dafs.dynamic_nreq.enable
off
rsh decker options dafs.max_requests 1000
rsh decker options dafs.idle_timeout 28800

rsh decker vol create tpccvol13 -r 27 -d 8.18
8.19 8.20 8.21 8.22 8.23 8.24 8.25 8.26 8.27
8.28 8.29 8.32 8.33 8.34 8.35 8.36 8.37 8.38
8.39 8.40 8.41 8.42 8.43 8.44 8.45 9.16
rsh decker vol options tpccvol13 nosnap on
rsh decker vol options tpccvol13 minra on
rsh decker vol options tpccvol13
no_atime_update on
rsh decker dafs export create tpccvol13
/vol/tpccvol13

rsh decker vol create tpccvol21 -r 27 -d 9.17
9.18 9.19 9.20 9.21 9.22 9.23 9.24 9.25 9.26
9.27 9.28 9.29 9.32 9.33 9.34 9.35 9.36 9.37
9.38 9.39 9.40 9.41 9.42 9.43 9.44 9.45
rsh decker vol options tpccvol21 nosnap on
rsh decker vol options tpccvol21 minra on
rsh decker vol options tpccvol21
no_atime_update on
rsh decker dafs export create tpccvol21
/vol/tpccvol21

rsh quark vol create tpccvol6 -r 27 -d 7.16
7.17 7.18 7.19 7.20 7.21 7.22 7.23 7.24 7.25
7.26 7.27 7.28 7.29 7.32 7.33 7.34 7.35 7.36
7.37 7.38 7.39 7.40 7.41 7.42 7.43 7.44
rsh quark vol options tpccvol6 nosnap on
rsh quark vol options tpccvol6 minra on
rsh quark vol options tpccvol6 no_atime_update
on
rsh quark options dafs.enable on
rsh quark dafs start
rsh quark options dafs.auth_none.enable on
rsh quark dafs export create tpccvol6
/vol/tpccvol6
rsh quark options dafs.dynamic_nreq.enable off
rsh quark options dafs.max_requests 1000
rsh quark options dafs.idle_timeout 28800

rsh quark vol create tpccvol14 -r 27 -d 8.18
8.19 8.20 8.21 8.22 8.23 8.24 8.25 8.26 8.27
8.28 8.29 8.32 8.33 8.34 8.35 8.36 8.37 8.38
8.39 8.40 8.41 8.42 8.43 8.44 8.45 9.16
rsh quark vol options tpccvol14 nosnap on
rsh quark vol options tpccvol14 minra on
rsh quark vol options tpccvol14
no_atime_update on
rsh quark dafs export create tpccvol14
/vol/tpccvol14

rsh quark vol create tpccvol22 -r 27 -d 9.17
9.18 9.19 9.20 9.21 9.22 9.23 9.24 9.25 9.26
9.27 9.28 9.29 9.32 9.33 9.34 9.35 9.36 9.37
9.38 9.39 9.40 9.41 9.42 9.43 9.44 9.45
rsh quark vol options tpccvol22 nosnap on
rsh quark vol options tpccvol22 minra on
rsh quark vol options tpccvol22
no_atime_update on
rsh quark dafs export create tpccvol22
/vol/tpccvol22

rsh laforge vol create tpccvol7 -r 27 -d 7.16
7.17 7.18 7.19 7.20 7.21 7.22 7.23 7.24 7.25
7.26 7.27 7.28 7.29 7.32 7.33 7.34 7.35 7.36
7.37 7.38 7.39 7.40 7.41 7.42 7.43 7.44
rsh laforge vol options tpccvol7 nosnap on
rsh laforge vol options tpccvol7 minra on
rsh laforge vol options tpccvol7
no_atime_update on
rsh laforge options dafs.enable on
```

```
rsh laforge dafs start
rsh laforge options dafs.auth_none.enable on
rsh laforge dafs export create tpccvol7
/vol/tpccvol7
rsh laforge options dafs.dynamic_nreq.enable
off
rsh laforge options dafs.max_requests 1000
rsh laforge options dafs.idle_timeout 28800

rsh laforge vol create tpccvol15 -r 27 -d 8.18
8.19 8.20 8.21 8.22 8.23 8.24 8.25 8.26 8.27
8.28 8.29 8.32 8.33 8.34 8.35 8.36 8.37 8.38
8.39 8.40 8.41 8.42 8.43 8.44 8.45 9.16
rsh laforge vol options tpccvol15 nosnap on
rsh laforge vol options tpccvol15 minra on
rsh laforge vol options tpccvol15
no_atime_update on
rsh laforge dafs export create tpccvol15
/vol/tpccvol15

rsh laforge vol create tpccvol23 -r 27 -d 9.17
9.18 9.19 9.20 9.21 9.22 9.23 9.24 9.25 9.26
9.27 9.28 9.29 9.32 9.33 9.34 9.35 9.36 9.37
9.38 9.39 9.40 9.41 9.42 9.43 9.44 9.45
rsh laforge vol options tpccvol23 nosnap on
rsh laforge vol options tpccvol23 minra on
rsh laforge vol options tpccvol23
no_atime_update on
rsh laforge dafs export create tpccvol23
/vol/tpccvol23

rsh bashir vol create tpccvol8 -r 27 -d 7.16
7.17 7.18 7.19 7.20 7.21 7.22 7.23 7.24 7.25
7.26 7.27 7.28 7.29 7.32 7.33 7.34 7.35 7.36
7.37 7.38 7.39 7.40 7.41 7.42 7.43 7.44
rsh bashir vol options tpccvol8 nosnap on
rsh bashir vol options tpccvol8 minra on
rsh bashir vol options tpccvol8
no_atime_update on
rsh bashir options dafs.enable on
rsh bashir dafs start
rsh bashir options dafs.auth_none.enable on
rsh bashir dafs export create tpccvol8
/vol/tpccvol8
rsh bashir options dafs.dynamic_nreq.enable
off
rsh bashir options dafs.max_requests 1000
rsh bashir options dafs.idle_timeout 28800

rsh bashir vol create tpccvol16 -r 27 -d 8.18
8.19 8.20 8.21 8.22 8.23 8.24 8.25 8.26 8.27
8.28 8.29 8.32 8.33 8.34 8.35 8.36 8.37 8.38
8.39 8.40 8.41 8.42 8.43 8.44 8.45 9.16
rsh bashir vol options tpccvol16 nosnap on
rsh bashir vol options tpccvol16 minra on
rsh bashir vol options tpccvol16
no_atime_update on
rsh bashir dafs export create tpccvol16
/vol/tpccvol16

rsh bashir vol create tpccvol24 -r 27 -d 9.17
9.18 9.19 9.20 9.21 9.22 9.23 9.24 9.25 9.26
9.27 9.28 9.29 9.32 9.33 9.34 9.35 9.36 9.37
9.38 9.39 9.40 9.41 9.42 9.43 9.44 9.45
rsh bashir vol options tpccvol24 nosnap on
rsh bashir vol options tpccvol24 minra on
rsh bashir vol options tpccvol24
no_atime_update on
rsh bashir dafs export create tpccvol24
/vol/tpccvol24
```

generic_procs.sh

```
#!/bin/sh -f
isql -Usa -P$PASSWORD <<EOF
use ${1:-tpcc}
go

if exists (select * from sysobjects where name
= “indexes” and type = ‘P’)
       drop proc indexes
go
if exists (select * from sysobjects where name
= “rowsize” and type = ‘P’)
       drop proc rowsize
go
if exists (select * from sysobjects where name
= “spaceused” and type = ‘P’)
       drop proc spaceused
go
if exists (select * from sysobjects where name
= “columns” and type = ‘P’)
       drop proc columns
go

create proc indexes as
       select       name, indid, ipgtrips,
status2
              from   sysindexes
              where  id > 1000
                     and indid > 0
go

create proc rowsize @table_name varchar(30) =
NULL as
       select
              o.name,
              bytes = sum(c.length),
              rowsPerPage =
```

```
2016/(sum(c.length)+4)
	from
		sysobjects o, syscolumns c
	where
		(@table_name = NULL OR o.name =
@table_name) AND
		o.type = 'U' AND
		o.id>999 AND
		o.id = c.id
	group by
		o.id
go

create proc spaceused as
declare@table_namechar(30)
declare@eot_so	int
declare c_msu_so cursor for
	select name
		from sysobjects
		where  id>9999 and
			type = 'U'
begin
	open c_msu_so
	select @eot_so = 0
	while (@eot_so = 0) begin
		fetch c_msu_so into @table_name
		if (@@sqlstatus > 0) begin
			select @eot_so = 1
			break
		end
		exec sp_spaceused @table_name
	end
end
go

create proc columns @table_name varchar(30) as
	select
		c.name, c.length, c.type, c.prec,
c.scale
	from
		sysobjects o, syscolumns c
	where
		o.name = @table_name AND
		o.type = 'U' AND
		o.id = c.id
go
EOF
```

msv_rpc_attach.sh

```
#!/bin/sh -f

isql -Usa -P$PASSWORD << EOF

/*
**	msv_rpc_attach.sql9.112/30/93
**
**	MSV_RPC_ATTACH.SQL
*/

/*
** stored procedure which returns one row con-
taining kernel start addr,
** kernel size, shmem flags and memory pool
table address
*/

if exists(select * from sysobjects
		where sysstat & 7 = 4
			and name =
'mon_rpc_attach')
begin
	drop procedure mon_rpc_attach
end
go
create procedure mon_rpc_attach as
begin
	 select @@kernel_addr, @@kernel_size,
@@shmem_flags, @@mempool_addr
return 0
end
go
grant execute on mon_rpc_attach to public
go
EOF
```

run_server.sh

```
#!/bin/sh

# Run the setup script if not already done
if [ -z "$SETUP_RUN" ]
then
	. ./bm_setup
fi

# The default dataserver is the one from the
release bin.

dataserver=$SQL_RELEASE/bin/dataserver
server_options=" -T1131 -T1231 -T3444 -T4057 -
T7821 -c./run.cfg"


# Do we override this ?
if [ $# != 0 ]
then
	if [ "$1" != "-" ]
```

```
        then
                dataserver=$1
        fi

        # Pick up the remaining arguments.
        shift
        server_options=$server_options" $*"
fi

# echo $dataserver -d$MASTER_DEVICE -
M/sybase/sybase12.0 $server_options
# exec $dataserver -d$MASTER_DEVICE -
M/sybase/sybase12.0 $server_options &
# echo $dataserver -d$MASTER_DEVICE -
M/mnt/sybase/ase12.5.64bit $server_options
# exec $dataserver -d$MASTER_DEVICE -
M/mnt/sybase/ase12.5.64bit $server_options &
echo $dataserver -d$MASTER_DEVICE -
M/mnt/sybase/12.5IR/home $server_options
exec $dataserver -d$MASTER_DEVICE -
M/mnt/sybase/12.5IR/home $server_options &
```

shutdown_server.sh

```
#!/bin/sh

# Run the setup script if not already done
if [ -z "$SETUP_RUN" ]
then
        . ./bm_setup
fi

# The default dataserver is the one from the
release bin.

dataserver=$SQL_RELEASE/bin/dataserver
server_options=" -T1131 -T1231 -T3444 -T4057 -
T7821 -c./run.cfg"


# Do we override this ?
if [ $# != 0 ]
then
        if [ "$1" != "-" ]
        then
                dataserver=$1
        fi

        # Pick up the remaining arguments.
        shift
        server_options=$server_options" $*"
fi

# echo $dataserver -d$MASTER_DEVICE -
M/sybase/sybase12.0 $server_options
# exec $dataserver -d$MASTER_DEVICE -
M/sybase/sybase12.0 $server_options &
# echo $dataserver -d$MASTER_DEVICE -
M/mnt/sybase/ase12.5.64bit $server_options
# exec $dataserver -d$MASTER_DEVICE -
M/mnt/sybase/ase12.5.64bit $server_options &
echo $dataserver -d$MASTER_DEVICE -
M/mnt/sybase/12.5IR/home $server_options
exec $dataserver -d$MASTER_DEVICE -
M/mnt/sybase/12.5IR/home $server_options &
```

spads_admin.sh

```
#!/bin/sh -f

dbname=${1:-tpcc}

#cat << EOF
isql -Usa -P$PASSWORD <<EOF
use $dbname
go


/* This table is used to insert TPCC data into
SpaDS */
if exists (select * from sysobjects where name
= 'spads_summary' and type = 'U')
        drop table spads_summary
go

create table spads_summary
(
        no_permin               decimal(7,3)
null,
        ptid_permin             decimal(7,3)
null,
        ptname_permin           decimal(7,3)
null,
        osid_permin             decimal(7,3)
null,
        osname_permin           decimal(7,3)
null,
        delivery_permin         decimal(7,3)
null,
        stock_permin            decimal(7,3)
null,
        no_response             decimal(7,3)
null,
        ptid_response           decimal(7,3)
null,
        ptname_reponse          decimal(7,3)
null,
        osid_response           decimal(7,3)
```

```
null,
        osname_reponse          decimal(7,3)
null,
        delivery_response       decimal(7,3)
null,
        stock_response          decimal(7,3)
null,
        no_variance             decimal(7,3)
null,
        ptid_variance           decimal(7,3)
null,
        ptname_variance         decimal(7,3)
null,
        osid_variance           decimal(7,3)
null,
        osname_variance         decimal(7,3)
null,
        delivery_variance       decimal(7,3)
null,
        stock_variance          decimal(7,3)
null,
        no_limit                decimal(7,3)
null,
        ptid_limit              decimal(7,3)
null,
        ptname_limit            decimal(7,3)
null,
        osid_limit              decimal(7,3)
null,
        osname_limit            decimal(7,3)
null,
        delivery_limit          decimal(7,3)
null,
        stock_limit             decimal(7,3)
null,
        no_prcover              decimal(7,3)
null,
        ptid_prcover            decimal(7,3)
null,
        ptname_prcover          decimal(7,3)
null,
        osid_prcover            decimal(7,3)
null,
        osname_prcover          decimal(7,3)
null,
        delivery_prcover        decimal(7,3)
null,
        stock_prcover           decimal(7,3)
null,
        no_deadlock             decimal(7,3)
null,
        ptid_deadlock           decimal(7,3)
null,
        ptname_deadlock         decimal(7,3)
null,
        osid_deadlock           decimal(7,3)
null,
        osname_deadlock         decimal(7,3)
null,
        delivery_deadlock       decimal(7,3)
null,
        stock_deadlock          decimal(7,3)
null,
        no_prcmix               decimal(7,3)
null,
        ptid_prcmix             decimal(7,3)
null,
       ptname_prcmix            decimal(7,3)
null,
        osid_prcmix             decimal(7,3)
null,
        osname_prcmix           decimal(7,3)
null,
        delivery_prcmix         decimal(7,3)
null,
        stock_prcmix            decimal(7,3)
null

)
go

if exists (select * from sysobjects where name
= 'insert_summary' and type = 'P')
        drop proc insert_summary
go

create proc insert_summary
as

        declare @no_permin              deci-
mal(7,3)
        declare @ptid_permin            deci-
mal(7,3)
        declare @ptname_permin          deci-
mal(7,3)
        declare @osid_permin            deci-
mal(7,3)
        declare @osname_permin          deci-
mal(7,3)
        declare @delivery_permin        deci-
mal(7,3)
        declare @stock_permin           deci-
mal(7,3)
        declare @no_response            deci-
mal(7,3)
        declare @ptid_response          deci-
mal(7,3)
        declare @ptname_response         dec-
imal(7,3)
        declare @osid_response          deci-
```

```
mal(7,3)
        declare @osname_response          dec-
imal(7,3)
        declare @delivery_response        deci-
mal(7,3)
        declare @stock_response           deci-
mal(7,3)
        declare @no_variance              deci-
mal(7,3)
        declare @ptid_variance            deci-
mal(7,3)
        declare @ptname_variance          deci-
mal(7,3)
        declare @osid_variance            deci-
mal(7,3)
        declare @osname_variance          deci-
mal(7,3)
        declare @delivery_variance        deci-
mal(7,3)
        declare @stock_variance           deci-
mal(7,3)
        declare @no_limit                 deci-
mal(7,3)
        declare @ptid_limit               deci-
mal(7,3)
        declare @ptname_limit             deci-
mal(7,3)
        declare @osid_limit               deci-
mal(7,3)
        declare @osname_limit             deci-
mal(7,3)
        declare @delivery_limit           deci-
mal(7,3)
        declare @stock_limit              deci-
mal(7,3)
        declare @no_prcover               deci-
mal(7,3)
        declare @ptid_prcover             deci-
mal(7,3)
        declare @ptname_prcover           deci-
mal(7,3)
        declare @osid_prcover             deci-
mal(7,3)
        declare @osname_prcover           deci-
mal(7,3)
        declare @delivery_prcover         deci-
mal(7,3)
        declare @stock_prcover            deci-
mal(7,3)
        declare @no_deadlock              deci-
mal(7,3)
        declare @ptid_deadlock            deci-
mal(7,3)
        declare @ptname_deadlock          deci-
mal(7,3)
        declare @osid_deadlock            deci-
mal(7,3)
        declare @osname_deadlock          deci-
mal(7,3)
        declare @delivery_deadlock        deci-
mal(7,3)
        declare @stock_deadlock           deci-
mal(7,3)
        declare @no_prcmix                deci-
mal(7,3)
        declare @ptid_prcmix              deci-
mal(7,3)
        declare @ptname_prcmix            deci-
mal(7,3)
       declare @osid_prcmix              deci-
mal(7,3)
        declare @osname_prcmix            deci-
mal(7,3)
        declare @delivery_prcmix          deci-
mal(7,3)
        declare @stock_prcmix             deci-
mal(7,3)

        declare @alltotal                 float
        declare @minutes                  float


begin


       select @alltotal = sum(total)
        from tc_stattab
        where not name = “delivery”

        select @minutes = convert(float,dura-
tion)/60.0 from tc_infotab


        /* new_order */
        select  @no_permin=convert(deci-
mal(6,0),total/@minutes),
               @no_response=convert(deci-
mal(6,3),sumResp/(total*1000.0)),
               @no_variance=convert(deci-
mal(6,3),
                           (sumRespSq-sum-
Resp*sumResp/total)/(total*1000000.0)),
              @no_limit=convert(decimal(6,3),
convert(float,  limit)/1000,0),
               @no_prcover=convert(deci-
mal(6,3),
                               (convert(float,
overlimit)*100.0)/total),
               @no_deadlock=convert(deci-
mal(4,0),deadlock),
```

```
                @no_prcmix=convert(deci-
mal(5,2),total/@alltotal*100.0)
                from tc_stattab
                where name=”new_order”
                and total > 0

        /* payment_byid */
        select  @ptid_permin=convert(deci-
mal(6,0),total/@minutes),
                @ptid_response=convert(deci-
mal(6,3),sumResp/(total*1000.0)),
                @ptid_variance=convert(deci-
mal(6,3),
                                       (sumRespSq-
sumResp*sumResp/total)/(total*1000000.0)),
                @ptid_limit=convert(deci-
mal(6,3), convert(float, limit)/1000,0),
                @ptid_prcover=convert(deci-
mal(6,3),
                                (convert(float,
overlimit)*100.0)/total),
                @ptid_deadlock=convert(deci-
mal(4,0),deadlock),
                @ptid_prcmix=convert(deci-
mal(5,2),total/@alltotal*100.0)
                from tc_stattab
                where name=”payment_byid”
                and total > 0

        /* payment_byname */
        select  @ptname_permin=convert(deci-
mal(6,0),total/@minutes),
                @ptname_response=convert(deci-
mal(6,3),sumResp/(total*1000.0)),
                @ptname_variance=convert(deci-
mal(6,3),
                                       (sumRespSq-
sumResp*sumResp/total)/(total*1000000.0)),
                @ptname_limit=convert(deci-
mal(6,3), convert(float, limit)/1000,0),
                @ptname_prcover=convert(deci-
mal(6,3),
                                (convert(float,
overlimit)*100.0)/total),
                @ptname_deadlock=convert(deci-
mal(4,0),deadlock),
                @ptname_prcmix=convert(deci-
mal(5,2),total/@alltotal*100.0)
                from tc_stattab
                where name=”payment_byname”
                and total > 0

       /* order_status_byid */
        select  @osid_permin=convert(deci-
mal(6,0),total/@minutes),
                @osid_response=convert(deci-
mal(6,3),sumResp/(total*1000.0)),
                @osid_variance=convert(deci-
mal(6,3),
                                       (sumRespSq-
sumResp*sumResp/total)/(total*1000000.0)),
                @osid_limit=convert(deci-
mal(6,3), convert(float, limit)/1000,0),
                @osid_prcover=convert(deci-
mal(6,3),
                                (convert(float,
overlimit)*100.0)/total),
                @osid_deadlock=convert(deci-
mal(4,0),deadlock),
                @osid_prcmix=convert(deci-
mal(5,2),total/@alltotal*100.0)
                from tc_stattab
                where name=”order_status_byid”
                and total > 0


        /* order_status byname */
        select  @osname_permin=convert(deci-
mal(6,0),total/@minutes),
                @osname_response=convert(deci-
mal(6,3),sumResp/(total*1000.0)),
                @osname_variance=convert(deci-
mal(6,3),
                                       (sumRespSq-
sumResp*sumResp/total)/(total*1000000.0)),
                @osname_limit=convert(deci-
mal(6,3), convert(float, limit)/1000,0),
                @osname_prcover=convert(deci-
mal(6,3),
                                (convert(float,
overlimit)*100.0)/total),
                @osname_deadlock=convert(deci-
mal(4,0),deadlock),
                @osname_prcmix=convert(deci-
mal(5,2),total/@alltotal*100.0)
                from tc_stattab
                where
name=”order_status_byname”
                and total > 0

        /* delivery */
        select  @delivery_permin=convert(deci-
mal(6,0),total/@minutes),
               @delivery_response=convert(dec-
imal(6,3),sumResp/(total*1000.0)),
               @delivery_variance=convert(dec-
imal(6,3),
                                       (sumRespSq-
sumResp*sumResp/total)/(total*1000000.0)),
                @delivery_limit=convert(deci-
```

```
mal(6,3), convert(float, limit)/1000,0),
                @delivery_prcover=convert(dec-
imal(6,3),
                                (convert(float,
overlimit)*100.0)/total),
               @delivery_deadlock=convert(dec-
imal(4,0),deadlock),
                @delivery_prcmix=convert(deci-
mal(5,2),total/@alltotal*100.0)
                from tc_stattab
                where name=”delivery”
                and total > 0

        /* stock_level */
         select  @stock_permin=convert(deci-
mal(6,0),total/@minutes),
                @stock_response=convert(deci-
mal(6,3),sumResp/(total*1000.0)),
                @stock_variance=convert(deci-
mal(6,3),
                                    (sumRespSq-
sumResp*sumResp/total)/(total*1000000.0)),
                @stock_limit=convert(deci-
mal(6,3), convert(float, limit)/1000,0),
                @stock_prcover=convert(deci-
mal(6,3),
                                (convert(float,
overlimit)*100.0)/total),
                @stock_deadlock=convert(deci-
mal(4,0),deadlock),
                @stock_prcmix=convert(deci-
mal(5,2),total/@alltotal*100.0)
                from tc_stattab
                where name=”stock_level”
                and total > 0


       insert into spads_summary values(
         @no_permin,
         @ptid_permin,
         @ptname_permin,
         @osid_permin,
         @osname_permin,
         @delivery_permin,
         @stock_permin,
         @no_response,
         @ptid_response,
         @ptname_response,
         @osid_response,
         @osname_response,
         @delivery_response,
         @stock_response,
         @no_variance,
         @ptid_variance,
         @ptname_variance,
         @osid_variance,
         @osname_variance,
         @delivery_variance,
         @stock_variance,
         @no_limit,
         @ptid_limit,
         @ptname_limit,
         @osid_limit,
         @osname_limit,
         @delivery_limit,
         @stock_limit,
         @no_prcover,
         @ptid_prcover,
         @ptname_prcover,
         @osid_prcover,
         @osname_prcover,
         @delivery_prcover,
         @stock_prcover,
         @no_deadlock,
         @ptid_deadlock,
         @ptname_deadlock,
         @osid_deadlock,
         @osname_deadlock,
         @delivery_deadlock,
         @stock_deadlock,
         @no_prcmix,
         @ptid_prcmix,
         @ptname_prcmix,
         @osid_prcmix,
         @osname_prcmix,
         @delivery_prcmix,
         @stock_prcmix

)

end

go


if exists (select * from sysobjects where name
= ‘tc_synctab’ and type = ‘U’)
       drop table tc_synctab
go
create table tc_synctab
(
       junk         int     default 0not null,
)
go

if exists (select * from sysobjects where name
= ‘tc_initialize’ and type = ‘P’)
       drop proc tc_initialize
go
create proc tc_initialize
```

```
       @clientsint,
       @durationint = 600,
       @enginesint = 0,
       @warehousesint = 0,
       @deliveryRatioint
as
declare@availwareint
declare@online      int

       select @availware = count(*) from ware-
house

       if (@availware < @warehouses or @ware-
houses = 0)
              select @warehouses = @availware

       select @online = count(*)
              from master..sysengines
              where status = “online”
       while (@engines > 0 and @engines >
@online)
       begin
              dbcc engine(“online”)
              select @online = @online + 1
       end
       while (@engines > 0 and @engines <
@online)
       begin
              dbcc engine(offline)
              select @online = @online + 1
       end

       begin transaction
       delete from tc_infotab
       insert into tc_infotab( clients,
arrived,  duration,
                              engines,  ware-
houses, deliveryRatio)
       values(@clients,       1, @duration,
@online, @warehouses, @deliveryRatio)
       commit transaction

       print “Using %1! warehouses (out of
%2!), %3! engines, and %4! users for %5! sec-
onds”,
                     @warehouses, @availware,
@online, @clients, @duration

       begin transaction
       delete from tc_synctab
       insert into tc_synctab(junk) values (0)
       commit transaction

       begin transaction
       delete from tc_stattab
       delete from spads_summary
       commit transaction

       select engines, warehouses from
tc_infotab
go

EOF
```

tpcc_admin.sh

```
#!/bin/sh -f

dbname=${1:-tpcc}

#cat << EOF
isql -Usa -P$PASSWORD <<EOF
use $dbname
go


if exists (select * from sysobjects where name
= ‘tc_infotab’ and type = ‘U’)
        drop table tc_infotab
go
create table tc_infotab
(
        clients         int     not null,
        duration        int     not null,
        engines         int     not null,
        warehouses      int     not null,
        arrived         int     not null,
        deliveryRatio   int     not null
)
go

if exists (select * from sysobjects where name
= ‘tc_stattab’ and type = ‘U’)
        drop table tc_stattab
go
create table tc_stattab
(
        name            char(30)
not null,
        total           int     default 0
not null,
        limit           int     default 0
not null,
        overlimit       int     default 0
not null,
        deadlock        int     default 0
not null,
        sumResp         float   default 0
not null,
```

```
       sumRespSq       float    default 0
not null,
       filler1         char(255) default “”
not null,
       filler2         char(255) default “”
not null,
       filler3         char(255) default “”
not null,
       filler4         char(255) default “”
not null,

        unique(name)
)
go

if exists (select * from sysobjects where name
= ‘tc_synctab’ and type = ‘U’)
      drop table tc_synctab
go
create table tc_synctab
(
      junk          int     default 0not null,
)
go

if exists (select * from sysobjects where name
= ‘tc_initialize’ and type = ‘P’)
      drop proc tc_initialize
go
create proc tc_initialize
      @clientsint,
      @durationint = 600,
      @enginesint = 0,
      @warehousesint = 0,
      @deliveryRatioint
as
declare@availwareint
declare@online      int

      select @availware = count(*) from ware-
house

      if (@availware < @warehouses or @ware-
houses = 0)
            select @warehouses = @availware

      select @online = count(*)
            from master..sysengines
            where status = “online”
      while (@engines > 0 and @engines >
@online)
      begin
            dbcc engine(“online”)
            select @online = @online + 1
      end

      while (@engines > 0 and @engines <
@online)
      begin
            dbcc engine(offline)
            select @online = @online + 1
      end

      begin transaction
      delete from tc_infotab
      insert into tc_infotab( clients,
arrived,  duration,
                              engines,  ware-
houses, deliveryRatio)
      values(@clients,       1, @duration,
@online, @warehouses, @deliveryRatio)
      commit transaction

      print “Using %1! warehouses (out of
%2!), %3! engines, and %4! users for %5! sec-
onds”,
                  @warehouses, @availware,
@online, @clients, @duration

      begin transaction
      delete from tc_synctab
      insert into tc_synctab(junk) values (0)
      commit transaction

      begin transaction
      delete from tc_stattab
      commit transaction

      select engines, warehouses from
tc_infotab
go

if exists (select * from sysobjects where name
= ‘tc_initstat’ and type = ‘P’)
      drop proc tc_initstat
go
create proc tc_initstat
      @name  char(30),
      @limit int
as
begin
      begin transaction
      insert into tc_stattab(name,limit) val-
ues (@name, @limit)
      commit transaction
end
go

if exists (select * from sysobjects where name
= ‘tc_startup’ and type = ‘P’)
      drop proc tc_startup
```

```
go
create proc tc_startup
as
declare@junk        int
begin
       update tc_infotab set arrived =
arrived+1
       select @junk=junk from tc_synctab
       select engines, warehouses from
tc_infotab
end
go

if exists (select * from sysobjects where name
= ‘tc_addstats’ and type = ‘P’)
       drop proc tc_addstats
go
create proc tc_addstats
       @name        char(30),
       @total       int,
       @overlimitint,
       @deadlockint,
       @sumRespfloat,
       @sumRespSqfloat
as
begin
       begin transaction
       update tc_stattab set
              total    = total+ @total,
              overlimit = overlimit+ @over-
limit,
              deadlock  = deadlock+ @deadlock,
              sumResp  = sumResp+ @sumResp,
              sumRespSq = sumRespSq+ @sumRe-
spSq
       where name = @name
       commit transaction
end
go

if exists (select * from sysobjects where name
= ‘showstats’ and type = ‘P’)
       drop proc showstats
go
create proc showstats
as
declare@alltotalfloat,
       @minutesfloat
begin
       select @alltotal = sum(total)
              from tc_stattab
              where not name = “delivery”
       select @minutes = convert(float,dura-
tion)/60.0 from tc_infotab
       select
              convert(char(20), name) as Name,
              convert(decimal(6,0),total/@min-
utes) as PerMin,
              convert(decimal(6,3),sum-
Resp/(total*1000.0)) as Average,
              convert(decimal(6,3),
                     (sumRespSq-sumResp*sum-
Resp/total)/(total*1000000.0))
                     as Variance,
              convert(decimal(6,3), con-
vert(float, limit)/1000.0) as Limit,
              convert(decimal(6,3),
                     (convert(float, over-
limit)*100.0)/total) as PercOver,
              convert(decimal(4,0),deadlock)
as Deadlocks,
              convert(decimal(5,2),total/@allto-
tal*100.0) as PercMix
           fromtc_stattab
           wheretotal>0
end
go

if exists (select * from sysobjects where name
= ‘spread’)
       drop view spread
go
create view spread as
select       d_w_id ware,
              d_id dist,
              (select min(no_o_id) from
new_order
                     where  no_w_id = d.d_w_id
                     and    no_d_id = d.d_id)
first,
       d_next_o_id next
       from district d
go
EOF
```

tpcc_cache_bind.sh

```
#!/usr/bin/sh -f

isql -Usa -P$PASSWORD -e << EOF

use master
```

```
go
sp_dboption tpcc, “single user”, true
go

use tpcc
go
checkpoint
go

/*
** Cache c_log
*/

sp_bindcache “c_log”, “tpcc”, “syslogs”
go

sp_bindcache “c_wid”, “tpcc”, “sysindexes”
go
sp_bindcache “c_wid”, “tpcc”, “sysindexes”,
“sysindexes”
go

use master
go
sp_dboption tpcc, “single user”, false
go

use tpcc
go
checkpoint
go

sp_bindcache “c_wid”, “tpcc”, “item”
go
sp_bindcache “c_wid”, “tpcc”, “item”, “i_clu”
go
sp_bindcache “c_wid”, “tpcc”, “district”
go
sp_bindcache “c_wid”, “tpcc”, “district”,
“d_clu”
go
sp_bindcache “c_wid”, “tpcc”, “warehouse”
go
sp_bindcache “c_wid”, “tpcc”, “warehouse”,
“w_clu”
go

/*
** Cache New Order
*/

sp_bindcache “c_no”, “tpcc”, “new_order”
go
sp_bindcache “c_no_order_index”, “tpcc”,
“new_order”, “no_clu”
go


/*
** Cache Order Line
*/
sp_bindcache “c_ol”, “tpcc”, “order_line”
go
sp_bindcache “c_ol_index”, “tpcc”,
“order_line”, “ol_clu”
go

/*
** Cache orders
*/

sp_bindcache “c_orders”, “tpcc”, “orders”
go
sp_bindcache “c_no_order_index”, “tpcc”,
“orders”, “o_clu”
go

/*
** Cache c_stock_index
*/

sp_bindcache “c_stock_index”, “tpcc”, “stock”,
“s_clu”
go

/*
** Cache c_stock
*/

sp_bindcache “c_stock”, “tpcc”, “stock”
go

/*
** Cache c_customer
*/

sp_bindcache “c_customer”, “tpcc”, “customer”
go

/*
** Cache c_customer_index
*/

sp_bindcache “c_cust_index”, “tpcc”, “cus-
tomer”, “c_clu”
go

/*
** Cache c_customer_non_index
*/
```

```
sp_bindcache “c_cust_non_index1”, “tpcc”,
“customer”, “c_non1”
go

/*
** Default cache
*/

sp_bindcache “default data cache”, “tpcc”,
“history”
go
EOF
```

tpcc_ext_log.sh

```
#Created to properly create the log devices.
# DCM 1/8/02

isql -Usa -P << EOF
use master
go
alter database tpcc
log on tpcc_log1=”32000M”
go
alter database tpcc
log on tpcc_log2=”32000M”
go
use tpcc
go
sp_helpdb tpcc
go
checkpoint
go
EOF
```

tpcc_ext_scache.sh

```
#Created to extend tempdb off master device
# RR 6/27/01

isql -Usa -P << EOF
use master
go
disk init
name = tpcc_scache,
physname = ‘/mnt/tpccvol14/tpcc_scache’,
size = 1024000,
vdevno = 253
go
use tpcc
go
sp_extendsegment Scache, tpcc, tpcc_scache
go
sp_dropsegment “default”,tpcc,tpcc_scache
go
checkpoint
go
EOF
```

tpcc_ext_tempdb.sh

```
#Created to extend tempdb off master device
# RR 6/27/01

isql -Usa -P << EOF
use master
go
disk init
name = tpcc_temp3,
physname = ‘/mnt/tpccvol20/tpcc_temp1’,
size = 1024000,
vdevno = 245
go
disk init
name = tpcc_temp4,
physname = ‘/mnt/tpccvol19/tpcc_temp2’,
size = 1024000,
vdevno = 246
go
alter database tempdb
on tpcc_temp3=”2000M”, tpcc_temp4=”2000M”
go
use tempdb
go
sp_dropsegment “default”,tempdb,master
go
sp_dropsegment logsegment,tempdb,master
go
sp_dropsegment system,tempdb,master
go
use tempdb
go
sp_helpdb tempdb
go
checkpoint
go
EOF
```

tpcc_indexes_parallel.sh

```
#!/bin/sh -f

isql -Usa -P$PASSWORD << EOF
```

```
/* This script will create the TPC-C indexes
that are best
   created after the load. */
use tpcc
go

alter table customer unpartition
go

select getdate()
go

create unique clustered index c_clu
       on customer(c_w_id, c_id, c_d_id)
       with sorted_data
       on Scustomer
go

select getdate()
go

create unique nonclustered index c_non1
        on customer(c_w_id, c_d_id, c_last,
c_first, c_id)
        on Scustomer
go
sp_spaceused customer, 1
go

select getdate()
go

alter table new_order unpartition
go
create unique clustered index no_clu
       on new_order(no_w_id, no_d_id, no_o_id)
       with sorted_data
       on Snew_order
go
dbcc tune(ascinserts, 1, new_order)
go
dbcc tune(oamtrips, 100, new_order)
go
sp_spaceused new_order, 1
go

select getdate()
go

alter table orders unpartition
go
create unique clustered index o_clu
       on orders(o_w_id, o_d_id, o_id)
       with sorted_data
       on Sorders
go
dbcc tune(ascinserts, 1, orders)
go
dbcc tune(oamtrips, 100, orders)
go
sp_spaceused orders, 1
go

select getdate()
go

alter table order_line unpartition
go
create unique clustered index ol_clu
       on order_line(ol_w_id, ol_d_id,
ol_o_id, ol_number)
       with sorted_data
       on Sorder_line
go
dbcc tune(ascinserts, 1, order_line)
go
dbcc tune(oamtrips, 100, order_line)
go
sp_spaceused order_line, 1
go

select getdate()
go

alter table item unpartition
go
create unique clustered index i_clu
       on item(i_id)
       with sorted_data
       on Sitem
go
dbcc tune(indextrips, 10, item)
go
sp_spaceused item, 1
go

select getdate()
go

alter table stock unpartition
go
create unique clustered index s_clu
       on stock(s_i_id, s_w_id)
       with sorted_data
       on Sstock
go
dbcc tune(indextrips, 10,stock)
go
sp_spaceused stock, 1
go
```

```
select getdate()
go

create unique clustered index w_clu
        on warehouse(w_id)
        with sorted_data
        on Swarehouse
go
dbcc tune(indextrips, 100,warehouse)
go
sp_spaceused warehouse, 1
go

select getdate()
go

create unique clustered index d_clu
        on district(d_w_id, d_id)
        with sorted_data
        on Sdistrict
go
dbcc tune(indextrips, 100,district)
go
sp_spaceused district, 1
go

select getdate()
go

checkpoint
go
EOF
```

tpcc_indextrips.sh

```
#!/bin/sh -f

isql -Usa -P$PASSWORD << EOF
use tpcc
go
dbcc tune(indextrips, 100, warehouse)
dbcc tune(indextrips, 100, district)
dbcc tune(indextrips, 100, item)
EOF
```

tpcc_largeio.sh

```
#!/bin/sh -f

isql -Usa -P $PASSWORD << EOF

use tpcc
go

dbcc iosize(“tpcc”, “orders”, 16)
go
dbcc iosize(“tpcc”, “order_line”, 16)
go
dbcc iosize(“tpcc”, “customer”, 16)
go

EOF
```

tpcc_load_parallel.sh

```
#!/bin/sh -f
partition=${3:--1}
load stock $1 $2 “$PASSWORD” $partition
echo “done with stock table “
date
load customer $1 $2 “$PASSWORD” $partition
echo “done with customer table “
date
load new_order $1 $2 “$PASSWORD” $partition
echo “done with new_order table “
date
load warehouse $1 $2 “$PASSWORD”
echo “done with warehouse table “
date
load district $1 $2 “$PASSWORD”
echo “done with district table “
date
load item 1 1 “$PASSWORD” $partition
echo “done with item table “
date
load orders $1 $2 “$PASSWORD” $partition
echo “done with orders table “
date
load history $1 $2 “$PASSWORD” 20
echo “done with history table “
date
wait
```

tpcc_nfs_2_dafs

```
#!/bin/ksh
#
# Rename all the files in given directories
from file to .file and
# do ddafs open on them.
#
# Takes a list of directory names as input or
uses default list below.
#
#
DEF_DIRS="
  /mnt/logs
  /mnt/tpccvol1
  /mnt/tpccvol2
  /mnt/tpccvol3
  /mnt/tpccvol4
  /mnt/tpccvol5
  /mnt/tpccvol6
  /mnt/tpccvol7
  /mnt/tpccvol8
  /mnt/tpccvol9
  /mnt/tpccvol10
  /mnt/tpccvol11
  /mnt/tpccvol12
  /mnt/tpccvol13
  /mnt/tpccvol14
  /mnt/tpccvol15
  /mnt/tpccvol16
  /mnt/tpccvol17
  /mnt/tpccvol18
  /mnt/tpccvol19
  /mnt/tpccvol20
  /mnt/tpccvol21
  /mnt/tpccvol22
  /mnt/tpccvol23
  /mnt/tpccvol24
"

USAGE="Usage: $0 [-h | -n] [dir_names]
where
  dir_names - list of directory names in which
to convert files
              Defaults to "$DEF_DIRS".
  -h - print usage message
  -n - don't run cmds, just echo them
i.e.,
  $0 /tmp/mnt/tpccvol\*
"

CMD=""

if [ "$1" = "-h" ] ; then
       echo "$USAGE"
       exit 1
elif [ "$1" = "-n" ] ; then
       CMD=print    # don't really do it, just
echo cmds
       echo "Echo'ing commands, not really
running them."
       shift
fi

if [ $# -gt 0 ] ; then
       dirs=$*
else
       dirs="$DEF_DIRS"
fi

curdir='pwd'

proceed=false

for dir in $dirs
do
       cd $curdir

       # make sure it exists
       if [ ! -d $dir ] ; then
              echo "$dir doesn't exist, skip-
ping it."
              continue
       fi

       cd $dir
       files='ls tpcc*'

       # renaming all files is serious, verify
it
       if [ $proceed = true ]
       then
              echo Renaming files in $dir
       else
              print -n "Rename files in $dir?
(y/n/p) "
              read ans
              if [ "$ans" = "p" ] ; then
                     proceed=true
                     ans=y
              fi
              if [ "$ans" != "y" ] ; then
                     echo "skipping $dir"
                     continue
              fi
       fi

       # rename them
       for file in $files
       do
```

```
            if [ ! -f $file ] ; then
                  continue
            fi
            dotfile=".${file}"
            if [ -f $dotfile ] ; then
                  $CMD rm $dotfile
            fi
            $CMD mv $file $dotfile
        $CMD ddafs open -f d -v -N $dotfile
$file
        # $CMD ddafs open -f d -v $dotfile
$file
            $CMD chmod 777 $file
      done
done

ddafs list|wc
```

tpcc_proc.sh

```
#############################################
###################################
#
# tpcc_proc_case.sh
#
#############################################
###################################
#
# This is the version of procs which was used
in the Compaq-Sybase 11.G
# TPC-C benchmark (with the last-minute fixes)
- March 26 1997
#
# This case script has the following changes
from tpcc_proc_spec.sh
#   In new_order (both local and remote), the
stock-item cursor, c_no_is
#   has been removed and replaced with an
update-set-local variable stmt.
# Also CASE statements replace the nested
if's.
#
# Also modified delivery proc where the ol and
order table cursors have
#   been replaced by update_set_local_variable
statements.
#
# In Payment procs, the cursor on customer,
c_pay_c has been removed. Instead.
# added two update statements (with set local
variables).
#
# Reinstated c_find cursor to find cust_id
given c_last;
# Stock_level is back o its "single query"
state!
#
# IMPORTANT NOTE : THIS IS A NEW STORED PROCE-
DURE AND YET TO BE AUDITED.
#
#############################################
###################################
#
#!/bin/sh -f

# Stored procedure for TPC-C 3.2 on SQL Server
11.1 and later
# Copyright Sybase 1997
#
isql -Usa -P$PASSWORD <<EOF
use tpcc
go
if exists ( SELECT name FROM sysobjects WHERE
name = 'neworder_local')
      DROP PROC neworder_local
go

CREATE PROC neworder_local (
      @w_id        smallint,
      @d_id        tinyint,
      @c_id        int,
      @o_ol_cnttinyint,

      @i_id  int = 0, @ol_qtytinyint = 0,
      @i_id2 int = 0, @ol_qty2tinyint = 0,
      @i_id3 int = 0, @ol_qty3tinyint = 0,
      @i_id4 int = 0, @ol_qty4tinyint = 0,
      @i_id5 int = 0, @ol_qty5tinyint = 0,
      @i_id6 int = 0, @ol_qty6tinyint = 0,
      @i_id7 int = 0, @ol_qty7tinyint = 0,
      @i_id8 int = 0, @ol_qty8tinyint = 0,
      @i_id9 int = 0, @ol_qty9tinyint = 0,
      @i_id10int = 0, @ol_qty10tinyint = 0,
      @i_id11int = 0, @ol_qty11tinyint = 0,
      @i_id12int = 0, @ol_qty12tinyint = 0,
      @i_id13int = 0, @ol_qty13tinyint = 0,
      @i_id14int = 0, @ol_qty14tinyint = 0,
      @i_id15int = 0, @ol_qty15tinyint = 0
)
as

declare
      @w_tax       real,        @d_tax real,
      @c_last      char(16),@c_creditchar(2),
      @c_discountreal,    @commit_flagtiny-
int,
      @c_ins_id int,
```

```
      @i_pricereal,
      @i_name      char(24),@i_data
char(50),

      @s_quantitysmallint,
      @s_ytd       int,         @s_order_cnt
int,
      @s_dist      char(24),@s_data
char(50),

      @ol_numbertinyint,@o_id    int,
      @o_entry_ddatetime,@b_g    char(1),
      @ol_amount real



begin

      begin transaction NO

      -- @#@# UPDATE district FROM district,
warehouse, customer
      --

      UPDATE district
      SET      d_next_o_id = d_next_o_id + 1
             , @o_id       = d_next_o_id
             , @d_tax = d_tax
             , @commit_flag= 1
             , @ol_number= 0
             , @o_entry_d= getdate()
      WHERE          d_w_id = @w_id
             AND     d_id   = @d_id

      while (@ol_number < @o_ol_cnt) begin
             SELECT @ol_number = @ol_number +
1
             ,@i_id = case @ol_number
        when 1 then @i_id2
               when 2 then @i_id3
               when 3 then @i_id4
               when 4 then @i_id5
               when 5 then @i_id6
               when 6 then @i_id7
               when 7 then @i_id8
               when 8 then @i_id9
               when 9 then @i_id10
               when 10 then @i_id11
               when 11 then @i_id12
               when 12 then @i_id13
               when 13 then @i_id14
               when 14 then @i_id15
               else @i_id
               end
             , @ol_qty = case @ol_number
               when 1 then @ol_qty2
               when 2 then @ol_qty3
               when 3 then @ol_qty4
               when 4 then @ol_qty5
               when 5 then @ol_qty6
               when 6 then @ol_qty7
               when 7 then @ol_qty8
               when 8 then @ol_qty9
               when 9 then @ol_qty10
               when 10 then @ol_qty11
               when 11 then @ol_qty12
               when 12 then @ol_qty13
               when 13 then @ol_qty14
               when 14 then @ol_qty15
               else @ol_qty
               end

      /* set i_id, ol_qty for this lineitem */
      /* this is replaced by case statement */

      /* convert c_no_is cursor to a simple
select */
      /* get item data (no one update item) */

             select @i_price = i_price,
                    @i_name = i_name,
                    @i_data = i_data
               from item HOLDLOCK
              where i_id = @i_id

             if (@@rowcount = 0)
               begin
                 select @commit_flag = 0
                select NULL, NULL, NULL, NULL
                 continue
               end
         /*Otherwise if the item is found */
                update stock
                   set s_ytd = s_ytd + @ol_qty,
                     @ol_amount = @ol_qty *
@i_price,
                         @s_quantity =
s_quantity - @ol_qty +
                                  case when
(s_quantity - @ol_qty < 10)
                                  then 91 else 0
end,
                           s_quantity =
s_quantity - @ol_qty +
                                  case when
(s_quantity - @ol_qty < 10)
                                  then 91 else 0
end,
                          s_order_cnt =
s_order_cnt + 1,
```

```
                        @s_data = s_data,
                    @s_dist = case @d_id
                                when 1 then
s_dist_01
                                when 2 then
s_dist_02
                                when 3 then
s_dist_03
                                when 4 then
s_dist_04
                                when 5 then
s_dist_05
                                when 6 then
s_dist_06
                                when 7 then
s_dist_07
                                when 8 then
s_dist_08
                                when 9 then
s_dist_09
                                when 10 then
s_dist_10
                                end
                          where s_w_id = @w_id
and
                               s_i_id = @i_id
              if (@@rowcount = 0)
                  begin
                    select @commit_flag = 0
                    select NULL, NULL, NULL,
NULL
                    continue
                 end
               /*Otherwise if the Stock is
found */

/* Compaq NT loader used Jan 01 1800 as NULL
date */
         INSERT INTO order_line (
              ol_o_id, ol_d_id, ol_w_id,
ol_number, ol_i_id,
              ol_supply_w_id, ol_delivery_d,
ol_quantity,
              ol_amount, ol_dist_info)
         VALUES (
              @o_id, @d_id, @w_id, @ol_number,
@i_id,
              @w_id, "18991231", @ol_qty,
              @ol_amount, @s_dist)


          /* send line-item data to client */
          select
             @i_name,
             @s_quantity,
             @i_price,
             b_g= case when((patindex("%ORIGI-
NAL%", @i_data) > 0) and
                                (patindex("%ORIG-
INAL%", @s_data) > 0))
                          then "B" else "G" end

       end /* while */

       SELECT  @c_last = c_last,
               @c_discount = c_discount,
               @c_credit   = c_credit,
               @c_ins_id= c_id
       FROM    customer (index c_clu prefetch 4
lru)  HOLDLOCK
       WHERE  c_w_id = @w_id
         AND  c_d_id = @d_id
         AND  c_id   = @c_id

       INSERT INTO orders (
              o_id, o_c_id, o_d_id, o_w_id,
              o_entry_d, o_carrier_id,
o_ol_cnt, o_all_local)
       VALUES (
              @o_id, @c_ins_id, @d_id, @w_id,
              @o_entry_d, -1, @o_ol_cnt, 1)
       INSERT INTO new_order (no_o_id,
no_d_id, no_w_id)
       VALUES (@o_id, @d_id, @w_id)


       SELECT  @w_tax = w_tax
       FROM warehouse HOLDLOCK
       WHERE w_id = @w_id



       if (@commit_flag = 1)
              commit transaction NO
       else
              rollback transaction NO

       select       /* Return to client */
              @w_tax, @d_tax, @o_id, @c_last,
              @c_discount, @c_credit,
@o_entry_d
end
go

if exists ( SELECT name FROM sysobjects WHERE
name = 'neworder_remote')
       DROP PROC neworder_remote
go
CREATE PROC neworder_remote (
       @w_id        smallint,
```

```
       @d_id        tinyint,
       @c_id        int,
       @o_ol_cnttinyint,

       @i_id  int = 0, @s_w_idsmallint = 0,
@ol_qtytinyint = 0,
       @i_id2 int = 0, @s_w_id2smallint = 0,
@ol_qty2tinyint = 0,
       @i_id3 int = 0, @s_w_id3smallint = 0,
@ol_qty3tinyint = 0,
       @i_id4 int = 0, @s_w_id4smallint = 0,
@ol_qty4tinyint = 0,
       @i_id5 int = 0, @s_w_id5smallint = 0,
@ol_qty5tinyint = 0,
       @i_id6 int = 0, @s_w_id6smallint = 0,
@ol_qty6tinyint = 0,
       @i_id7 int = 0, @s_w_id7smallint = 0,
@ol_qty7tinyint = 0,
       @i_id8 int = 0, @s_w_id8smallint = 0,
@ol_qty8tinyint = 0,
       @i_id9 int = 0, @s_w_id9smallint = 0,
@ol_qty9tinyint = 0,
       @i_id10int = 0, @s_w_id10smallint = 0,
@ol_qty10tinyint = 0,
       @i_id11int = 0, @s_w_id11smallint = 0,
@ol_qty11tinyint = 0,
       @i_id12int = 0, @s_w_id12smallint = 0,
@ol_qty12tinyint = 0,
       @i_id13int = 0, @s_w_id13smallint = 0,
@ol_qty13tinyint = 0,
       @i_id14int = 0, @s_w_id14smallint = 0,
@ol_qty14tinyint = 0,
       @i_id15int = 0, @s_w_id15smallint = 0,
@ol_qty15tinyint = 0
)
as

declare
       @w_tax       real,        @d_tax real,
       @c_last      char(16),@c_creditchar(2),
       @c_discountreal,    @commit_flagtiny-
int,
       @c_ins_id int,

       @i_pricereal,
       @i_name      char(24),@i_data
char(50),

       @s_quantitysmallint,
       @s_ytd       int,         @s_order_cnt
int,
       @s_dist      char(24),@s_data
char(50),
       @s_remote_cntint,   @remote      int,
       @ol_numbertinyint,@o_id    int,
       @o_entry_ddatetime,@b_g    char(1),
       @ol_amount real


begin

       begin transaction NO

       -- @#@# UPDATE district FROM district,
warehouse, customer
       --

       UPDATE district
       SET      d_next_o_id = d_next_o_id + 1
              , @o_id       = d_next_o_id
              , @d_tax = d_tax
              , @commit_flag= 1
              , @ol_number= 0
              , @o_entry_d= getdate()
       WHERE         d_w_id = @w_id
              AND    d_id   = @d_id

       while (@ol_number < @o_ol_cnt) begin
              SELECT @ol_number = @ol_number +
1
              ,@i_id = case @ol_number
         when 1 then @i_id2
                when 2 then @i_id3
                when 3 then @i_id4
                when 4 then @i_id5
                when 5 then @i_id6
                when 6 then @i_id7
                when 7 then @i_id8
                when 8 then @i_id9
                when 9 then @i_id10
                when 10 then @i_id11
                when 11 then @i_id12
                when 12 then @i_id13
                when 13 then @i_id14
                when 14 then @i_id15
                else @i_id
                end
              , @ol_qty = case @ol_number
                when 1 then @ol_qty2
                when 2 then @ol_qty3
                when 3 then @ol_qty4
                when 4 then @ol_qty5
                when 5 then @ol_qty6
                when 6 then @ol_qty7
                when 7 then @ol_qty8
                when 8 then @ol_qty9
                when 9 then @ol_qty10
                when 10 then @ol_qty11
                when 11 then @ol_qty12
```

```
                when 12 then @ol_qty13
                when 13 then @ol_qty14
                when 14 then @ol_qty15
                else @ol_qty
                end
              ,@s_w_id = case @ol_number
          when 1 then @s_w_id2
                when 2 then @s_w_id3
                when 3 then @s_w_id4
                when 4 then @s_w_id5
                when 5 then @s_w_id6
                when 6 then @s_w_id7
                when 7 then @s_w_id8
                when 8 then @s_w_id9
                when 9 then @s_w_id10
                when 10 then @s_w_id11
                when 11 then @s_w_id12
                when 12 then @s_w_id13
                when 13 then @s_w_id14
                when 14 then @s_w_id15
                else @s_w_id
                end

        /* convert c_no_is cursor to a simple
select */
        /* get item data (no one update item)
*/
               select @i_price = i_price,
                      @i_name = i_name ,
                      @i_data = i_data
                  from item HOLDLOCK
                  where i_id = @i_id

              if (@@rowcount = 0)
                 begin
                       select @commit_flag = 0
                      select NULL, NULL, NULL,
NULL
                      continue
                 end
          /* Otherwise if the item is found */
             update stock
               set s_ytd = s_ytd + @ol_qty,
                  @ol_amount = @ol_qty *
@i_price,
                   @s_quantity = s_quantity -
@ol_qty +
                                 case when
(s_quantity - @ol_qty < 10)
                                 then 91 else 0
end,
                   s_quantity = s_quantity -
@ol_qty +
                    case when (s_quantity -
@ol_qty < 10)
                      then 91 else 0 end,
                   @s_data = s_data,
                   @s_dist = case @d_id
                          when 1 then s_dist_01
                          when 2 then s_dist_02
                          when 3 then s_dist_03
                          when 4 then s_dist_04
                          when 5 then s_dist_05
                          when 6 then s_dist_06
                          when 7 then s_dist_07
                          when 8 then s_dist_08
                          when 9 then s_dist_09
                         when 10 then s_dist_10
                          end,
                  s_order_cnt = s_order_cnt + 1,
                  s_remote_cnt = s_remote_cnt +
                  case when (@s_w_id = @w_id)
                           then 0 else 1 end
               where s_w_id = @s_w_id and
                     s_i_id = @i_id

               if (@@rowcount = 0)
                 begin
                   select @commit_flag = 0
                  select NULL, NULL, NULL, NULL
                   continue
                 end

         INSERT INTO order_line (
              ol_o_id, ol_d_id, ol_w_id,
ol_number, ol_i_id,
              ol_supply_w_id, ol_delivery_d,
ol_quantity,
              ol_amount, ol_dist_info)
          VALUES (
               @o_id, @d_id, @w_id, @ol_number,
@i_id,
              @s_w_id, “18991231”, @ol_qty,
              @ol_amount, @s_dist)

               /* send line-item to client */
               select
                 @i_name,
               @s_quantity,
               @i_price,
               b_g = case when ((patin-
dex(“%ORIGINAL%”, @i_data) > 0) and
                                (patindex(“%ORIG-
INAL%”, @s_data) > 0))
                     then “B” else “G” end
       end /* while */


       SELECT  @c_last = c_last,
               @c_discount = c_discount,
```

```
              @c_credit = c_credit,
              @c_ins_id= c_id
       FROM    customer (index c_clu prefetch 4
lru)  HOLDLOCK
       WHERE  c_w_id = @w_id
         AND  c_d_id = @d_id
         AND  c_id   = @c_id

       INSERT INTO orders (
              o_id, o_c_id, o_d_id, o_w_id,
              o_entry_d, o_carrier_id,
o_ol_cnt, o_all_local)
       VALUES (
              @o_id, @c_ins_id, @d_id, @w_id,
              @o_entry_d, -1, @o_ol_cnt, 0)
       INSERT INTO new_order (no_o_id,
no_d_id, no_w_id)
       VALUES (@o_id, @d_id, @w_id)

       SELECT  @w_tax = w_tax
       FROM warehouse HOLDLOCK
       WHERE w_id = @w_id

       if (@commit_flag = 1)
              commit transaction NO
       else
              rollback transaction NO

       select       /* Return to client */
              @w_tax, @d_tax, @o_id, @c_last,
              @c_discount, @c_credit,
@o_entry_d
end
go
if exists (select * from sysobjects where name
= 'payment_byid')
       DROP PROC payment_byid
go
CREATE PROC payment_byid
       @w_id        smallint,@c_w_id    small-
int,
       @h_amount float,
       @d_id        tinyint,@c_d_id     tiny-
int,
       @c_id  int
as
declare@c_last      char(16)

declare@w_street_1char(20),@w_street_2
char(20),
       @w_city      char(20),@w_statechar(2),
       @w_zip       char(9),@w_name
char(10),
       @w_ytd       float,
@w_id_retrieved smallint

declare@d_street_1char(20),@d_street_2
char(20),
       @d_city      char(20),@d_statechar(2),
       @d_zip       char(9),@d_name
char(10),
       @d_ytd       float

declare@c_firstchar(16),@c_middlechar(2),
       @c_street_1char(20),@c_street_2
char(20),
       @c_city      char(20),@c_statechar(2),
       @c_zip       char(9),@c_phonechar(16),
       @c_sincedatetime,@c_creditchar(2),
       @c_credit_limnumeric(12,0),@c_balance
float,
       @c_discountreal,
       @data1       char(250),@data2
char(250),
       @c_data_1char(250),@c_data_2char(250)

declare @screen_datachar(200),@today datetime


BEGIN TRANSACTION PID

       UPDATE district
              SET d_ytd = d_ytd + @h_amount
                 ,@d_ytd = d_ytd
                 ,@d_street_1 = d_street_1
                 ,@d_street_2 = d_street_2
                 ,@d_city = d_city
                 ,@d_state = d_state
                 ,@d_zip = d_zip
                 ,@d_name = d_name
              WHERE d_w_id= @w_id
              AND    d_id   = @d_id

       UPDATE warehouse
              SET w_ytd = w_ytd + @h_amount
                 ,@w_ytd = w_ytd
                 ,@w_id_retrieved = w_id
                 ,@w_street_1 = w_street_1
                 ,@w_street_2 = w_street_2
                 ,@w_city = w_city
                 ,@w_state = w_state
                 ,@w_zip = w_zip
                 ,@w_name = w_name
              WHERE w_id = @w_id


       /* Customer data */
       UPDATE customer SET
                @c_first = c_first
              , @c_middle = c_middle
```

```
              , @c_last = c_last
              , @c_street_1 = c_street_1
              , @c_street_2 = c_street_2
              , @c_city = c_city
              , @c_state = c_state
              , @c_zip = c_zip
              , @c_phone = c_phone
              , @c_credit = c_credit
              , @c_credit_lim = c_credit_lim
              , @c_discount = c_discount
              , c_balance = c_balance -
@h_amount
              , @c_balance = c_balance -
@h_amount
              , c_ytd_payment = c_ytd_payment +
@h_amount
              , c_payment_cnt = c_payment_cnt +
1
              , @c_since = c_since
              , @data1 = c_data1
              , @data2 = c_data2
              , @today = getdate()
       where
              c_id = @c_id
              and c_w_id = @c_w_id
              and c_d_id = @c_d_id

       if (@c_credit = “BC”)
       begin
              SELECT  @c_data_2 =
                     substring(@data1, 209, 42)
+
                     substring(@data2, 1, 208)
                     ,@c_data_1 =
                     convert(char(5), @c_id) +
                     convert(char(4), @c_d_id)
+
                     convert(char(5), @c_w_id)
+
                     convert(char(4), @d_id) +
                     convert(char(5), @w_id) +
                     convert(char(19),
@h_amount) + substring(@data1, 1, 208)

              UPDATE customer SET
                       c_data1 = @c_data_1
                     , c_data2 = @c_data_2
                     , @screen_data = sub-
string(@c_data_1, 1, 200)
              WHERE
                     c_id = @c_id
                     AND c_w_id = @c_w_id
                     AND c_d_id = @c_d_id
       end /* if */

       /* Create the history record */
       INSERT INTO history (
              h_c_id, h_c_d_id, h_c_w_id,
h_d_id, h_w_id,
              h_date, h_amount, h_data)
       VALUES (
              @c_id, @c_d_id, @c_w_id, @d_id,
@w_id_retrieved,
              @today, @h_amount, (@w_name + “
“ + @d_name))

COMMIT TRANSACTION PID

       select       /* Return to client */
              @c_id,
              @c_last,
              @today,
              @w_street_1,
              @w_street_2,
              @w_city,
              @w_state,
              @w_zip,

              @d_street_1,
              @d_street_2,
              @d_city,
              @d_state,
              @d_zip,

              @c_first,
              @c_middle,
              @c_street_1,
              @c_street_2,
              @c_city,
              @c_state,
              @c_zip,
              @c_phone,
              @c_since,
              @c_credit,
              @c_credit_lim,
              @c_discount,
              @c_balance,
              @screen_data
go
if exists (select * from sysobjects where name
= ‘payment_byname’)
       DROP PROC payment_byname
go
CREATE PROC payment_byname
       @w_id        smallint,@c_w_id    small-
int,
       @h_amount float,
       @d_id        tinyint,@c_d_id     tiny-
int,
       @c_lastchar(16)
```

```
as
declare@n    int,   @c_id  int

declare@w_street_1char(20),@w_street_2
char(20),
       @w_city      char(20),@w_statechar(2),
       @w_zip       char(9),@w_name
char(10),
       @w_ytd       float,
@w_id_retrieved smallint

declare@d_street_1char(20),@d_street_2
char(20),
       @d_city      char(20),@d_statechar(2),
       @d_zip       char(9),@d_name
char(10),
       @d_ytd       float

declare@c_firstchar(16),@c_middlechar(2),
       @c_street_1char(20),@c_street_2
char(20),
       @c_city      char(20),@c_statechar(2),
       @c_zip       char(9),@c_phonechar(16),
       @c_sincedatetime,@c_creditchar(2),
       @c_credit_limnumeric(12,0),@c_balance
float,
       @c_discountreal,
       @data1       char(250),@data2
char(250),
       @c_data_1char(250),@c_data_2char(250)

declare @screen_datachar(200),@today datetime



BEGIN TRANSACTION PNM
       SELECT @n = (count(*)+1)/2
       FROM customer (index c_non1 prefetch 4
lru) HOLDLOCK
       WHERE  c_w_id = @c_w_id and
              c_d_id = @c_d_id and
              c_last = @c_last

       set rowcount @n

       -- @#@ SELECT FROM customer HOLDLOCK
       SELECT @c_id = c_id
       FROM customer (index c_non1 prefetch 4
lru) HOLDLOCK
       WHERE  c_w_id = @c_w_id and
              c_d_id = @c_d_id and
              c_last = @c_last

       -- Reset, so as to do full retrievals
hereafter.
       set rowcount 0

       UPDATE district
              SET d_ytd = d_ytd + @h_amount
                 ,@d_ytd = d_ytd
                 ,@d_street_1 = d_street_1
                 ,@d_street_2 = d_street_2
                 ,@d_city = d_city
                 ,@d_state = d_state
                 ,@d_zip = d_zip
                 ,@d_name = d_name
              WHERE d_w_id= @w_id
              AND    d_id   = @d_id

       UPDATE warehouse
              SET w_ytd = w_ytd + @h_amount
                 ,@w_ytd = w_ytd
                 ,@w_id_retrieved = w_id
                 ,@w_street_1 = w_street_1
                 ,@w_street_2 = w_street_2
                 ,@w_city = w_city
                 ,@w_state = w_state
                 ,@w_zip = w_zip
                 ,@w_name = w_name
              WHERE w_id = @w_id

       /* Customer data */
       UPDATE customer SET
                @c_first = c_first
              , @c_middle = c_middle
              , @c_last = c_last
              , @c_street_1 = c_street_1
              , @c_street_2 = c_street_2
              , @c_city = c_city
              , @c_state = c_state
              , @c_zip = c_zip
              , @c_phone = c_phone
              , @c_credit = c_credit
              , @c_credit_lim = c_credit_lim
              , @c_discount = c_discount
              , c_balance = c_balance -
@h_amount
              , @c_balance = c_balance -
@h_amount
              , c_ytd_payment = c_ytd_payment +
@h_amount
              , c_payment_cnt = c_payment_cnt +
1
              , @c_since = c_since
              , @data1 = c_data1
              , @data2 = c_data2
              , @today = getdate()
       where
              c_id = @c_id
              and c_w_id = @c_w_id
```

```
              and c_d_id = @c_d_id

       SELECT @screen_data = NULL
       if (@c_credit = “BC”)
       begin
              SELECT  @c_data_2 =
                     substring(@data1, 209, 42)
+
                     substring(@data2, 1, 208)
                     ,@c_data_1 =
                     convert(char(5), @c_id) +
                     convert(char(4), @c_d_id)
+
                     convert(char(5), @c_w_id)
+
                     convert(char(4), @d_id) +
                     convert(char(5), @w_id) +
                     convert(char(19),
@h_amount) + substring(@data1, 1, 208)

              UPDATE customer SET
                       c_data1 = @c_data_1
                     , c_data2 = @c_data_2
                     , @screen_data = sub-
string(@c_data_1, 1, 200)
              WHERE
                     c_id = @c_id
                     AND c_w_id = @c_w_id
                     AND c_d_id = @c_d_id
       end /* if */

       /* Create the history record */
       INSERT INTO history (
              h_c_id, h_c_d_id, h_c_w_id,
h_d_id, h_w_id,
              h_date, h_amount, h_data)
       VALUES (
              @c_id, @c_d_id, @c_w_id, @d_id,
@w_id_retrieved,
              @today, @h_amount, (@w_name + “
“ + @d_name))

COMMIT TRANSACTION PNM

       select       /* Return to client */
              @c_id,
              @c_last,
              @today,
              @w_street_1,
              @w_street_2,
              @w_city,
              @w_state,
              @w_zip,

              @d_street_1,
              @d_street_2,
              @d_city,
              @d_state,
              @d_zip,

              @c_first,
              @c_middle,
              @c_street_1,
              @c_street_2,
              @c_city,
              @c_state,
              @c_zip,
              @c_phone,
              @c_since,
              @c_credit,
              @c_credit_lim,
              @c_discount,
              @c_balance,
              @screen_data
go
if exists (select * from sysobjects where name
= ‘order_status_byid’)
       DROP PROC order_status_byid
go
CREATE PROC order_status_byid
       @w_id        smallint,
       @d_id        tinyint,
       @c_id        int
as

DECLARE@o_id        int,
       @o_entry_ddatetime,
       @o_carrier_idsmallint

BEGIN TRANSACTION OSID

/* Get the latest order made by the customer */
       set rowcount 1
       SELECT @o_id = o_id, @o_carrier_id =
o_carrier_id,
              @o_entry_d = o_entry_d
       FROM   orders (index o_clu prefetch 16
mru) HOLDLOCK
       WHERE  o_w_id = @w_id
       AND    o_d_id = @d_id
       AND    o_c_id = @c_id
       ORDER BY o_w_id DESC, o_d_id DESC, o_id
DESC
       set rowcount 0

/* Select order lines for the current order */
       select /* Return multiple rows to client
*/
              ol_supply_w_id,
              ol_i_id,
```

```
              ol_quantity,
              ol_amount,
              ol_delivery_d
       FROM   order_line HOLDLOCK
       WHERE  ol_o_id = @o_id
       AND    ol_d_id = @d_id
       AND    ol_w_id = @w_id

       select /* Return single row to client */
              @c_id, c_last, c_first, c_middle,
c_balance,
              @o_id,
              @o_entry_d,
              @o_carrier_id
       FROM   customer (index c_clu prefetch 4
lru) HOLDLOCK
       WHERE   c_id    = @c_id
       AND    c_d_id  = @d_id
       AND    c_w_id  = @w_id

COMMIT TRANSACTION OSID
go
if exists (select * from sysobjects where name
= 'order_status_byname')
       DROP PROC order_status_byname
go
CREATE PROC order_status_byname
       @w_id         smallint,
       @d_id         tinyint,
       @c_lastchar(16)
as

DECLARE@o_id         int,
       @o_entry_ddatetime,
       @o_carrier_idsmallint

declare@n    int,   @c_id  int

BEGIN TRANSACTION OSNM
       SELECT @n = (count(*)+1)/2
       FROM customer (index c_non1 prefetch 4
lru) HOLDLOCK
       WHERE  c_w_id = @w_id and
              c_d_id = @d_id and
              c_last = @c_last


       -- Retrieve upto mid-point number of
rows.
       set rowcount @n

       -- @#@# SELECT FROM customer HOLDLOCK
       SELECT @c_id = c_id
       FROM customer (index c_non1 prefetch 4
lru) HOLDLOCK
       WHERE  c_w_id = @w_id and
              c_d_id = @d_id and
              c_last = @c_last


/* Get the latest order made by the customer */
       set rowcount 1
       SELECT @o_id = o_id, @o_carrier_id =
o_carrier_id,
              @o_entry_d = o_entry_d
       FROM   orders (index o_clu prefetch 16
mru) HOLDLOCK
       WHERE  o_w_id = @w_id
       AND    o_d_id = @d_id
       AND    o_c_id = @c_id
       ORDER BY o_w_id DESC, o_d_id DESC, o_id
DESC
       set rowcount 0

/* Select order lines for the current order */
       select /* Return multiple rows to client
*/
              ol_supply_w_id,
              ol_i_id,
              ol_quantity,
              ol_amount,
              ol_delivery_d
       FROM   order_line HOLDLOCK
       WHERE  ol_o_id = @o_id
       AND    ol_d_id = @d_id
       AND    ol_w_id = @w_id

       select /* Return single row to client */
              @c_id, c_last, c_first, c_middle,
c_balance,
              @o_id,
              @o_entry_d,
              @o_carrier_id
       FROM   customer (index c_clu prefetch 4
lru) HOLDLOCK
       WHERE   c_id    = @c_id
       AND    c_d_id  = @d_id
       AND    c_w_id  = @w_id

COMMIT TRANSACTION OSNM
go

if exists (select * from sysobjects where name
= 'delivery')
        drop proc delivery
go

CREATE PROC delivery
        @w_id           smallint,
        @o_carrier_id   smallint,
```

```
        @d_id           tinyint = 1
as

declare @no_o_id     int,        @o_c_id
smallint,
        @ol_total       float,
@ol_amount      float,
        @junk_id        smallint,
        @today          datetime

declare c_del_no CURSOR FOR
        SELECT  no_o_id
        FROM    new_order (index no_clu) HOLD-
LOCK
        WHERE   no_d_id = @d_id
        AND     no_w_id = @w_id
        FOR UPDATE

        /*
        ** The only purpose of the index hint
in the above is to ensure
        ** that the clustered index is used.
As it turns out, our optimizer
       ** chooses the clustered index anyway -
- with or without the hint.
        */
begin

    while (@d_id <= 10) begin
    BEGIN TRANSACTION DEL

        OPEN c_del_no

        FETCH c_del_no INTO @no_o_id

        if (@@sqlstatus != 0)
        begin
                COMMIT TRANSACTION DEL
                select NULL
                CLOSE c_del_no
        end
        else
        begin
                DELETE  FROM new_order
                WHERE   CURRENT OF c_del_no
                CLOSE   c_del_no

              SELECT  @ol_total = 0.0, @today
= getdate()

             -- @#@# UPDATE order_line
                UPDATE order_line
              SET       ol_delivery_d = @today
                      , @ol_total = @ol_total
+ ol_amount
                WHERE   ol_o_id = @no_o_id
                AND     ol_d_id = @d_id
             AND    ol_w_id = @w_id

             -- @#@# UPDATE orders
                UPDATE  orders
                SET       o_carrier_id =
@o_carrier_id
                        , @o_c_id = o_c_id
                WHERE   o_id    = @no_o_id
                AND     o_d_id  = @d_id
        AND   o_w_id = @w_id

                UPDATE  customer
                SET     c_balance       =
c_balance + @ol_total,
                        c_delivery_cnt  =
c_delivery_cnt + 1
                WHERE   c_id    = @o_c_id
                AND     c_d_id  = @d_id
                AND     c_w_id  = @w_id

                COMMIT TRANSACTION DEL

               select      /* Return to client
*/
                        @no_o_id
         end
         select @d_id = @d_id + 1
    end
end
go


if exists ( SELECT name FROM sysobjects WHERE
name = 'stock_level')
    DROP PROC stock_level
go

CREATE PROC stock_level
       @w_id  smallint,
       @d_id  tinyint,
       @threshold smallint
as
       select  s_i_id /* Return to client */
       FROM   district,
              order_line (index ol_clu prefetch
4 lru),
              stock (index s_clu prefetch 4
lru)
       WHERE  d_w_id =      @w_id
       AND    d_id   =      @d_id
       AND    ol_w_id=      @w_id
       AND    ol_d_id=      @d_id
       AND    ol_o_idbetween (d_next_o_id - 20)
```

```
and (d_next_o_id - 1)
       AND    s_w_id =      ol_w_id
       AND    s_i_id =      ol_i_id
       AND     s_quantity < @threshold
go
EOF
```

tpcc_tables_parallel.sh

```
#!/bin/sh -fx

partition=${2:-1}

isql -Usa -P$PASSWORD << EOF
/* This script will create all the tables
required for TPC-C benchmark */
/* It will also create some of the indexes. */
sp_dboption tpcc,"select into/bulkcopy",true
go
use tpcc
go
checkpoint
go

if exists ( select name from sysobjects where
name = 'warehouse' )
       drop table warehouse
go
create table warehouse (
       w_id         smallint,
       w_name       char(10),
       w_street_1char(20),
       w_street_2char(20),
       w_city       char(20),
       w_state      char(2),
       w_zip        char(9),
       w_tax        real,
       w_ytd        float        /*- Updated
by PID, PNM */
) with max_rows_per_page = 1 on Swarehouse
go

if exists ( select name from sysobjects where
name = 'district' )
       drop table district
go
create table district (
       d_id         tinyint,
       d_w_id       smallint,
       d_name       char(10),
       d_street_1char(20),
       d_street_2char(20),
       d_city       char(20),
       d_state      char(2),
       d_zip        char(9),
       d_tax        real,
       d_ytd        float,       /*- Updated
by PID, PNM */
       d_next_o_idint      /*- Updated by NO
*/
) with max_rows_per_page = 10 on Sdistrict
go

if exists ( select name from sysobjects where
name = 'customer' )
       drop table customer
go
create table customer (
       c_id         int,
       c_d_id       tinyint,
       c_w_id       smallint,
       c_first      char(16),
       c_middlechar(2),
       c_last       char(16),
       c_street_1char(20),
       c_street_2char(20),
       c_city       char(20),
       c_state      char(2),
       c_zip        char(9),
       c_phone      char(16),
       c_since      datetime,
       c_creditchar(2),
       c_credit_limnumeric(12,2),
       c_discountreal,
       c_delivery_cntsmallint,
       c_payment_cntsmallint,/*- Updated by
PNM, PID */
       c_balancefloat,     /*- Updated by PNM,
PID */
       c_ytd_paymentfloat, /*- Updated by PNM,
PID */
       c_data1      char(250),/*- Updated (?)
by PNM, PID */
       c_data2      char(250)/*- Updated (?)
by PNM, PID */
) on Scustomer
go
alter table customer partition $partition
go

if exists ( select name from sysobjects where
name = 'history' )
       drop table history
go
create table history (
       h_c_id       int,
       h_c_d_idtinyint,
       h_c_w_idsmallint,
```

```
	h_d_id		tinyint,
	h_w_id		smallint,
	h_date		datetime,
	h_amountfloat,
	h_data		char(24)
) on Shistory
go
alter table history partition 512
go

if exists ( select name from sysobjects where
name = 'new_order' )
	drop table new_order
go
create table new_order (
	no_o_id		int,
	no_d_id		tinyint,
	no_w_id		smallint,
) on Snew_order
go

alter table new_order partition $partition
go


if exists ( select name from sysobjects where
name = 'orders' )
	drop table orders
go
create table orders (
	o_id		int,
	o_c_id		int,
	o_d_id		tinyint,
	o_w_id		smallint,
	o_entry_ddatetime,
	o_carrier_idsmallint,/*- Updated by D
*/
	o_ol_cnttinyint,
	o_all_localtinyint
) on Sorders
go
alter table orders partition $partition
go

if exists ( select name from sysobjects where
name = 'order_line' )
	drop table order_line
go
create table order_line (
	ol_o_id		int,
	ol_d_id		tinyint,
	ol_w_id		smallint,
	ol_numbertinyint,
	ol_i_id		int,
	ol_supply_w_idsmallint,
	ol_delivery_ddatetime,/*- Updated by D
*/
	ol_quantitysmallint,
	ol_amountfloat,
	ol_dist_infochar(24)
) on Sorder_line
go

alter table order_line partition $partition
go

if exists ( select name from sysobjects where
name = 'item' )
	drop table item
go
create table item (
	i_id		int,
	i_im_id		int,
	i_name		char(24),
	i_price		float,
	i_data		char(50)
) on Sitem
go

alter table item partition $partition
go

if exists ( select name from sysobjects where
name = 'stock' )
	drop table stock
go
create table stock (
	s_i_id		int,
	s_w_id		smallint,
	s_quantitysmallint,/*- Updated by NO */
	s_ytd		int,		/*- Updated
by NO */
	s_order_cntsmallint,/*- Updated by NO
*/
	s_remote_cntsmallint,/*- Updated by NO
*/
	s_dist_01char(24),
	s_dist_02char(24),
	s_dist_03char(24),
	s_dist_04char(24),
	s_dist_05char(24),
	s_dist_06char(24),
	s_dist_07char(24),
	s_dist_08char(24),
	s_dist_09char(24),
	s_dist_10char(24),
	s_data		char(50)
) on Sstock
go
alter table stock partition $partition
```

```
go

checkpoint
go
EOF
```

Code to populate

build.jam

```
SSPart svr sql generic tools tpcc loader ;

SSModifyDirCcFlags + SYBLP64_FLAG TPCC_DEFINES
;

if $(SSPLAT) in linux
{
    SSSysLibs tpcc_loader : blk$(L64) ct$(L64)
cs$(L64) sybtcl$(L64) comn$(L64) intl$(L64)
sybdb$(L64) ;
}
else
{
    SSSysLibs tpcc_loader : blk$(L64) ct$(L64)
cs$(L64) tcl$(L64) comn$(L64) intl$(L64)
sybdb$(L64) ;
}

SSMain tpcc_loader  : tpcc_ld_bulk_sybase.c
tpcc_ld_error.c tpcc_ld_load.c ;
```

tpcc_ld_bulk_sybase.c

```
/**********************************************
********************
**********************************************
**********************

Sybase Specific Routines

**********************************************
**********************
**********************************************
**********************/

#include <stdio.h>
#include <stdlib.h>
#include <ctpublic.h>
#include <bkpublic.h>
#include <unistd.h>
#ifdef _NTINTEL
#include <sys/timeb.h>
#include <io.h>
#else
#include <sys/time.h>
#endif
#include <string.h>
#include “tpcc_ld_loader.h”
void
datetime(date)
DBDATETIME *date;
{
#ifdef _NTINTEL
       time_t time1;
       time(&time1);
       date->dtdays = time1 / (60*60*24)
                     + (1970-1900)*365 + (1970-
1900)/4;
       date->dttime = (time1 % (60*60*24))*300
;
#else
       struct timeval time;
       gettimeofday(&time, NULL);
       date->dtdays = time.tv_sec / (60*60*24)
                     + (1970-1900)*365 + (1970-
1900)/4;
       date->dttime = (time.tv_sec %
(60*60*24))*300
                     +
time.tv_usec*300/1000000;
#endif
return;
}


/* define the type information for each field
*/
typedef struct
{
       char *terminator;
       int termlen;
       int type;
       int cstype;
} bind_parm;

bind_parm parm[MAX_T] =
{
       /* COUNT */{NULL, 0, CS_INT_TYPE,
SYBINT4},
       /* ID  */     {NULL, 0, CS_INT_TYPE,
SYBINT4},
       /* MONEY */{NULL, 0, CS_FLOAT_TYPE,
SYBFLT8},
       /* FLOAT */{NULL, 0, CS_FLOAT_TYPE,
SYBFLT8},
        /* TEXT */      {“”,   1, CS_CHAR_TYPE,
SYBCHAR},
       /* DATE */{NULL, 0, CS_DATETIME_TYPE,
SYBDATETIME},
       /* LOGICAL */{NULL, 0, CS_INT_TYPE,
SYBINT4}
};

#define MAXOPENS 10
```

```
#define RETURN_IF(a,b)                  \
       if (a != CS_SUCCEED)             \
       {                                \
              fprintf(stderr, “Error in: %s\n”,
b);    \
              return a;                 \
       }

#define EXIT_IF(a) if (a != CS_SUCCEED)\
       { fprintf(stderr, “FATAL ERROR! Line
%d\n”, __LINE__); exit(-1);}

CS_CONNECTION *dbconn[MAXOPENS];
int    count[MAXOPENS];

CS_RETCODE    init_db();
CS_RETCODE    connect_db();
CS_RETCODE    send_sql();
void          handle_returns();

extern  CS_INT   server_msg_handler();
extern  CS_INT   cl_err_handler();
extern  CS_INT   cs_err_handler();


CS_CONTEXT    *cntx_ptr;
int bulk_open(database, table, password,
blk_desc, sliceNum)
       CS_CHAR    *database;
       CS_CHAR    *table;
       CS_CHAR    *password;
       CS_BLKDESC **blk_desc;
       CS_INT    *sliceNum;
{
       int db;

       CS_RETCODE     retcode=0;
       CS_CONNECTION *conn_ptr;
       CS_COMMAND    *cmd_ptr;

       char    tblname[25];

       /* make note we have established a con-
nection */
       for (db=0; db<MAXOPENS; db++)
              if (dbconn[db] == NULL) break;
       count[db] = 0;

       retcode = CS_SUCCEED;
       retcode = init_db(&cntx_ptr);
       EXIT_IF(retcode);

       strcpy(tblname, table);
       retcode = connect_db(cntx_ptr,
&conn_ptr, “sa”, “”, table);
       EXIT_IF(retcode);

       retcode = ct_cmd_alloc(conn_ptr,
&cmd_ptr);
       EXIT_IF(retcode);

       retcode = send_sql(cmd_ptr, “use
tpcc”);
       EXIT_IF(retcode);

       handle_returns(cmd_ptr);
       /* prepare to do a bulk copy */
       retcode = blk_alloc(conn_ptr,
CS_CURRENT_VERSION, blk_desc);
       EXIT_IF(retcode);

       if ( *sliceNum != INVALID_SLICE_NUM )
       {
              retcode =  retcode =
blk_props(*blk_desc, CS_SET, BLK_SLICENUM,
                                  sliceNum,
CS_UNUSED, NULL) ;
                       EXIT_IF(retcode);
       }

       retcode = blk_init(*blk_desc,
CS_BLK_IN,
                     tblname, strlen(tblname));
       EXIT_IF(retcode);

       return db;
}

CS_RETCODE
init_db(cntx_ptr)
CS_CONTEXT **cntx_ptr;
{
       CS_RETCODE    retcode=CS_SUCCEED;

       retcode =
cs_ctx_alloc(CS_CURRENT_VERSION, cntx_ptr);
       RETURN_IF(retcode, “init_db:
cs_ctx_alloc”);

       retcode = ct_init(*cntx_ptr,
CS_CURRENT_VERSION);
       RETURN_IF(retcode, “init_db: ct_init”);

       retcode = ct_callback(*cntx_ptr, NULL,
CS_SET,
                     CS_SERVERMSG_CB, (CS_VOID
*)server_msg_handler);
       RETURN_IF(retcode, “init_db:
ct_callback-server_msg”);
```

```
	retcode = cs_config(*cntx_ptr, CS_SET,
CS_MESSAGE_CB,
			(CS_VOID *)cs_err_handler,
CS_UNUSED, NULL);
	RETURN_IF(retcode, “init_db: cs_config-
cs_err”);

	retcode = ct_callback(*cntx_ptr, NULL,
CS_SET, CS_CLIENTMSG_CB,
			(CS_VOID
*)cl_err_handler);
	RETURN_IF(retcode, “init_db:
ct_callback- cl_err”);
	return retcode;
} /* end of init_db() */

CS_RETCODE
connect_db(cntx_ptr, conn_ptr, user_name,
password, table)
CS_CONTEXT	*cntx_ptr;
CS_CONNECTION	**conn_ptr;
CS_CHAR		*user_name;
CS_CHAR		*password;
CS_CHAR		*table;
{
	CS_RETCODE	retcode=0;
	CS_BOOL		bool;
	CS_INT		packetsize=4096;

	retcode = ct_con_alloc(cntx_ptr,
conn_ptr);
	RETURN_IF(retcode, “connect_db:
ct_con_alloc”);

	retcode = ct_con_props(*conn_ptr,
CS_SET, CS_USERNAME,
			user_name, CS_NULLTERM,
NULL);
	RETURN_IF(retcode, “connect_db:
ct_con_props”);

	retcode = ct_con_props(*conn_ptr,
CS_SET, CS_PASSWORD,
			password, CS_NULLTERM,
NULL);
	RETURN_IF(retcode, “connect_db:
ct_con_props”);

	retcode = ct_con_props(*conn_ptr,
CS_SET, CS_APPNAME,
			table, CS_NULLTERM, NULL);
	RETURN_IF(retcode, “connect_db:
ct_con_props”);

	retcode = ct_con_props(*conn_ptr,
CS_SET, CS_PACKETSIZE,
			&packetsize, CS_UNUSED,
NULL);
	RETURN_IF(retcode, “connect_db:
ct_con_props”);

	bool = CS_TRUE;
	retcode = ct_con_props(*conn_ptr,
CS_SET, CS_BULK_LOGIN,
			&bool, CS_UNUSED, NULL);
	RETURN_IF(retcode, “connect_db:
ct_con_props”);

	/*establish a connection with the
server specified by DSQUERY */
	retcode = ct_connect(*conn_ptr, NULL,
0);
	RETURN_IF(retcode, “connect_db:
ct_connect”);
	return retcode;
}  /* connect_db() */

CS_RETCODE
send_sql(cmd_ptr, sqltext)
CS_COMMAND	*cmd_ptr;
CS_CHAR		*sqltext;
{
	CS_RETCODE	retcode=0;

	retcode = ct_command(cmd_ptr,
CS_LANG_CMD, sqltext,
			CS_NULLTERM, CS_UNUSED);
	RETURN_IF(retcode,
“send_sql:ct_command”);

	retcode = ct_send(cmd_ptr);
	RETURN_IF(retcode, “send_sql:ct_send”);
	return retcode;
}  /* end of send_sql()  */

void
handle_returns(cmd_ptr)
CS_COMMAND	*cmd_ptr;
{
	CS_INT		result_type;

	while (ct_results(cmd_ptr,
&result_type) == CS_SUCCEED)
	{
		;
	}
	return;
}  /* end of handle_returns()  */
```

```
CS_DATAFMT    datafmt[255];
CS_INT        datalen[255];

CS_RETCODE
bulk_bind(int db, int column, char *name, void
*address, int type, CS_BLKDESC *blk_desc)
{
       CS_RETCODE    retcode=0;


       datafmt[column].locale = 0;
       datafmt[column].count  = 1;
       datafmt[column].datatype =
parm[type].type;

       if (parm[type].cstype == SYBCHAR)
       {
              datafmt[column].maxlength = 255;
                datalen[column] =
strlen(address);
       }
       else
       {
              datafmt[column].maxlength =
sizeof(CS_INT);
              datalen[column] = CS_UNUSED;
       }

       retcode = blk_bind(blk_desc, column,
&datafmt[column],
                     address, &datalen[column],
NULL);
       RETURN_IF(retcode, “bulk_bind:
blk_bind”);
       return retcode;
}

/*
void
bulk_null(db, column)
       int db;
       int column;
{
       if (bcp_collen(dbproc[db], 0, column)
!= SUCCEED)
              printf(“Can’t null column %d\n”,
column);
}




void
bulk_non_null(db, column)
       int db;
       int column;
{
       if (bcp_collen(dbproc[db], -1, column)
!= SUCCEED)
              printf(“Can’t non-null column
%d\n”, column);
}
*/

/*void */
CS_RETCODE
bulk_load(db, blk_desc)
       int db;
       CS_BLKDESC*blk_desc;
{
       CS_RETCODE retcode=CS_SUCCEED;
       CS_INT     numofrows=0;

       count[db]++;
       retcode = blk_rowxfer(blk_desc);
       RETURN_IF(retcode, “bulk_load:
blk_rowxfer”);

       if (count[db]%batch_size == 0  )
       {
              retcode = blk_done(blk_desc,
CS_BLK_BATCH, &numofrows);
              if (retcode == CS_FAIL)
              {
                     printf(“bulk_load: Can’t
post rows\n”);
                     RETURN_IF(retcode,
“bulk_load:blk_done”);
              }
       }

#ifdef _NTINTEL
       if (count[db]%1000 == 0)
_write(1,”.”,1);
       if (count[db]%50000 == 0)
_write(1,”\n”,1);
#else
       if (count[db]%1000 == 0)
write(1,”.”,1);
       if (count[db]%50000 == 0)
write(1,”\n”,1);
#endif

       RETURN_IF(retcode,
“bulk_load:blk_done”);
       return retcode;
}
```

```
void
bulk_close(db, blk_desc)
int db;
CS_BLKDESC *blk_desc;
{
       CS_INT numofrows=0;

       if (blk_done(blk_desc, CS_BLK_ALL,
&numofrows) == CS_FAIL)
              printf("Problems completing the
bulk copy.\n");

       if (blk_drop(blk_desc) == CS_FAIL)
       {
              printf("blk_drop failed\n");
       }

       dbconn[db] = NULL;
       if (count[db] >= 1000) write(1,"\n",1);
       return;
}
```

tpcc_ld_error.c

```
#include "stdio.h"
#include "ctpublic.h"
#ifdef _NTINTEL
#include <stdlib.h>
#include <windows.h>
#endif

CS_INT  cl_err_handler(context, connection,
errmsg)
CS_CONTEXT      *context;
CS_CONNECTION   *connection;
CS_CLIENTMSG    *errmsg;

{
       fprintf(stderr, "Open Client Error (%d,
%d, %d, %d)\n",
                     CS_LAYER(errmsg->msgnum-
ber),
                     CS_ORIGIN(errmsg->msgnum-
ber),
                     CS_SEVERITY(errmsg-
>msgnumber),
                     CS_NUMBER(errmsg->msgnum-
ber));

       fprintf(stderr, "%s\n", errmsg->msg-
string);

       if (errmsg->osstringlen > 0)
       {
              fprintf(stderr, "Operating Sys-
tem Error:\n%s\n",
                     errmsg->osstring);
       }

       return (CS_SUCCEED);
}   /* end of cl_err_handler () */

CS_INT server_msg_handler(context, connection,
srvmsg)
CS_CONTEXT     *context;
CS_CONNECTION  *connection;
CS_SERVERMSG   *srvmsg;

{
       if (srvmsg->msgnumber != 5701)
       {
              fprintf(stderr, "Server message
%ld, Severity %ld, State %ld\n",
                     srvmsg->msgnumber, srvmsg-
>severity, srvmsg->state);

              if (srvmsg->svrnlen > 0)
              {
                     fprintf(stderr, "Server
'%s' ", srvmsg->svrname);
              }

              if (srvmsg->proclen > 0)
              {
                     fprintf(stderr, "Procedure
'%s' ", srvmsg->proc);
              }

              if (srvmsg->line > 0)
              {
                     fprintf(stderr, "Line '%d'
", srvmsg->line);
              }

              fprintf(stderr, "\n %s\n",
srvmsg->text);
       }

       return(CS_SUCCEED);
}  /* end of server_msg_handler()  */

CS_INT cs_err_handler(context, errmsg)
CS_CONTEXT      *context;
CS_CLIENTMSG    *errmsg;
{
       fprintf(stderr, "CS Lib Error
%d:\n%s\n",
                     errmsg->msgnumber, errmsg-
>msgstring);
```

```
        return (CS_SUCCEED);
}  /* end of cs_err_handler() */
```

tpcc_ld_load.c

```
typedefunsigned longBitVector;
#define WSZ   (sizeof(BitVector)*8)
/* For axposf use WAREBATCH of 144 */
#ifndef WAREBATCH
#ifdef _NTINTEL
#defineWAREBATCH288
#else
#defineWAREBATCH10000
#endif
#endif
#definenthbit(map,n)map[(n)/WSZ] &  (((BitVec-
tor)0x1)<< ((n)%WSZ))
#definesetbit(map,n)map[(n)/WSZ] |= (((BitVec-
tor)0x1)<< ((n)%WSZ))

/***Load TPCC tables*******/
#include “stdio.h”
#include <stdlib.h>
#include <ctpublic.h>
#include <bkpublic.h>
#include <unistd.h>
#include “string.h”
#include “tpcc_ld_loader.h”

int load_item;
int load_warehouse;
int load_district;
int load_history;
int load_orders;
int load_new_order;
int load_order_line;
int load_customer;
int load_stock;
ID  w1, w2;
ID  i1, i2;
ID  warehouse;
int batch_size = 1000;
char   password[10];
int    sliceNum = INVALID_SLICE_NUM;
int    numOfSlices = INVALID_SLICE_NUM;
int    perm[MAXITEMS+1];
CS_BLKDESC    *blkdesc, *blkdesc1;

#ifdef _NTINTEL
       BitVector * bmp;
#else
       BitVector original[WAREBATCH][((MAX-
ITEMS+(WSZ-1))/WSZ)], * bmp;
#endif


#define print_prologue(name) \
       if(1)\
       {\
              printf(“Loading %s table from
warehouse %d to %d into partition %d\n”,
(name),  w1, w2, sliceNum);\
       }\
       else
#define print_prologue_item(name) \
       if(1)\
       {\
              printf(“Loading %s table from
warehouse %d to %d and from item %d to %d into
partition %d\n”, (name),  w1, w2, i1, i2, sli-
ceNum);\
       }\
       else

int main(argn, argv)
       int argn;
       char **argv;
{
       ID starting_wid, ending_wid, winc,
last_wid;
       ID starting_iid, ending_iid, iinc;
       int current_slicenum;
       pid_t pid = 0, parallel_parent = 0;



       getargs(argn, argv);
       /* This is needed globally,  because
only 10% of the total item
       ** must be marked Original and we need
to create a global
       ** permutation so that all the threads
look into the same
       ** permutation array.
       */
       if (load_item)
       {
              RandomPermutation(perm, MAX-
ITEMS);
       }

       /*
       ** Except for item and stock all others
are parallely
       ** loaded based on the warehouse id.
Item and Stock
       ** are loaded parallely based on the
item.
```

```
       **
       ** We make sure that at least 2 wae-
houses are given
       ** to each thread. For Item and Stock we
don't need
       ** to do this check, because we will be
sharing the
       ** load of MAXITEMS equally among all
threads.
       */
       if ( numOfSlices != INVALID_SLICE_NUM
&&
            ((load_item || load_stock) || (w2
>= (numOfSlices*2))))
       {
              winc = w2 / numOfSlices;
              iinc = MAXITEMS / numOfSlices;

              /*
              ** One restriction that we are
posing is that, number of
              ** warehouses  should be divisi-
ble by
              ** number of slices. This is
because some bug reported
              ** by Sun while doing a parallel
loading with 5000 warehouses
              ** and slicing across 6 parti-
tions where each slice gets
              ** 833.3. To get around this
problem we won't allow fractions.
              */

              if( !( load_item || load_stock)
&& ((w2 % numOfSlices) != 0) )
              {
                     printf(" Please use the
number of slices which devides %d evenly.\n",
w2 );
                     exit(1);
              }

              last_wid = w2;
              for( starting_wid = w1,
ending_wid = winc, current_slicenum =1,
                   starting_iid = 1, ending_iid
= iinc;
                   current_slicenum <= numOfS-
lices;
                   starting_wid += winc,
                   ending_wid += winc,
                   starting_iid += iinc,
                   ending_iid += iinc,
                   current_slicenum++)
              {
                     if ( current_slicenum ==
numOfSlices )
                     {
                            /* Last thread.
Give the remaining */
                            ending_wid =
last_wid;
                            ending_iid = MAX-
ITEMS;
                     }
                     sliceNum =
current_slicenum;
                     if ( load_item ||
load_stock)
                     {
                            i1 = starting_iid;
                            i2 = ending_iid;
                            printf("Loading
from item %d to %d on Slice num %d \n", i1, i2,
sliceNum);
                     }
                     else
                     {
                            w1 = starting_wid;
                            w2 = ending_wid;
                            printf("Loading
from warehouse %d to %d on Slice num %d \n",
w1, w2, sliceNum);
                     }
                     if ( (pid = fork()) < 0 )
                     {
                            printf("Fork failed
\n");
                            exit(1);
                     }else if ( pid > 0 )
                     {
                            parallel_parent =
1;
                            /* Parent */
                            continue;
                     }else
                     {
                            parallel_parent =
0;
                            /* child */
                            break;
                     }
              }
       }

       if ( parallel_parent )
       {
                     while( wait(0) != -1 );
                     printf("Done Loading \n");
                     exit(0);
```

```
	}
	else
	{
		Randomize();
		if (load_item)LoadItems();
		if (load_warehouse)LoadWare-
house(w1, w2);
		if (load_district)LoadDis-
trict(w1, w2);
		if (load_history)LoadHist(w1,
w2);
		if (load_customer)LoadCus-
tomer(w1, w2);
		if (load_stock)LoadStock(w1,
w2);
		if (load_orders)LoadOrd(w1, w2);
		if (load_new_order)LoadNew(w1,
w2);
		return 0;
	}
}


/*********Warehouse********/


ID   w_id;
TEXT w_name[10+1];
TEXT w_street_1[20+1];
TEXT w_street_2[20+1];
TEXT w_city[20+1];
TEXT w_state[2+1];
TEXT w_zip[9+1];
FLOAT w_tax;
MONEY w_ytd;


int bulk_w;
void
LoadWarehouse(w1, w2)
	ID w1, w2;
{

	print_prologue("Warehouse");
	bulk_w = bulk_open("tpcc", "ware-
house", password, &blkdesc, &sliceNum);

	for (warehouse=w1; warehouse<=w2; ware-
house++)
	{
		printf("Loading warehouse for
warehouse %d\n", warehouse);

		w_id = warehouse;
		MakeAlphaString(6, 10, w_name);
		MakeAddress(w_street_1,
w_street_2, w_city, w_state, w_zip);

		w_tax = RandomNumber(0, 2000) /
10000.0;
		w_ytd = 300000.00 * 100;

		begin_warehouse_load();
		warehouse_load();

		printf("loaded warehouse for
warehouse %d\n", warehouse);
	}
	end_warehouse_load();
	return;
}

void
begin_warehouse_load()
{
	int    i = 1;


	bulk_bind(bulk_w, i++, "w_id",   &w_id,
ID_T,blkdesc);
	bulk_bind(bulk_w, i++, "w_name",
w_name, TEXT_T,blkdesc);
	bulk_bind(bulk_w, i++, "w_street_1",
w_street_1, TEXT_T,blkdesc);
	bulk_bind(bulk_w, i++, "w_street_2",
w_street_2, TEXT_T,blkdesc);
	bulk_bind(bulk_w, i++, "w_city",
w_city, TEXT_T,blkdesc);
	bulk_bind(bulk_w, i++, "w_state",
w_state, TEXT_T,blkdesc);
	bulk_bind(bulk_w, i++, "w_zip", w_zip,
TEXT_T,blkdesc);
	bulk_bind(bulk_w, i++, "w_tax", &w_tax,
FLOAT_T,blkdesc);
	bulk_bind(bulk_w, i++, "w_ytd", &w_ytd,
MONEY_T,blkdesc);
	return;
}

void
warehouse_load()
{
	debug("Loading Warehouse %d\n", w_id);
	bulk_load(bulk_w, blkdesc);
}

void
end_warehouse_load()
{
	bulk_close(bulk_w, blkdesc);
```

```
}



/**********District*********/



ID d_id;
ID d_w_id;
TEXT d_name[10+1];
TEXT d_street_1[20+1];
TEXT d_street_2[20+1];
TEXT d_city[20+1];
TEXT d_state[2+1];
TEXT d_zip[9+1];
FLOAT d_tax;
MONEY d_ytd;
ID   d_next_o_id;

int bulk_d;

void
LoadDistrict(w1, w2)
       ID w1, w2;
{
       ID w_id;

       print_prologue("District");
       bulk_d = bulk_open("tpcc", "district",
password, &blkdesc, &sliceNum);
       for (w_id=w1; w_id<=w2; w_id++)
       {
              printf("Loading districts for
warehouse %d\n", w_id);

              d_w_id = w_id;
              d_ytd = 30000.00 * 100;
              d_next_o_id = 3001;

              for (d_id = 1;  d_id <=
DIST_PER_WARE;  d_id++)
              {

                     MakeAlphaString(6, 10,
d_name);
                     MakeAddress(d_street_1,
d_street_2, d_city, d_state, d_zip);
                     d_tax = RandomNum-
ber(0,2000) / 10000.0;

                     begin_district_load();
                     district_load();
              }
              printf("loaded district for ware-
house %d\n", w_id);
       }
       end_district_load();
       return;
}



void
begin_district_load()
{
       int    i = 1;


       bulk_bind(bulk_d, i++, "d_id", &d_id,
ID_T,blkdesc);
       bulk_bind(bulk_d, i++, "d_w_id",
&d_w_id, ID_T,blkdesc);
       bulk_bind(bulk_d, i++, "d_name",
d_name, TEXT_T,blkdesc);
       bulk_bind(bulk_d, i++, "d_street_1",
d_street_1, TEXT_T,blkdesc);
       bulk_bind(bulk_d, i++, "d_street_2",
d_street_2, TEXT_T,blkdesc);
       bulk_bind(bulk_d, i++, "d_city",
d_city, TEXT_T,blkdesc);
       bulk_bind(bulk_d, i++, "d_state",
d_state, TEXT_T,blkdesc);
       bulk_bind(bulk_d, i++, "d_zip", d_zip,
TEXT_T,blkdesc);
       bulk_bind(bulk_d, i++, "d_tax", &d_tax,
FLOAT_T,blkdesc);
       bulk_bind(bulk_d, i++, "d_ytd", &d_ytd,
MONEY_T,blkdesc);
       bulk_bind(bulk_d, i++, "d_next_o_id",
&d_next_o_id, ID_T,blkdesc);
       return;
}
void
district_load()
{
       debug("District d_id=%d  w_id=%d
d_name=%s d_street_1=%s d_street_2=%s
d_city=%s d_state=%s d_zip=%s d_tax=%f
d_ytd=%f d_next_o_id=%d\n", d_id, d_w_id,
d_name, d_street_1, d_street_2, d_city,
d_state, d_zip, d_tax, d_ytd, d_next_o_id);
       bulk_load(bulk_d, blkdesc);
       return;
}
void
end_district_load()
{
       bulk_close(bulk_d, blkdesc);
```

```
	return;
}



/*******Item*********/



ID i_id;
ID i_im_id;
TEXT i_name[24+1];
MONEY i_price;
TEXT  i_data[50+1];

int bulk_i;
void
LoadItems()
{

	print_prologue_item("Item");

	bulk_i = bulk_open("tpcc", "item",
password, &blkdesc, &sliceNum);

	/* select exactly 10% of items to be
labeled "original" */

	/* do for each item */
	for (i_id=i1;   i_id <= i2; i_id++)
	{

		/* Generate Item Data */
		MakeAlphaString(14, 24, i_name);
		i_price = RandomNum-
ber(100,10000);
		MakeAlphaString(26, 50, i_data);
		if (perm[i_id] <= (MAX-
ITEMS+9)/10)
			Original(i_data);

		/* Generate i_im_id for V 3.0 */
		i_im_id = RandomNumber(1, 10000);

		begin_item_load();
		item_load();
	 }

	end_item_load();
	return;
}
```

```
void
begin_item_load()
{
	int    i = 1;


	/* bind the variables to the sybase col-
umns */
	bulk_bind(bulk_i, i++, "i_id", &i_id,
ID_T,blkdesc);
	bulk_bind(bulk_i, i++, "i_im_id",
&i_im_id, ID_T,blkdesc);
	bulk_bind(bulk_i, i++, "i_name",
i_name, TEXT_T,blkdesc);
	bulk_bind(bulk_i, i++, "i_price",
&i_price, MONEY_T,blkdesc);
	bulk_bind(bulk_i, i++, "i_data",
i_data, TEXT_T,blkdesc);
	return;
}



void
item_load()
{
	debug("i_id=%3d price=%5.2f data=%s\n",
	   i_id, i_price, i_data);
	bulk_load(bulk_i, blkdesc);
	return;
}

void
end_item_load()
{
	bulk_close(bulk_i, blkdesc);
	return;
}


/******History************/



ID h_c_id;
ID h_c_d_id;
ID h_c_w_id;
ID h_d_id;
ID h_w_id;
DATE h_date;
MONEY h_amount;
TEXT h_data[24+1];

int bulk_h;
void
```

```
LoadHist(w1, w2)
       ID w1, w2;
{
       ID w_id;
       ID d_id, c_id;

       print_prologue(“History”);
       bulk_h = bulk_open(“tpcc”, “history”,
password, &blkdesc, &sliceNum);
       for (w_id=w1; w_id<=w2; w_id++)
       {

              for (d_id=1;  d_id <=
DIST_PER_WARE; d_id++)
              {
                     for (c_id=1; c_id <=
CUST_PER_DIST; c_id++)
                            LoadCustHist(w_id,
d_id, c_id);
              }

              printf(“\nLoaded history for
warehouse %d\n”, w_id);
       }
       end_history_load();
       return;
}

void
LoadCustHist(w_id, d_id, c_id)
       ID w_id, d_id, c_id;
{

       h_c_id = c_id;
       h_c_d_id = d_id;
       h_c_w_id = w_id;
       h_d_id = d_id;
       h_w_id = w_id;
       h_amount = 10.0 * 100;
       MakeAlphaString(12, 24, h_data);
       datetime(&h_date);
       begin_history_load();
       history_load();
       return;
}


void
begin_history_load()
{
       int    i = 1;


       bulk_bind(bulk_h, i++, “h_c_id”,
&h_c_id,   ID_T,blkdesc);
       bulk_bind(bulk_h, i++, “h_c_d_id”,
&h_c_d_id, ID_T,blkdesc);
       bulk_bind(bulk_h, i++, “h_c_w_id”,
&h_c_w_id, ID_T,blkdesc);
       bulk_bind(bulk_h, i++, “h_d_id”,
&h_d_id,   ID_T,blkdesc);
       bulk_bind(bulk_h, i++, “h_w_id”,
&h_w_id,   ID_T,blkdesc);
       bulk_bind(bulk_h, i++, “h_date”,
&h_date,   DATE_T,blkdesc);
       bulk_bind(bulk_h, i++, “h_amount”,
&h_amount, MONEY_T,blkdesc);
       bulk_bind(bulk_h, i++, “h_data”,
h_data,    TEXT_T,blkdesc);
       return;
}
void
history_load()
{

       debug(“h_c_id=%d  h_amount=%g\n”,
h_c_id, h_amount);
       bulk_load(bulk_h, blkdesc);
       return;
}

void
end_history_load()
{
       bulk_close(bulk_h, blkdesc);
       return;
}




/*****Customer**********/

/* static variables containing fields for cus-
tomer record */
ID c_id;
ID c_d_id;
ID c_w_id;
TEXT c_first[16+1];
TEXT c_middle[2+1] = “OE”;
TEXT c_last[16+1];
TEXT c_street_1[20+1];
TEXT c_street_2[20+1];
TEXT c_city[20+1];
TEXT c_state[2+1];
TEXT c_zip[9+1];
TEXT c_phone[16+1];
DATE c_since;
TEXT c_credit[2+1] = “?C”;
MONEY c_credit_lim = 50000.0 * 100;
```

```
FLOAT c_discount;
MONEY c_balance = -10.0 * 100;
MONEY c_ytd_payment = 10.0 * 100;
COUNT c_payment_cnt = 1;
COUNT c_delivery_cnt = 0;
TEXT c_data[500+1];
TEXT c_data1[250+1];
TEXT c_data2[250+1];
ID     len;

int bulk_c;

void
LoadCustomer(w1, w2)
       ID w1, w2;
{
       ID w_id;

       print_prologue("Customer");
       bulk_c = bulk_open("tpcc", "customer",
password, &blkdesc, &sliceNum);
       for (w_id=w1; w_id<=w2; w_id++)
       {
              Customer(w_id);
              printf("\nLoaded customer for
warehouse %d into partition %d \n", w_id, sli-
ceNum);
       }
       end_customer_load();
       return;
}
void
Customer(w_id)
       int w_id;
{
       BitVector bad-
credit[DIST_PER_WARE][(3000+WSZ-1)/WSZ], *
bmp;
       int i, j;
       ID d_id;

       /* Mark exactly 10% of customers as hav-
ing bad credit */
       for (d_id=1;  d_id <= DIST_PER_WARE;
d_id++)
       {
              bmp = badcredit[d_id-1];
              for (i=0; i<(3000+WSZ-1)/WSZ;
i++)
                     bmp[i] = (BitVec-
tor)0x0000;
              for (i=0; i<(3000+9)/10; i++)
              {
                     do {
                            j = RandomNum-
ber(0,3000-1);
                     } while (nthbit(bmp,j));
                     setbit(bmp,j);
              }
       }

       c_w_id = w_id;
       for (i=0; i<CUST_PER_DIST; i++)
       {
              c_id = i+1;
              for (d_id=1;  d_id <=
DIST_PER_WARE;  d_id++)
              {
                     c_d_id = d_id;
                     LastName(i<1000?i:NURandomNum-
ber(255,NURAND_C,0,999),c_last);
                     MakeAlphaString(8, 16,
c_first);
                     MakeAd-
dress(c_street_1,c_street_2,c_city,c_state,c_z
ip);
                     MakeNumberString(16, 16,
c_phone);
                     MakeAlphaString(300, 500,
c_data);
                     datetime(&c_since);
                     c_credit[0] = nthbit(bad-
credit[d_id-1],i) ? 'B' : 'G';
                     c_discount = RandomNum-
ber(0, 5000) / 10000.0;

                     /* Break the string c_data
into 2 pieces */
                     len = strlen(c_data);
                     if (len > 250)
                            {
                            memcpy(c_data1,
c_data, 250);
                            c_data1[250]='\0';
                            memcpy(c_data2,
c_data+250, len-250 +1);
                            }

                     else
                            {
                            memcpy(c_data1,
c_data, len+1);
                            strcpy(c_data2,"");
                            }
                     begin_customer_load();
                     customer_load();
              }
       }
       return;
}
```

```
void
begin_customer_load()
{
       int    i = 1;


       bulk_bind(bulk_c, i++, “c_id”,  &c_id,
ID_T,blkdesc);
       bulk_bind(bulk_c, i++, “c_d_id”,
&c_d_id, ID_T,blkdesc);
       bulk_bind(bulk_c, i++, “c_w_id”,
&c_w_id, ID_T,blkdesc);
       bulk_bind(bulk_c, i++, “c_first”,
c_first, TEXT_T,blkdesc);
       bulk_bind(bulk_c, i++, “c_middle”,
c_middle, TEXT_T,blkdesc);
       bulk_bind(bulk_c, i++, “c_last”,c_last,
TEXT_T,blkdesc);
       bulk_bind(bulk_c, i++, “street_1”,
c_street_1 , TEXT_T,blkdesc);
       bulk_bind(bulk_c, i++, “street_2”,
c_street_2,  TEXT_T,blkdesc);
       bulk_bind(bulk_c, i++, “c_city”,c_city,
TEXT_T,blkdesc);
       bulk_bind(bulk_c, i++, “c_state”,
c_state, TEXT_T,blkdesc);
       bulk_bind(bulk_c, i++, “c_zip”, c_zip,
TEXT_T,blkdesc);
       bulk_bind(bulk_c, i++, “c_phone”,
c_phone, TEXT_T,blkdesc);
       bulk_bind(bulk_c, i++, “c_since”,
&c_since, DATE_T,blkdesc);
       bulk_bind(bulk_c, i++, “c_credit”,
c_credit,TEXT_T,blkdesc);
       bulk_bind(bulk_c, i++, “c_credit_lim”,
&c_credit_lim, MONEY_T,blkdesc);
       bulk_bind(bulk_c, i++, “c_discount”,
&c_discount, FLOAT_T,blkdesc);
       bulk_bind(bulk_c, i++,
“c_delivery_cnt”, &c_delivery_cnt,
COUNT_T,blkdesc);
       bulk_bind(bulk_c, i++,
“c_payment_cnt”,&c_payment_cnt, COUNT_T,blk-
desc);
       bulk_bind(bulk_c, i++, “c_balance”,
&c_balance, MONEY_T,blkdesc);
       bulk_bind(bulk_c, i++, “c_ytd_payment”,
&c_ytd_payment, MONEY_T,blkdesc);
       bulk_bind(bulk_c, i++, “c_data_1”,
c_data1, TEXT_T,blkdesc);
       bulk_bind(bulk_c, i++, “c_data_2”,
c_data2, TEXT_T,blkdesc);
       return;
}
void
customer_load()
{
       debug(“c_id=%-5d  d_id=%-5d  w_id=%-5d
c_last=%s\n”,
          c_id,c_d_id,  c_w_id,  c_last);

       /* load the data */
       bulk_load(bulk_c, blkdesc);
       return;
}
void
end_customer_load()
{
       bulk_close(bulk_c, blkdesc);
       return;
}




/****Order, Order line, New order****/

/* Order row */
ID o_id;
ID o_c_id;
ID o_d_id;
ID o_w_id;
DATE o_entry_d;
ID o_carrier_id;
COUNT o_ol_cnt;
LOGICAL o_all_local;

/* Order line row */
ID   ol_o_id;
ID   ol_d_id;
ID   ol_w_id;
ID   ol_number;
ID   ol_i_id;
ID   ol_supply_w_id;
DATE ol_delivery_d;
COUNT ol_quantity;
MONEY ol_amount;
TEXT  ol_dist_info[24+1];

/* new order row */
ID no_o_id;
ID no_d_id;
ID no_w_id;


int  o_bulk;
int ol_bulk;
int no_bulk;
```

```
void
LoadOrd(w1, w2)
       ID w1, w2;
{
       ID w_id;
       ID d_id;

       print_prologue(“Order”);
       ol_bulk = bulk_open(“tpcc”,
“order_line”, password, &blkdesc1, &sliceNum);
       o_bulk = bulk_open(“tpcc”, “orders”,
password, &blkdesc, &sliceNum);
       for (w_id=w1; w_id<=w2; w_id++)
       {

              for (d_id = 1;  d_id <=
DIST_PER_WARE;  d_id++)
                     Orders(w_id, d_id);

              printf(“\nLoaded order +
order_line for warehouse %d\n”, w_id);
       }
       end_order_line_load();
       end_order_load();
       return;
}
void
LoadNew(w1, w2)
       ID w1, w2;
{
       ID w_id;
       ID d_id;

       print_prologue(“NewOrder”);
       no_bulk = bulk_open(“tpcc”,
“new_order”, password, &blkdesc, &sliceNum);
       for (w_id=w1; w_id<=w2; w_id++)
       {
              for (d_id = 1;  d_id <=
DIST_PER_WARE;  d_id++)
              {
                     no_d_id = d_id;
                     no_w_id = w_id;
                     for (no_o_id=2101;
no_o_id <= ORD_PER_DIST; no_o_id++)
                     {

begin_new_order_load();
                            new_order_load();
                     }
              }
              printf(“\nLoaded new_order for
warehouse %d\n”, w_id);
       }
       end_new_order_load();
       return;
}
void
Orders(w_id, d_id)
       ID w_id;
       ID d_id;
{
       int cust[ORD_PER_DIST+1];
       int ol_cnt[ORD_PER_DIST+1], sum;
       ID ol;

       printf(“\nLoading orders and order
lines  for warehouse %d district %d\n”,
        w_id, d_id);

       RandomPermutation(cust, ORD_PER_DIST);


       for (o_id = 1, sum=0;  o_id <=
ORD_PER_DIST;  o_id++)
              sum += (ol_cnt[o_id] = Random-
Number(5, 15));

        while (sum > 10*ORD_PER_DIST)
        {
                do {
                        o_id = RandomNum-
ber(1,ORD_PER_DIST);
                } while (ol_cnt[o_id]==5);
                ol_cnt[o_id]--;
                sum--;
        }

        while (sum < 10*ORD_PER_DIST)
        {
                do {
                        o_id = RandomNum-
ber(1,ORD_PER_DIST);
                } while (ol_cnt[o_id]==15);
                ol_cnt[o_id]++;
                sum++;
        }

       for (o_id = 1;  o_id <= ORD_PER_DIST;
o_id++)
       {
       o_c_id = cust[o_id];
       o_d_id = d_id;
       o_w_id = w_id;
       datetime(&o_entry_d);
       if (o_id <= 2100)
       o_carrier_id = RandomNumber(1,10);
       else o_carrier_id = -1;
       o_ol_cnt = ol_cnt[o_id];
       /* o_ol_cnt = RandomNumber(5, 15); */
```

```
	o_all_local = 1;
	begin_order_load();
	order_load();

	for (ol=1;  ol<=o_ol_cnt; ol++)
		OrderLine(ol);

	}
	return;
}



void
OrderLine(ol)
	ID ol;
{

	ol_o_id = o_id;
	ol_d_id = o_d_id;
	ol_w_id = o_w_id;
	ol_number = ol;
	ol_i_id = RandomNumber(1, MAXITEMS);
	ol_supply_w_id = o_w_id;
	ol_delivery_d = o_entry_d;
	ol_quantity = 5;
	if (o_id <= 2100) ol_amount = 0;
	else		ol_amount = RandomNum-
ber(1, 999999);
	MakeAlphaString(24, 24, ol_dist_info);
	begin_order_line_load();
	order_line_load();
	return;
}

void
NewOrder(w_id, d_id)
	ID w_id, d_id;
{

	no_d_id = o_d_id;
	no_w_id = o_w_id;
	for (no_o_id=2101;  no_o_id <=
ORD_PER_DIST; no_o_id++)
	{
		begin_new_order_load();
		new_order_load();
	}
	return;
}



void
begin_order_load()
{
	int	i = 1;


	bulk_bind(o_bulk, i++, “o_id”, &o_id,
ID_T,blkdesc);
	bulk_bind(o_bulk, i++, “o_c_id”,
&o_c_id, ID_T,blkdesc);
	bulk_bind(o_bulk, i++, “o_d_id”,
&o_d_id, ID_T,blkdesc);
	bulk_bind(o_bulk, i++, “o_w_id”,
&o_w_id, ID_T,blkdesc);
	bulk_bind(o_bulk, i++, “o_entry_d”,
&o_entry_d, DATE_T,blkdesc);
	bulk_bind(o_bulk, i++, “o_carrier_id”,
&o_carrier_id, ID_T,blkdesc);
	bulk_bind(o_bulk, i++, “o_ol_cnt”,
&o_ol_cnt, COUNT_T,blkdesc);
	bulk_bind(o_bulk, i++, “o_all_local”,
&o_all_local, LOGICAL_T,blkdesc);
	return;
}
void
order_load()
{
	debug(“o_id=%d o_c_id=%d  count=%d\n”,
o_id, o_c_id, o_ol_cnt);
	bulk_load(o_bulk, blkdesc);
	return;
}
void
end_order_load()
{
	bulk_close(o_bulk, blkdesc);
	return;
}



void
begin_order_line_load()
{
	int	i = 1;


	bulk_bind(ol_bulk, i++, “ol_o_id”,
&ol_o_id, ID_T,blkdesc1);
	bulk_bind(ol_bulk, i++, “ol_d_id”,
&ol_d_id, ID_T,blkdesc1);
	bulk_bind(ol_bulk, i++, “ol_w_id”,
&ol_w_id, ID_T,blkdesc1);
	bulk_bind(ol_bulk, i++, “ol_number”,
&ol_number, ID_T,blkdesc1);
	bulk_bind(ol_bulk, i++, “ol_i_id”,
&ol_i_id, ID_T,blkdesc1);
```

```
	bulk_bind(ol_bulk, i++,
“ol_supply_w_id”, &ol_supply_w_id,
ID_T,blkdesc1);
	bulk_bind(ol_bulk, i++,
“ol_delivery_d”, &ol_delivery_d,
DATE_T,blkdesc1);
	bulk_bind(ol_bulk, i++, “ol_quantity”,
&ol_quantity, COUNT_T,blkdesc1);
	bulk_bind(ol_bulk, i++, “ol_amount”,
&ol_amount, MONEY_T,blkdesc1);
	bulk_bind(ol_bulk, i++, “ol_dist_info”,
ol_dist_info, TEXT_T,blkdesc1);
	return;
}
void
order_line_load()
{
	debug(“  ol_o_id=%d  ol_number=%d
ol_amount=%g\n”,
		ol_o_id,ol_number,ol_amount);
	bulk_load(ol_bulk, blkdesc1);
	return;
}

void
end_order_line_load()
{
	bulk_close(ol_bulk, blkdesc1);
	return;
}
void
begin_new_order_load()
{
	int    i = 1;


	bulk_bind(no_bulk, i++, “no_o_id”,
&no_o_id, ID_T,blkdesc);
	bulk_bind(no_bulk, i++, “no_d_id”,
&no_d_id, ID_T,blkdesc);
	bulk_bind(no_bulk, i++, “no_w_id”,
&no_w_id, ID_T,blkdesc);
	return;
}
void
new_order_load()
{
	debug(“  no_o_id=%d  \n”, no_o_id);
	bulk_load(no_bulk, blkdesc);
	return;
}

void
end_new_order_load()
{
	bulk_close(no_bulk, blkdesc);
	return;
}


/*******Stock********/


ID s_i_id;
ID s_w_id;
COUNT s_quantity;
TEXT s_dist_01[24+1];
TEXT s_dist_02[24+1];
TEXT s_dist_03[24+1];
TEXT s_dist_04[24+1];
TEXT s_dist_05[24+1];
TEXT s_dist_06[24+1];
TEXT s_dist_07[24+1];
TEXT s_dist_08[24+1];
TEXT s_dist_09[24+1];
TEXT s_dist_10[24+1];
COUNT s_ytd;
COUNT s_order_cnt;
COUNT s_remote_cnt;
TEXT s_data[50+1];

int bulk_s;

/*
** On loading stock in major order of item_id:
** 10% of the MAXITEMS items in each warehouse
need to marked as original
** (i.e., s_data like ‘%ORIGINAL%’.)  This is
a bit harder to do when we
** load by item number, rather than by ware-
houses.  The trick is to first
** generate a huge WAREBATCH * MAXITEMS bit-
map, initialize all bits to zero,
** and then set 10% of bits in each row to 1.
While loading item i in
** warehouse w, we simply lookup bitmap[w][i]
to see whether it needs to
** be marked as original.
*/

#ifdef _NTINTEL
/* On NT stack overflow happens when you
define the following on stack
*/
	BitVector original[WAREBATCH][((MAX-
ITEMS+(WSZ-1))/WSZ)] ;
#endif
void
LoadStock(w1, w2)
	ID w1, w2;
```

```
{
       ID w_id;


       int w, i, j;

       print_prologue_item(“Stock”);
       if (w2-w1+1 > WAREBATCH)
       {
              fprintf(stderr, “Can’t load stock
for %d warehouses.\n”,
                     w2-w1+1);
              fprintf(stderr, “Please use
batches of %d.\n”, WAREBATCH);
       }

       for (w=w1; w<=w2; w++)
       {
              bmp = original[w-w1];
              /* Mark all items as not “origi-
nal” */
              for (i=0; i<(MAXITEMS+(WSZ-
1))/WSZ; i++)
                     bmp[i] = (BitVec-
tor)0x0000;
              /* Mark exactly 10% of items as
“original” */
              for (i=0; i<(MAXITEMS+9)/10; i++)
              {
                     do {
                            j = RandomNum-
ber(0,MAXITEMS-1);
                     } while (nthbit(bmp,j));
                     setbit(bmp,j);
              }
       }

       printf(“Loading stock for warehouse %d
to %d.\n”, w1, w2);
       bulk_s = bulk_open(“tpcc”, “stock”,
password, &blkdesc, &sliceNum);

       /* do for each item */
       for (s_i_id=i1;   s_i_id <= i2;
s_i_id++)
       {
              for (w_id=w1; w_id<=w2; w_id++)
              {
              /* Generate Stock Data */
                     s_w_id = w_id;
                     s_quantity = RandomNum-
ber(10,100);
                     MakeAlphaString(24, 24,
s_dist_01);
                     MakeAlphaString(24, 24,
s_dist_02);
                     MakeAlphaString(24, 24,
s_dist_03);
                     MakeAlphaString(24, 24,
s_dist_04);
                     MakeAlphaString(24, 24,
s_dist_05);
                     MakeAlphaString(24, 24,
s_dist_06);
                     MakeAlphaString(24, 24,
s_dist_07);
                     MakeAlphaString(24, 24,
s_dist_08);
                     MakeAlphaString(24, 24,
s_dist_09);
                     MakeAlphaString(24, 24,
s_dist_10);
                     s_ytd = 0;
                     s_order_cnt = 0;
                     s_remote_cnt = 0;
                     MakeAlphaString(26, 50,
s_data);
                     if (nthbit(original[w_id-
w1],s_i_id-1))
                     {
                            Original(s_data);
                     }
                     begin_stock_load();
                     stock_load();
              }
       }
       end_stock_load();
       printf(“\nLoaded stock for warehouses
%d to %d.\n”, w1, w2);
       return;
}
void
begin_stock_load()
{
       int    i = 1;


       bulk_bind(bulk_s, i++, “s_i_id”,
&s_i_id, ID_T,blkdesc);
       bulk_bind(bulk_s, i++, “s_w_id”,
&s_w_id, ID_T,blkdesc);
       bulk_bind(bulk_s, i++, “s_quantity”,
&s_quantity, COUNT_T,blkdesc);
       bulk_bind(bulk_s, i++, “s_ytd”, &s_ytd,
COUNT_T,blkdesc);
       bulk_bind(bulk_s, i++, “s_order_cnt”,
&s_order_cnt, COUNT_T,blkdesc);
       bulk_bind(bulk_s, i++, “s_remote_cnt”,
&s_remote_cnt, COUNT_T,blkdesc);
       bulk_bind(bulk_s, i++, “s_dist_01”,
```

```
s_dist_01, TEXT_T,blkdesc);
       bulk_bind(bulk_s, i++, “s_dist_02”,
s_dist_02, TEXT_T,blkdesc);
       bulk_bind(bulk_s, i++, “s_dist_03”,
s_dist_03, TEXT_T,blkdesc);
       bulk_bind(bulk_s, i++, “s_dist_04”,
s_dist_04, TEXT_T,blkdesc);
       bulk_bind(bulk_s, i++, “s_dist_05”,
s_dist_05, TEXT_T,blkdesc);
       bulk_bind(bulk_s, i++, “s_dist_06”,
s_dist_06, TEXT_T,blkdesc);
       bulk_bind(bulk_s, i++, “s_dist_07”,
s_dist_07, TEXT_T,blkdesc);
       bulk_bind(bulk_s, i++, “s_dist_08”,
s_dist_08, TEXT_T,blkdesc);
       bulk_bind(bulk_s, i++, “s_dist_09”,
s_dist_09, TEXT_T,blkdesc);
       bulk_bind(bulk_s, i++, “s_dist_10”,
s_dist_10, TEXT_T,blkdesc);
       bulk_bind(bulk_s, i++, “s_data”,s_data,
TEXT_T,blkdesc);
       return;
}



void
stock_load()
{
       debug(“s_i_id=%d w_id=%d s_data=%s\n”,
          s_i_id, s_w_id,   s_data);
       bulk_load(bulk_s, blkdesc);
       return;
}

void
end_stock_load()
{
       bulk_close(bulk_s, blkdesc);
       return;
}


void
test(){
       return;
}

void
getargs(argc, argv)

/*****************************************
*******************
configure configures the load stuff
   By default, loads all the tables for a the
specified warehouse.
               When loading warehouse 1, also
loads the item table.
*****************************************
*********************/
       int argc;
       char **argv;
{
       if (argc == 1)
       {
              printf(“Usage: %s <table>
<w_first> [<w_last>]\n”, argv[0]);
              exit(0);
       }

       /* define the defaults */
       load_item = load_warehouse =
load_district = load_history =
       load_orders = load_new_order =
load_order_line =
       load_customer = load_stock = NO;


       if      (strcmp(argv[1], “warehouse”)
== 0) load_warehouse = YES;
       else if (strcmp(argv[1], “district”) ==
0)  load_district = YES;
       else if (strcmp(argv[1], “stock”) == 0)
load_stock = YES;
       else if (strcmp(argv[1], “item”) == 0)
load_item = YES;
       else if (strcmp(argv[1], “history”) ==
0)  load_history = YES;
       else if (strcmp(argv[1], “orders”) ==
0)  load_orders = YES;
       else if (strcmp(argv[1], “customer”) ==
0)  load_customer = YES;
       else if (strcmp(argv[1], “new_order”)
==0) load_new_order = YES;
       else if (strcmp(argv[1], “-v”) ==0)
       {
              printf(“Usage: %s <table>
<w_first> [<w_last>]\n”, argv[0]);
              exit(0);
       }
       else
       {
               printf(“%s is not a valid table
name\n”, argv[1]);
               exit(0);


       }

       /* Set the w1 and w2 to argv[2] and
arg[3] */
```

```
	if (argc < 3)
	{
		printf("Usage: %s <table>
<w_first> [<w_last>]\n", argv[0]);
		exit(1);
	}
	{
		w1 = atoi(argv[2]);
		if (argc >= 3)
			w2 = atoi(argv[3]);
		else
			w2 = w1;
	}

	i1 = 1;
	i2 = MAXITEMS;
	/* Get the password for sa */
	if (argc > 4)
	strcpy(password,argv[4]);

	if (argc > 5)
	{
		numOfSlices = atoi(argv[5]);
	}
	else
	{
		numOfSlices = INVALID_SLICE_NUM;
	}
	printf (" Number of partition is %d \n",
numOfSlices);

	/* Check if warehouse is within the
range */
	if (w1 <= 0 || w1 > w2)
	{
		printf("Warehouse id is out of
range\n");
		exit(0);
	}
	return;
}

#ifndef _NTINTEL
double drand48();
#endif
void
MakeAddress(str1, str2, city, state, zip)
	TEXT str1[20+1];
	TEXT str2[20+1];
	TEXT city[20+1];
	TEXT state[2+1];
	TEXT zip[9+1];
{
	MakeAlphaString(10,20,str1);
	MakeAlphaString(10,20,str2);
	MakeAlphaString(10,20,city);
	MakeAlphaString(2,2,state);
	MakezipString(0,9999,zip);

	/* Changed for TPCC V 3.0 */
	strcat(zip, "11111");

	return;
}


void
LastName(num, name)
/*****************************************
****************
Lastname generates a lastname from a number.
*****************************************
****************/
	int num;
	char name[20+1];
{
	static char *n[] = {"BAR", "OUGHT",
"ABLE", "PRI", "PRES",
                                        "ESE",
"ANTI", "CALLY", "ATION", "EING"};

	strcpy(name, n[(num/100)%10]);
	strcat(name, n[(num/10) %10]);
	strcat(name, n[(num/1)  %10]);
	return;
}


int MakeNumberString(min, max, num)
	int min;
	int max;
	TEXT num[];
{
	static char digit[]="0123456789";
	int length;
	int i;

	length = RandomNumber(min, max);

	for (i=0;  i<length;  i++)
	num[i] = digit[RandomNumber(0,9)];
	num[length] = '\0';

	return length;
}

int MakezipString(min, max, num)
	  int min;
	  int max;
	  TEXT num[];
```

```
{
        static char digit[]="0123456789";
        int length;
        int i;

        length = 4;

        for (i=0;  i<length;  i++)
        num[i] = digit[RandomNumber(0,9)];
        num[length] = '\0';

        return length;
}



int MakeAlphaString(min, max, str)
       int min;
       int max;
       TEXT str[];
{
       static char character[] =

"abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRS
TUVWXYZ0123456789";
       int length;
       int i;

       length = RandomNumber(min, max);

       for (i=0;  i<length;  i++)
       str[i] = character[RandomNumber(0,
sizeof(character)-2)];
       str[length] = '\0';

       return length;
}

void
Original(str)
       TEXT str[];
{
       int pos;
       int len;

       len = strlen(str);
       if (len < 8) return;

       pos = RandomNumber(0,len-8);

       str[pos+0] = 'O';
       str[pos+1] = 'R';
       str[pos+2] = 'I';
       str[pos+3] = 'G';
       str[pos+4] = 'I';
       str[pos+5] = 'N';
       str[pos+6] = 'A';
       str[pos+7] = 'L';
       return;
}



void
RandomPermutation(perm, n)
       int perm[];
       int n;
{
       int i, r, t;

       /* generate the identity permutation to
start with */
       for (i=1; i<=n; i++)
       perm[i] = i;

       /* randomly shuffle the permutation */
       for (i=1; i<=n; i++)
       {
       r = RandomNumber(i, n);
       t = perm[i]; perm[i] = perm[r]; perm[r]
= t;
       }
       return;
}



void
Randomize()
{
#ifdef _NTINTEL
       srand(time(0)+_getpid());
#else
       srand48(time(0)+getpid());
#endif
       return;
}


int
RandomNumber(int min, int max)
{
       int r;
#ifdef _NTINTEL
       r =
(int)((float)rand()/(float)(RAND_MAX + 1) *
(max - min + 1)) + min;
#else
       r = (int)(drand48() * (max - min + 1)) +
```

```
min;
#endif
       return r;
}


int
NURandomNumber(int a, int c, int min, int max)
{
       int r;

       r = ((RandomNumber(0, a) | RandomNum-
ber(min, max)) + c)
       % (max - min + 1) + min;

       return r;
}

void
donothing(const char *fmt, ...)
{
       return;
}
```

tpcc_ld_loader.h

```
#ifndef TPCC_INCLUDED
#define TPCC_INCLUDED

#include “sybfront.h”
#include “sybdb.h”
#ifdef _NTINTEL
#include <sys/timeb.h>
#include <stdlib.h>
#include <process.h>
#endif
#include <time.h>


/* Population constants */
#ifdef CACHED
#define MAXITEMS 10000
#define CUST_PER_DIST 300
#define DIST_PER_WARE 10
#define ORD_PER_DIST 300
#else
#define MAXITEMS 100000
#define CUST_PER_DIST 3000
#define DIST_PER_WARE 10
#define ORD_PER_DIST 3000
#endif

#defineNURAND_C 123

/* Types of application variables */
typedef int     COUNT;
typedef int     ID;
typedef double  MONEY;
typedef double  FLOAT;
typedef char    TEXT;
typedef struct { int x[2];} DATE;
typedef int     LOGICAL;

typedef enum
    {COUNT_T, ID_T, MONEY_T, FLOAT_T, TEXT_T,
     DATE_T, LOGICAL_T, MAX_T}
    DATA_TYPE;

typedef struct timeval TIME;


#define YES 1
#define NO 0
#define EOF (-1)

#ifndef NULL
#define NULL ((void *)0)
#endif

#define INVALID_SLICE_NUM -1

#ifdef DEBUG
#define debug printf
#else
#define debug donothing
#endif

/* define function types */

CS_INT  cl_err_handler(CS_CONTEXT      *con-
text, CS_CONNECTION   *connection,
CS_CLIENTMSG    *errmsg);
CS_INT server_msg_handler( CS_CONTEXT
*context, CS_CONNECTION  *connection,
CS_SERVERMSG   *srvmsg);
CS_INT cs_err_handler( CS_CONTEXT      *con-
text, CS_CLIENTMSG    *errmsg);
extern int    batch_size;

#endif /* TPCC_INCLUDED */

#ifdef _NTINTEL
#define drand48() rand()
#endif
void LoadWarehouse() ;
void begin_warehouse_load() ;
void warehouse_load() ;
void end_warehouse_load() ;
```

```
void LoadDistrict() ;
void begin_district_load() ;
void district_load() ;
void end_district_load() ;
void LoadItems() ;
void begin_item_load() ;
void item_load() ;
void end_item_load() ;
void LoadHist() ;
void LoadCustHist() ;
void begin_history_load() ;
void history_load() ;
void end_history_load() ;
void LoadCustomer() ;
void Customer() ;
void begin_customer_load() ;
void customer_load() ;
void end_customer_load() ;
void LoadOrd() ;
void LoadNew() ;
void Orders() ;
void OrderLine() ;
void NewOrder() ;
void begin_order_load() ;
void order_load() ;
void end_order_load() ;
void begin_order_line_load() ;
void order_line_load() ;
void end_order_line_load() ;
void begin_new_order_load() ;
void new_order_load() ;
void end_new_order_load() ;
void LoadStock() ;
void begin_stock_load() ;
void stock_load() ;
void end_stock_load() ;
void test() ;
void getargs() ;
void MakeAddress() ;
void LastName() ;
void Original() ;
void RandomPermutation() ;
void Randomize() ;
void datetime();
void bulk_null() ;
void bulk_non_null() ;
void bulk_close() ;
int MakeNumberString() ;
int MakezipString() ;
int MakeAlphaString() ;
int RandomNumber() ;
int NURandomNumber() ;
int bulk_open() ;
CS_RETCODE bulk_load() ;
CS_RETCODEbulk_bind(int, int, char *, void *,
int, CS_BLKDESC *);
void donothing(const char *, ...);
```

Appendix D. Tunable Parameters and Options

ddafs.conf

```
#ident “@(#)ddafs.conf 1.2 93/09/09”

# ddafs is a pseudo driver, hence there is
only one instance
name=”ddafs” parent=”pseudo” instance=0;

max_nics=9;
max_paths_per_nic=1;
max_number_of_servers=9;

# This is the max nReq, i.e. the maximum num-
ber of dafs operations
# that will be oustdanding on one connection
at any time.  This value
# might also be reduced by the DAFS server.

#11/27@21:25 bumped nreq and dropped all xfer
sizes
#2/16/02 - dropped max_requests for new split
queue ddafs.
max_dafs_requests=250;
#alternate spelling ...
max_write_dafs_requests=20;
max_dafs_write_requests=20;
#need many mpool chunks to support 1024
nreq...
recv_buf_pool_ck_size=4096;
send_buf_pool_ck_size=4096;
#force all _reads_and_writes_ to go direct,
and keep max low to save memory.
max_write_inline_size=512;
max_write_direct_size=16384;
max_read_inline_size=512;
max_read_direct_size=16384;
#11/27 end

use_separate_write_sess=1;
max_write_copy_inline_size=0;

# Provider name strings - the following name
must match the driver
# module names that are reported by the
Solaris ‘modinfo’ command.  In
# the order specified, dDAFS will search for
and use the following
# loadable driver modules with its Kernel VI
Providers interface.
#
pnames=”orvi”;

# ddafs cache enable: turn on/off the ddafs
driver cache. Enabling
# this cache will enhance performance in the
io data path by optimizing
# IOMMU loads and unloads (aka enabling dvma).
Default is enabled.
ddafs_cache_enable=1;
okdafs-enabled=0;
force_interrupt_completion=1;
separate_rw_kstats=1;
```

getconf

```
_CS_PATH:
/usr/xpg4/bin:/usr/ccs/bin:/usr/bin:/opt/SUN-
Wspro/bin
_POSIX2_C_VERSION:199209
_POSIX_ARG_MAX:     4096
_POSIX_ASYNCHRONOUS_IO:1
_POSIX_CHILD_MAX:6
_POSIX_CHOWN_RESTRICTED:1
_POSIX_FSYNC:       1
_POSIX_JOB_CONTROL:1
_POSIX_LINK_MAX:8
_POSIX_MAPPED_FILES:1
_POSIX_MAX_CANON:255
_POSIX_MAX_INPUT:255
_POSIX_MEMLOCK:     1
_POSIX_MEMLOCK_RANGE:1
_POSIX_MEMORY_PROTECTION:1
_POSIX_MESSAGE_PASSING:1
_POSIX_NAME_MAX:14
_POSIX_NGROUPS_MAX:0
_POSIX_NO_TRUNC:1
_POSIX_OPEN_MAX:16
_POSIX_PATH_MAX:255
_POSIX_PIPE_BUF:512
_POSIX_PRIORITIZED_IO:undefined
_POSIX_PRIORITY_SCHEDULING:1
_POSIX_REALTIME_SIGNALS:1
_POSIX_SAVED_IDS:1
_POSIX_SEMAPHORES:1
_POSIX_SHARED_MEMORY_OBJECTS:1
_POSIX_SSIZE_MAX:32767
_POSIX_STREAM_MAX:8
_POSIX_SYNCHRONIZED_IO:1
_POSIX_THREADS:     1
_POSIX_TIMERS:      1
_POSIX_TZNAME_MAX:3
_POSIX_VDISABLE:getconf: Invalid argument
(_POSIX_VDISABLE or .)

_POSIX_VERSION:     199506
_XBS5_ILP32_OFF32:1
_XBS5_ILP32_OFF32_CFLAGS:
_XBS5_ILP32_OFF32_LDFLAGS:
_XBS5_ILP32_OFF32_LIBS:
```

```
_XBS5_ILP32_OFF32_LINTFLAGS:
_XBS5_ILP32_OFFBIG:1
_XBS5_ILP32_OFFBIG_CFLAGS:-Xa -Usun -Usparc -
Uunix -Ui386 -D_LARGEFILE_SOURCE -
D_FILE_OFFSET_BITS=64
_XBS5_ILP32_OFFBIG_LDFLAGS:
_XBS5_ILP32_OFFBIG_LIBS:
_XBS5_ILP32_OFFBIG_LINTFLAGS:-Xa -
D_LARGEFILE_SOURCE -D_FILE_OFFSET_BITS=64
_XBS5_LP64_OFF64:1
_XBS5_LP64_OFF64_CFLAGS:-xarch=v9
_XBS5_LP64_OFF64_LDFLAGS:-xarch=v9
_XBS5_LP64_OFF64_LIBS:
_XBS5_LP64_OFF64_LINTFLAGS:-xarch=v9
_XBS5_LPBIG_OFFBIG:1
_XBS5_LPBIG_OFFBIG_CFLAGS:-xarch=v9
_XBS5_LPBIG_OFFBIG_LDFLAGS:-xarch=v9
_XBS5_LPBIG_OFFBIG_LIBS:
_XBS5_LPBIG_OFFBIG_LINTFLAGS:-xarch=v9
_XOPEN_CRYPT:       1
_XOPEN_ENH_I18N:1
_XOPEN_LEGACY:      1
_XOPEN_REALTIME:1
_XOPEN_REALTIME_THREADS:1
_XOPEN_SHM:         1
_XOPEN_UNIX:        1
_XOPEN_VERSION:     4
_XOPEN_XCU_VERSION:4
_XOPEN_XPG2:        undefined
_XOPEN_XPG3:        0
_XOPEN_XPG4:        0
AIO_LISTIO_MAX:     4096
AIO_MAX:            undefined
AIO_PRIO_DELTA_MAX:0
ARG_MAX:            1048320
ATEXIT_MAX:         32
BC_BASE_MAX:        99
BC_DIM_MAX:         2048
BC_SCALE_MAX:       99
BC_STRING_MAX:      1000
CHAR_BIT:           8
CHAR_MAX:           127
CHAR_MIN:           -128
CHARCLASS_NAME_MAX:14
CHILD_MAX:          2053
CLK_TCK:            100
COLL_WEIGHTS_MAX:10
CS_PATH:
/usr/xpg4/bin:/usr/ccs/bin:/usr/bin:/opt/SUN-
Wspro/bin
DELAYTIMER_MAX:     2147483647
EXPR_NEST_MAX:      32
FILESIZEBITS:       43
INT_MAX:            2147483647
INT_MIN:            -2147483648
IOV_MAX:            16
LFS64_CFLAGS:       -D_LARGEFILE64_SOURCE
LFS64_LDFLAGS:
LFS64_LIBS:
LFS64_LINTFLAGS:-D_LARGEFILE64_SOURCE
LFS_CFLAGS:         -D_LARGEFILE_SOURCE -
D_FILE_OFFSET_BITS=64
LFS_LDFLAGS:
LFS_LIBS:
LFS_LINTFLAGS:      -D_LARGEFILE_SOURCE -
D_FILE_OFFSET_BITS=64
LINE_MAX:           2048
LINK_MAX:           65535
LOGIN_NAME_MAX:     9
LONG_BIT:           32
LONG_MAX:           2147483647
LONG_MIN:           -2147483648
MAX_CANON:          getconf: Invalid argument
(MAX_CANON or .)

MAX_INPUT:          getconf: Invalid argument
(MAX_INPUT or .)

MB_LEN_MAX:         5
MQ_OPEN_MAX:        32
MQ_PRIO_MAX:        32
NAME_MAX:           255
NGROUPS_MAX:        16
NL_ARGMAX:          9
NL_LANGMAX:         14
NL_MSGMAX:          32767
NL_NMAX:            1
NL_SETMAX:          255
NL_TEXTMAX:         2048
NZERO:                    20
OPEN_MAX:           256
PAGE_SIZE:          8192
PAGESIZE:           8192
PASS_MAX:           8
PATH:
/usr/xpg4/bin:/usr/ccs/bin:/usr/bin:/opt/SUN-
Wspro/bin
PATH_MAX:           1024
PIPE_BUF:           getconf: Invalid argument
(PIPE_BUF or .)

POSIX2_BC_BASE_MAX:99
POSIX2_BC_DIM_MAX:2048
POSIX2_BC_SCALE_MAX:99
POSIX2_BC_STRING_MAX:1000
POSIX2_C_BIND:      1
POSIX2_C_DEV:       1
POSIX2_CHAR_TERM:1
POSIX2_COLL_WEIGHTS_MAX:10
POSIX2_EXPR_NEST_MAX:32
```

```
POSIX2_FORT_DEV:undefined
POSIX2_FORT_RUN:undefined
POSIX2_LINE_MAX:2048
POSIX2_LOCALEDEF:1
POSIX2_RE_DUP_MAX:255
POSIX2_SW_DEV:      1
POSIX2_UPE:         1
POSIX2_VERSION:     199209
POSIX_THREAD_ATTR_STACKADDR:1
POSIX_THREAD_ATTR_STACKSIZE:1
POSIX_THREAD_PRIO_INHERIT:1
POSIX_THREAD_PRIO_PROTECT:1
POSIX_THREAD_PRIORITY_SCHEDULING:1
POSIX_THREAD_PROCESS_SHARED:1
POSIX_THREAD_SAFE_FUNCTIONS:1
PTHREAD_DESTRUCTOR_ITERATIONS:undefined
PTHREAD_KEYS_MAX:undefined
PTHREAD_STACK_MIN:undefined
PTHREAD_THREADS_MAX:undefined
RE_DUP_MAX:         255
RTSIG_MAX:          8
SCHAR_MAX:          127
SCHAR_MIN:          -128
SEM_NSEMS_MAX:      2147483647
SEM_VALUE_MAX:      2147483647
SHRT_MAX:           32767
SHRT_MIN:           -32768
SIGQUEUE_MAX:       32
SIZE_MAX:           4294967295
SSIZE_MAX:          32767
STREAM_MAX:         256
TIMER_MAX:          32
TMP_MAX:            17576
TTY_NAME_MAX:       128
TZNAME_MAX:         undefined
UCHAR_MAX:          255
UINT_MAX:           4294967295
ULONG_MAX:          4294967295
USHRT_MAX:          65535
WORD_BIT:           32
XBS5_ILP32_OFF32:1
XBS5_ILP32_OFF32_CFLAGS:
XBS5_ILP32_OFF32_LDFLAGS:
XBS5_ILP32_OFF32_LIBS:
XBS5_ILP32_OFF32_LINTFLAGS:
XBS5_ILP32_OFFBIG:1
XBS5_ILP32_OFFBIG_CFLAGS:-Xa -Usun -Usparc -
Uunix -Ui386 -D_LARGEFILE_SOURCE -
D_FILE_OFFSET_BITS=64
XBS5_ILP32_OFFBIG_LDFLAGS:
XBS5_ILP32_OFFBIG_LIBS:
XBS5_ILP32_OFFBIG_LINTFLAGS:-Xa -
D_LARGEFILE_SOURCE -D_FILE_OFFSET_BITS=64
XBS5_LP64_OFF64:1
XBS5_LP64_OFF64_CFLAGS:-xarch=v9
XBS5_LP64_OFF64_LDFLAGS:-xarch=v9
XBS5_LP64_OFF64_LIBS:
XBS5_LP64_OFF64_LINTFLAGS:-xarch=v9
XBS5_LPBIG_OFFBIG:1
XBS5_LPBIG_OFFBIG_CFLAGS:-xarch=v9
XBS5_LPBIG_OFFBIG_LDFLAGS:-xarch=v9
XBS5_LPBIG_OFFBIG_LIBS:
XBS5_LPBIG_OFFBIG_LINTFLAGS:-xarch=v9
```

run.cfg

```
##############################################
#Configuration File for the Sybase SQL Server
#Please read the System Administration Guide
(SAG)
#before changing any of the values in this
file.
##############################################



[Configuration Options]

[General Information]

[Backup/Recovery]
       recovery interval in minutes = 32767
       print recovery information = DEFAULT
       tape retention in days = DEFAULT

[Cache Manager]
       number of oam trips = DEFAULT
       number of index trips = DEFAULT
       memory alignment boundary = DEFAULT
       global async prefetch limit = 0
       global cache partition number = 16

[Named Cache:c_cust_index]
       cache size = 510M
       cache status = mixed cache
       cache status = HK ignore cache
       cache replacement policy = relaxed LRU
replacement
       local cache partition number = 16

[4K I/O Buffer Pool]
       pool size = 510M
       wash size = 80 K
       local async prefetch limit = 0

[Named Cache:c_cust_non_index1]
       cache size = 3202M
       cache status = mixed cache
       cache status = HK ignore cache
```

```
       cache replacement policy = DEFAULT
       local cache partition number = 16

[4K I/O Buffer Pool]
       pool size = 3202M
       wash size = 512 K
       local async prefetch limit = 0

[Named Cache:c_customer]
       cache size = 175M
       cache status = mixed cache
       cache replacement policy = DEFAULT
       local cache partition number = 8

[4K I/O Buffer Pool]
       pool size = 175M
       wash size = 3M
       local async prefetch limit = 0

[Named Cache:c_log]
       cache size = 22M
       cache status = log only
       cache replacement policy = DEFAULT
       local cache partition number = 1

[8K I/O Buffer Pool]
       pool size = 20M
       wash size = 8192 K
       local async prefetch limit = 0

[Named Cache:c_no]
       cache size = 1000M
       cache status = mixed cache
       cache replacement policy = relaxed LRU
replacement
       local cache partition number = 4

[4K I/O Buffer Pool]
       pool size = 1000M
       wash size = 2M
       local async prefetch limit = 0

[Named Cache:c_no_order_index]
       cache size = 70M
       cache status = mixed cache
       cache status = HK ignore cache
       cache replacement policy = relaxed LRU
replacement
       local cache partition number = 4

[4K I/O Buffer Pool]
       pool size = 70M
       wash size = 70 K
       local async prefetch limit = 0

[Named Cache:c_ol]
       cache size = 1850M
       cache status = mixed cache
       cache replacement policy = relaxed LRU
replacement
       local cache partition number = 16

[4K I/O Buffer Pool]
       pool size = 1850M
       wash size = 2M
       local async prefetch limit = 0

[Named Cache:c_ol_index]
       cache size = 800M
       cache status = mixed cache
       cache status = HK ignore cache
       cache replacement policy = relaxed LRU
replacement
       local cache partition number = 16

[4K I/O Buffer Pool]
       pool size = 800M
       wash size = 1M
       local async prefetch limit = 0

[Named Cache:c_orders]
       cache size = 2400M
       cache status = mixed cache
       cache replacement policy = DEFAULT
       local cache partition number = 16

[4K I/O Buffer Pool]
       pool size = 1450M
       wash size = 8M
       local async prefetch limit = 0

[16K I/O Buffer Pool]
       pool size = 950M
       wash size = 4M
       local async prefetch limit = 0

[Named Cache:c_stock]
       cache size = 47300M
       cache status = mixed cache
       cache replacement policy = DEFAULT
       local cache partition number = 16

[4K I/O Buffer Pool]
       pool size = 47300M
       wash size = 36M
       local async prefetch limit = 0

[Named Cache:c_stock_index]
       cache size = 810M
       cache status = mixed cache
```

```
       cache status = HK ignore cache
       cache replacement policy = relaxed LRU
replacement
       local cache partition number = 16

[4K I/O Buffer Pool]
       pool size = 810M
       wash size = 160 K
       local async prefetch limit = 0

[Named Cache:c_wid]
       cache size = 100M
       cache status = mixed cache
       cache status = HK ignore cache
       cache replacement policy = relaxed LRU
replacement
       local cache partition number = 16

[4K I/O Buffer Pool]
       pool size = 100M
       wash size = 256 K
       local async prefetch limit = 0

[Named Cache:default data cache]
       cache size = 50M
       cache status = default data cache
       cache replacement policy = relaxed LRU
replacement
       local cache partition number = 8

[4K I/O Buffer Pool]
       pool size = 50M
       wash size = 512 K
       local async prefetch limit = 0

[Meta-Data Caches]
       number of open databases = 5
       number of open objects = DEFAULT
       open object spinlock ratio = DEFAULT
       number of open indexes = DEFAULT
       open index hash spinlock ratio = DEFAULT
       open index spinlock ratio = DEFAULT
       partition groups = DEFAULT
       partition spinlock ratio = DEFAULT

[Disk I/O]
       disk i/o structures = 16384
       number of large i/o buffers = DEFAULT
       page utilization percent = 60
       number of devices = 255
       disable disk mirroring = 1
       allow sql server async i/o = DEFAULT

[Languages]
       disable character set conversions =
DEFAULT

[Unicode]
       enable unicode normalization = DEFAULT
       enable surrogate processing = DEFAULT
       enable unicode conversions = DEFAULT
       size of unilib cache = DEFAULT

[Network Communication]
       default network packet size = 4096
       max network packet size = 4096
       remote server pre-read packets =
DEFAULT
       number of remote connections = 700
       number of remote logins = DEFAULT
       number of remote sites = DEFAULT
       max number network listeners = DEFAULT
       tcp no delay = DEFAULT
       allow sendmsg = DEFAULT
       syb_sendmsg port number = DEFAULT
       allow remote access = DEFAULT

[O/S Resources]
       max async i/os per engine = 4096
       max async i/os per server = 8192

[Parallel Query]
       number of worker processes = DEFAULT
       memory per worker process = DEFAULT
       max parallel degree = DEFAULT
       max scan parallel degree = DEFAULT

[Physical Resources]

[Physical Memory]
       max memory = 31760384
       additional network memory = 4915200
       shared memory starting address =
DEFAULT
       allocate max shared memory = 1
       dynamic allocation on demand = 0
       lock shared memory = DEFAULT
       heap memory per user = 16384

[Processors]
       max online engines = 16
       number of engines at startup = 16

[SQL Server Administration]
       procedure cache size = 128000
       default database size = DEFAULT
       identity burning set factor = DEFAULT
       allow nested triggers = DEFAULT
       allow updates to system tables = DEFAULT
       default fill factor percent = DEFAULT
```

```
       default exp_row_size percent = DEFAULT
       number of mailboxes = DEFAULT
       number of messages = DEFAULT
       number of alarms = DEFAULT
       number of pre-allocated extents =
DEFAULT
       event buffers per engine = DEFAULT
       cpu accounting flush interval =
2147483647
       i/o accounting flush interval =
2147483647
       sql server clock tick length = DEFAULT
       runnable process search count = DEFAULT
       i/o polling process count = DEFAULT
       time slice = DEFAULT
       cpu grace time = DEFAULT
       number of sort buffers = DEFAULT
       size of auto identity column = DEFAULT
       identity grab size = DEFAULT
       housekeeper free write percent = 0
       enable housekeeper GC = 0
       allow resource limits = DEFAULT
       number of aux scan descriptors = DEFAULT
       SQL Perfmon Integration = DEFAULT
       allow backward scans = DEFAULT
       license information = DEFAULT
       enable sort-merge join and JTC = DEFAULT
       abstract plan load = DEFAULT
       abstract plan dump = DEFAULT
       abstract plan replace = DEFAULT
       abstract plan cache = DEFAULT
       text prefetch size = DEFAULT
       enable HA = DEFAULT

[User Environment]
       number of user connections = 700
       stack size = 129024
       stack guard size = DEFAULT
       permission cache entries = DEFAULT
       user log cache size = DEFAULT
       user log cache spinlock ratio = DEFAULT

[Lock Manager]
       number of locks = 20000
       deadlock checking period = 3000
       lock spinlock ratio = 10
       lock address spinlock ratio = 1
       lock table spinlock ratio = 1
       lock hashtable size = DEFAULT
       lock scheme = DEFAULT
       lock wait period = DEFAULT
       read committed with lock = DEFAULT
       print deadlock information = DEFAULT
       deadlock retries = DEFAULT
       page lock promotion HWM = DEFAULT
       page lock promotion LWM = DEFAULT
       page lock promotion PCT = DEFAULT
       row lock promotion HWM = DEFAULT
       row lock promotion LWM = DEFAULT
       row lock promotion PCT = DEFAULT

[Security Related]
       systemwide password expiration =
DEFAULT
       audit queue size = DEFAULT
       curread change w/ open cursors = DEFAULT
       allow procedure grouping = DEFAULT
       select on syscomments.text = DEFAULT
       auditing = DEFAULT
       current audit table = DEFAULT
       suspend audit when device full = DEFAULT
       enable row level access = DEFAULT
       check password for digit = DEFAULT
       minimum password length = DEFAULT
       maximum failed logins = DEFAULT
       enable ssl = DEFAULT
       unified login required = DEFAULT
       use security services = DEFAULT
       msg confidentiality reqd = DEFAULT
       msg integrity reqd = DEFAULT
       secure default login = DEFAULT

[Extended Stored Procedure]
       esp unload dll = DEFAULT
       esp execution priority = DEFAULT
       esp execution stacksize = DEFAULT
       xp_cmdshell context = DEFAULT
       start mail session = DEFAULT

[Error Log]
       event logging = DEFAULT
       log audit logon success = DEFAULT
       log audit logon failure = DEFAULT
       event log computer name = DEFAULT

[Rep Agent Thread Administration]
       enable rep agent threads = DEFAULT

[Component Integration Services]
       enable cis = 0
       cis connect timeout = DEFAULT
       cis bulk insert batch size = DEFAULT
       max cis remote connections = DEFAULT
       cis packet size = DEFAULT
       cis cursor rows = DEFAULT
       enable file access = DEFAULT
       cis bulk insert array size = DEFAULT
       enable full-text search = DEFAULT
       cis rpc handling = DEFAULT
```

```
[Java Services]
       enable java = DEFAULT
       size of process object heap = DEFAULT
       size of shared class heap = DEFAULT
       size of global fixed heap = DEFAULT
       number of java sockets = DEFAULT
       enable enterprise java beans = DEFAULT

[DTM Administration]
       enable DTM = DEFAULT
       enable xact coordination = 0
       xact coordination interval = DEFAULT
       number of dtx participants = DEFAULT
       strict dtm enforcement = DEFAULT
       txn to pss ratio = DEFAULT
       dtm lock timeout period = DEFAULT
       dtm detach timeout period = DEFAULT

[Diagnostics]
       dump on conditions = DEFAULT
       maximum dump conditions = DEFAULT
       number of ccbs = DEFAULT
       caps per ccb = DEFAULT
       average cap size = DEFAULT

[Monitoring]
       enable monitoring = DEFAULT
       sql text pipe active = DEFAULT
       sql text pipe max messages = DEFAULT
       plan text pipe active = DEFAULT
       plan text pipe max messages = DEFAULT
       statement pipe active = DEFAULT
       statement pipe max messages = DEFAULT
       errorlog pipe active = DEFAULT
       errorlog pipe max messages = DEFAULT
       deadlock pipe active = DEFAULT
       deadlock pipe max messages = DEFAULT
       wait event timing = DEFAULT
       process wait events = DEFAULT
       object lockwait timing = DEFAULT
       SQL batch capture = DEFAULT
       statement statistics active = DEFAULT
       per object statistics active = DEFAULT
       max SQL text monitored = DEFAULT
```

sysctl

```
#
# Linux sysctl parameters
#
sunrpc.nlm_debug = 0
sunrpc.nfsd_debug = 0
sunrpc.nfs_debug = 0
sunrpc.rpc_debug = 0
dev.cdrom.check_media = 0
dev.cdrom.lock = 1
dev.cdrom.debug = 0
dev.cdrom.autoeject = 0
dev.cdrom.autoclose = 1
dev.cdrom.info = CD-ROM information, Id:
cdrom.c 3.12 2000/10/18
dev.cdrom.info =
dev.cdrom.info = drive name:hda
dev.cdrom.info = drive speed:24
dev.cdrom.info = drive # of slots:1
dev.cdrom.info = Can close tray:1
dev.cdrom.info = Can open tray:1
dev.cdrom.info = Can lock tray:1
dev.cdrom.info = Can change speed:1
dev.cdrom.info = Can select disk:0
dev.cdrom.info = Can read multisession:1
dev.cdrom.info = Can read MCN:1
dev.cdrom.info = Reports media changed:1
dev.cdrom.info = Can play audio:1
dev.cdrom.info = Can write CD-R:0
dev.cdrom.info = Can write CD-RW:0
dev.cdrom.info = Can read DVD:0
dev.cdrom.info = Can write DVD-R:0
dev.cdrom.info = Can write DVD-RAM:0
dev.cdrom.info =
dev.cdrom.info =
fs.binfmt_misc.status = enabled
fs.lease-break-time = 45
fs.dir-notify-enable = 1
fs.leases-enable = 1
fs.overflowgid = 65534
fs.overflowuid = 65534
fs.dentry-state = 149812214500   0
fs.dquot-max = 0
fs.dquot-nr = 0825
fs.super-max = 256
fs.super-nr = 12
fs.file-max = 1048576
fs.file-nr = 10613104191048576
fs.inode-state = 825800    0      0      0
fs.inode-nr = 8258
net.ipv4.conf.eth1.arp_filter = 0
net.ipv4.conf.eth1.tag = 0
net.ipv4.conf.eth1.log_martians = 0
net.ipv4.conf.eth1.bootp_relay = 0
net.ipv4.conf.eth1.proxy_arp = 0
net.ipv4.conf.eth1.accept_source_route = 1
net.ipv4.conf.eth1.send_redirects = 1
net.ipv4.conf.eth1.rp_filter = 0
net.ipv4.conf.eth1.shared_media = 1
net.ipv4.conf.eth1.secure_redirects = 1
net.ipv4.conf.eth1.accept_redirects = 1
net.ipv4.conf.eth1.mc_forwarding = 0
net.ipv4.conf.eth1.forwarding = 0
```

```
net.ipv4.conf.eth0.arp_filter = 0
net.ipv4.conf.eth0.tag = 0
net.ipv4.conf.eth0.log_martians = 0
net.ipv4.conf.eth0.bootp_relay = 0
net.ipv4.conf.eth0.proxy_arp = 0
net.ipv4.conf.eth0.accept_source_route = 1
net.ipv4.conf.eth0.send_redirects = 1
net.ipv4.conf.eth0.rp_filter = 0
net.ipv4.conf.eth0.shared_media = 1
net.ipv4.conf.eth0.secure_redirects = 1
net.ipv4.conf.eth0.accept_redirects = 1
net.ipv4.conf.eth0.mc_forwarding = 0
net.ipv4.conf.eth0.forwarding = 0
net.ipv4.conf.lo.arp_filter = 0
net.ipv4.conf.lo.tag = 0
net.ipv4.conf.lo.log_martians = 0
net.ipv4.conf.lo.bootp_relay = 0
net.ipv4.conf.lo.proxy_arp = 0
net.ipv4.conf.lo.accept_source_route = 1
net.ipv4.conf.lo.send_redirects = 1
net.ipv4.conf.lo.rp_filter = 0
net.ipv4.conf.lo.shared_media = 1
net.ipv4.conf.lo.secure_redirects = 1
net.ipv4.conf.lo.accept_redirects = 1
net.ipv4.conf.lo.mc_forwarding = 0
net.ipv4.conf.lo.forwarding = 0
net.ipv4.conf.default.arp_filter = 0
net.ipv4.conf.default.tag = 0
net.ipv4.conf.default.log_martians = 0
net.ipv4.conf.default.bootp_relay = 0
net.ipv4.conf.default.proxy_arp = 0
net.ipv4.conf.default.accept_source_route = 1
net.ipv4.conf.default.send_redirects = 1
net.ipv4.conf.default.rp_filter = 0
net.ipv4.conf.default.shared_media = 1
net.ipv4.conf.default.secure_redirects = 1
net.ipv4.conf.default.accept_redirects = 1
net.ipv4.conf.default.mc_forwarding = 0
net.ipv4.conf.default.forwarding = 0
net.ipv4.conf.all.arp_filter = 0
net.ipv4.conf.all.tag = 0
net.ipv4.conf.all.log_martians = 0
net.ipv4.conf.all.bootp_relay = 0
net.ipv4.conf.all.proxy_arp = 0
net.ipv4.conf.all.accept_source_route = 0
net.ipv4.conf.all.send_redirects = 1
net.ipv4.conf.all.rp_filter = 1
net.ipv4.conf.all.shared_media = 1
net.ipv4.conf.all.secure_redirects = 1
net.ipv4.conf.all.accept_redirects = 1
net.ipv4.conf.all.mc_forwarding = 0
net.ipv4.conf.all.forwarding = 0
net.ipv4.neigh.eth1.locktime = 100
net.ipv4.neigh.eth1.proxy_delay = 80
net.ipv4.neigh.eth1.anycast_delay = 100
net.ipv4.neigh.eth1.proxy_qlen = 64
net.ipv4.neigh.eth1.unres_qlen = 3
net.ipv4.neigh.eth1.gc_stale_time = 60
net.ipv4.neigh.eth1.delay_first_probe_time = 5
net.ipv4.neigh.eth1.base_reachable_time = 30
net.ipv4.neigh.eth1.retrans_time = 100
net.ipv4.neigh.eth1.app_solicit = 0
net.ipv4.neigh.eth1.ucast_solicit = 3
net.ipv4.neigh.eth1.mcast_solicit = 3
net.ipv4.neigh.eth0.locktime = 100
net.ipv4.neigh.eth0.proxy_delay = 80
net.ipv4.neigh.eth0.anycast_delay = 100
net.ipv4.neigh.eth0.proxy_qlen = 64
net.ipv4.neigh.eth0.unres_qlen = 3
net.ipv4.neigh.eth0.gc_stale_time = 60
net.ipv4.neigh.eth0.delay_first_probe_time = 5
net.ipv4.neigh.eth0.base_reachable_time = 30
net.ipv4.neigh.eth0.retrans_time = 100
net.ipv4.neigh.eth0.app_solicit = 0
net.ipv4.neigh.eth0.ucast_solicit = 3
net.ipv4.neigh.eth0.mcast_solicit = 3
net.ipv4.neigh.lo.locktime = 100
net.ipv4.neigh.lo.proxy_delay = 80
net.ipv4.neigh.lo.anycast_delay = 100
net.ipv4.neigh.lo.proxy_qlen = 64
net.ipv4.neigh.lo.unres_qlen = 3
net.ipv4.neigh.lo.gc_stale_time = 60
net.ipv4.neigh.lo.delay_first_probe_time = 5
net.ipv4.neigh.lo.base_reachable_time = 30
net.ipv4.neigh.lo.retrans_time = 100
net.ipv4.neigh.lo.app_solicit = 0
net.ipv4.neigh.lo.ucast_solicit = 3
net.ipv4.neigh.lo.mcast_solicit = 3
net.ipv4.neigh.default.gc_thresh3 = 1024
net.ipv4.neigh.default.gc_thresh2 = 512
net.ipv4.neigh.default.gc_thresh1 = 128
net.ipv4.neigh.default.gc_interval = 30
net.ipv4.neigh.default.locktime = 100
net.ipv4.neigh.default.proxy_delay = 80
net.ipv4.neigh.default.anycast_delay = 100
net.ipv4.neigh.default.proxy_qlen = 64
net.ipv4.neigh.default.unres_qlen = 3
net.ipv4.neigh.default.gc_stale_time = 60
net.ipv4.neigh.default.delay_first_probe_time
= 5
net.ipv4.neigh.default.base_reachable_time =
30
net.ipv4.neigh.default.retrans_time = 100
net.ipv4.neigh.default.app_solicit = 0
net.ipv4.neigh.default.ucast_solicit = 3
net.ipv4.neigh.default.mcast_solicit = 3
net.ipv4.tcp_adv_win_scale = 2
net.ipv4.tcp_app_win = 31
net.ipv4.tcp_rmem = 409687380174760
net.ipv4.tcp_wmem = 409616384131072
```

```
net.ipv4.tcp_mem = 195584196096196608
net.ipv4.tcp_dsack = 1
net.ipv4.tcp_reordering = 3
net.ipv4.tcp_fack = 1
net.ipv4.tcp_orphan_retries = 0
net.ipv4.inet_peer_gc_maxtime = 120
net.ipv4.inet_peer_gc_mintime = 10
net.ipv4.inet_peer_maxttl = 600
net.ipv4.inet_peer_minttl = 120
net.ipv4.inet_peer_threshold = 4104
net.ipv4.igmp_max_memberships = 20
net.ipv4.route.min_adv_mss = 256
net.ipv4.route.min_pmtu = 552
net.ipv4.route.mtu_expires = 600
net.ipv4.route.gc_elasticity = 8
net.ipv4.route.error_burst = 500
net.ipv4.route.error_cost = 100
net.ipv4.route.redirect_silence = 2048
net.ipv4.route.redirect_number = 9
net.ipv4.route.redirect_load = 2
net.ipv4.route.gc_interval = 60
net.ipv4.route.gc_timeout = 300
net.ipv4.route.gc_min_interval = 5
net.ipv4.route.max_size = 131072
net.ipv4.route.gc_thresh = 8192
net.ipv4.route.max_delay = 10
net.ipv4.route.min_delay = 2
net.ipv4.icmp_echoreply_rate = 0
net.ipv4.icmp_paramprob_rate = 100
net.ipv4.icmp_timeexceed_rate = 100
net.ipv4.icmp_destunreach_rate = 100
net.ipv4.icmp_ignore_bogus_error_responses = 0
net.ipv4.icmp_echo_ignore_broadcasts = 0
net.ipv4.icmp_echo_ignore_all = 0
net.ipv4.ip_local_port_range = 3276861000
net.ipv4.tcp_max_syn_backlog = 1024
net.ipv4.tcp_rfc1337 = 0
net.ipv4.tcp_stdurg = 0
net.ipv4.tcp_abort_on_overflow = 0
net.ipv4.tcp_tw_recycle = 0
net.ipv4.tcp_syncookies = 0
net.ipv4.tcp_fin_timeout = 60
net.ipv4.tcp_retries2 = 15
net.ipv4.tcp_retries1 = 3
net.ipv4.tcp_keepalive_intvl = 75
net.ipv4.tcp_keepalive_probes = 9
net.ipv4.tcp_keepalive_time = 7200
net.ipv4.ipfrag_time = 30
net.ipv4.ip_dynaddr = 0
net.ipv4.ipfrag_low_thresh = 196608
net.ipv4.ipfrag_high_thresh = 262144
net.ipv4.tcp_max_tw_buckets = 180000
net.ipv4.tcp_max_orphans = 32768
net.ipv4.tcp_synack_retries = 5
net.ipv4.tcp_syn_retries = 5
net.ipv4.ip_nonlocal_bind = 0
net.ipv4.ip_no_pmtu_disc = 0
net.ipv4.ip_autoconfig = 0
net.ipv4.ip_default_ttl = 64
net.ipv4.ip_forward = 0
net.ipv4.tcp_retrans_collapse = 1
net.ipv4.tcp_sack = 1
net.ipv4.tcp_window_scaling = 1
net.ipv4.tcp_timestamps = 1
net.core.hot_list_length = 128
net.core.optmem_max = 10240
net.core.message_burst = 50
net.core.message_cost = 5
net.core.mod_cong = 290
net.core.lo_cong = 100
net.core.no_cong = 20
net.core.no_cong_thresh = 20
net.core.netdev_max_backlog = 300
net.core.rmem_default = 65535
net.core.wmem_default = 65535
net.core.rmem_max = 131071
net.core.wmem_max = 131071
vm.page-cluster = 4
vm.pagetable_cache = 2550
vm.kswapd = 512328
vm.pagecache = 21575
vm.buffermem = 21060
vm.overcommit_memory = 0
vm.bdflush = 306464 256    500    3000   6000
vm.freepages = 63812761914
kernel.overflowgid = 65534
kernel.overflowuid = 65534
kernel.random.uuid = 15ce8c38-5281-42ce-8493-
ce0f740944b3
kernel.random.boot_id = 4e1fbebb-01c4-4146-
b17a-5782f963679c
kernel.random.write_wakeup_threshold = 128
kernel.random.read_wakeup_threshold = 8
kernel.random.entropy_avail = 0
kernel.random.poolsize = 512
kernel.threads-max = 65532
kernel.sem = 1200039321600010032768
kernel.msgmnb = 4194304
kernel.msgmni = 1048576
kernel.msgmax = 8192
kernel.shmmni = 4096
kernel.shmall = 2097152
kernel.shmmax = 33554432
kernel.rtsig-max = 1024
kernel.rtsig-nr = 0
kernel.hotplug = /sbin/hotplug
kernel.modprobe = /sbin/modprobe
kernel.printk = 641 7
kernel.ctrl-alt-del = 0
kernel.real-root-dev = 2056
```

```
kernel.panic = 0
kernel.domainname =
kernel.hostname = tpcccl1
kernel.version = #12 SMP Thu Aug 23 14:08:47
EDT 2001
kernel.osrelease = 2.4.5
kernel.ostype = Linux
```

system

```
*Ident "@(#)system1.1897/06/27 SMI" /* SVR4
1.5 */
*
* SYSTEM SPECIFICATION FILE
*

* moddir:
*
*      Set the search path for modules.  This
has a format similar to the
*      csh path variable. If the module isn't
found in the first directory
*      it tries the second and so on. The
default is /kernel /usr/kernel
*
*      Example:
*             moddir: /kernel /usr/kernel
/other/modules



* root device and root filesystem configura-
tion:
*
*      The following may be used to override
the defaults provided by
*      the boot program:
*
*      rootfs:      Set the filesystem type of
the root.
*
*      rootdev:Set the root device.  This
should be a fully
*                   expanded physical path-
name.  The default is the
*                   physical pathname of the
device where the boot
*                   program resides.  The
physical pathname is
*                   highly platform and con-
figuration dependent.
*
*      Example:
*             rootfs:ufs
*             root-
dev:/sbus@1,f8000000/esp@0,800000/sd@3,0:a
*
*      (Swap device configuration should be
specified in /etc/vfstab.)



* exclude:
*
*      Modules appearing in the moddir path
which are NOT to be loaded,
*      even if referenced. Note that 'exclude'
accepts either a module name,
*      or a filename which includes the direc-
tory.
*
*      Examples:
*             exclude: win
*             exclude: sys/shmsys



* forceload:
*
*      Cause these modules to be loaded at boot
time, (just before mounting
*      the root filesystem) rather than at
first reference. Note that
*      forceload expects a filename which
includes the directory. Also
*      note that loading a module does not nec-
essarily imply that it will
*      be installed.
*
*      Example:
*             forceload: drv/foo
 forceload: drv/orhw
 forceload: drv/orvi



* set:
*
*      Set an integer variable in the kernel or
a module to a new value.
*      This facility should be used with cau-
tion.  See system(4).
*
*      Examples:
*
*      To set variables in 'unix':
*
*             set nautopush=32
*             set maxusers=40
```

```
*
*      To set a variable named ‘debug’ in the
module named ‘test_module’
*
*            set test_module:debug = 0x13

* Begin FJSVssf (do not edit)
* set ftrace_atboot = 1
* set kmem_flags = 0x100
* set kmem_lite_maxalign = 8192
* End FJSVssf (do not edit)
forceload:drv/FJSVpanel
* Semaphores
*-------------------
* set shmsys:shminfo_shmmax=33285996544
# set shmsys:shminfo_shmmax=68719476736
set shmsys:shminfo_shmmax=0xffffffffffffffff
set shmsys:shminfo_shmmin=1
set shmsys:shminfo_shmmni=900
set shmsys:shminfo_shmseg=200
set semsys:seminfo_semmap=100
set semsys:seminfo_semmni=8000
set semsys:seminfo_semmns=8000
set semsys:seminfo_semmnu=8000
set semsys:seminfo_semume=100
set semsys:seminfo_semmsl=512

*--------------------
* Message Queue
*--------------------
set msgsys:msginfo_msgmax=1048576
set msgsys:msginfo_msgmnb=4194304
set msgsys:msginfo_msgmni=4400
set msgsys:msginfo_msgssz=128
set msgsys:msginfo_msgtql=32768
set msgsys:msginfo_msgseg=32767
set msgsys:msginfo_msgmap=3002

* settings other
set maxphys=4194304
* set pt_cnt=4096
* set priority_paging=1
set autoup=900
set bufhwm=8000
set rlim_fd_max=10240
* get more memory with Sybase (ludger)
set segspt_minfree=4000
set swapfs_minfree=256
set pages_pp_maximum=256
set rlim_fd_cur=256
set rechoose_interval=300
set disable_memscrub=1

set lotsfree = 8192
set desfree = 4096
set minfree = 2048


* Increases the size of STREAMS synchroniza-
tion
set sq_max_size = 1600
set nstrpush = 90
set ncsize = 2000
set maxusers = 128
set nfs:nfs3_max_threads = 48
set nfs:nfs3_nra = 10
* set maxphys=524288
* set priority_paging=1
*
set dosynctodr=0
```

tpcc.params

```
#
# TPCC Parameters for multi-threaded client
run
# Parameters for controlling a tpcc run.
# Used by scripts that do the runs.
#

# Run parameters (seconds)
TPCC_WAITTIME=5     # from create of master
file to start of rampup
TPCC_RAMPUP=3600
TPCC_DURATION=7200# measurement interval
TPCC_RAMPDOWN=30
# Run parameter (minutes)
TPCC_CHECKPOINT_INTERVAL=30

#
# User control.
#
TPCC_USER_COUNT=7500
TPCC_THREADS_PER_PROC=1000
TPCC_THREADS_PER_CTX=50

# This wants to be at least TPCC_USER_COUNT /
TPCC_RAMPUP
TPCC_RAMPUP_USERS_PER_SEC=7
export TPCC_RAMPUP_USERS_PER_SEC

#
# Concurrent logins.
#
TPCC_PARALLEL_LOGINS=3
export TPCC_PARALLEL_LOGINS

#
# NURAND parameter
```

```
#
TPCC_NURAND_C1=208

export TPCC_WAITTIME TPCC_RAMPUP TPCC_DURATION
TPCC_RAMPDOWN
export TPCC_USER_COUNT TPCC_THREADS_PER_PROC
TPCC_THREADS_PER_CTX
export TPCC_NURAND_C1

# Stats
TPCC_COLLECT_STATS=false
TPCC_TOP_INTERVAL=10# interval for linux "top"
command
TPCC_SAR_INTERVAL=10# interval for sar on DB
server
TPCC_VMSTAT_INTERVAL=10# interval for vmstat
on all machines
TPCC_IOSTAT_INTERVAL=10
TPCC_SYSSTAT_INTERVAL=10
export TPCC_COLLECT_STATS TPCC_TOP_INTERVAL
TPCC_SAR_INTERVAL
export TPCC_VMSTAT_INTERVAL
TPCC_IOSTAT_INTERVAL TPCC_SYSSTAT_INTERVAL

#
# The Database Server
#
SERVER=sarak
export SERVER

# The master RTE system:
MASTER=tpccdr1; export MASTER

RTES="$MASTER tpccdr2 tpccdr3 tpccdr4 tpccdr5
tpccdr6 tpccdr7 tpccdr8"
RTES="$RTES tpccdr9 tpccdr10"
RTES="$RTES tpcccl1 tpcccl2"
export RTES

CLIENTS="tpcccl4 tpcccl5 tpcccl6 tpcccl7
tpcccl8 tpcccl9"

export CLIENTS

# Entire list of machines
HOSTS="$RTES $CLIENTS $SERVER"

# Filers
TPCC_FILERS="wynn laforge decker quark bashir
betor keiko pike"
TPCC_LOG_FILERS="pike"
export TPCC_FILERS TPCC_LOG_FILERS

TPCC_SYNCPF=7
export TPCC_SYNCPF

# Login in
TPCC_RLOGINNAME=tpcc
TPCC_PASSWD=tpcc
export TPCC_RLOGINNAME TPCC_PASSWD

# Max warehouses and districts
#TPCC_WH=100# this is calculated later on.
TPCC_DT=10# DO NOT CHANGE THIS
export TPCC_WH TPCC_DT

#
# These delays are used by the RTE.  They
should all be zero.
#
TPCC_ME_DELAY=0
TPCC_NO_DELAY=0
TPCC_PY_DELAY=0
TPCC_OS_DELAY=0
TPCC_DL_DELAY=0
TPCC_SL_DELAY=0
export TPCC_ME_DELAY TPCC_NO_DELAY
TPCC_PY_DELAY
export TPCC_OS_DELAY TPCC_DL_DELAY
TPCC_SL_DELAY

L_rte_count=0
L_incr='expr $TPCC_USER_COUNT / $TPCC_DT'
for i in $RTES
do
       L_rte_count='expr $L_rte_count + 1'
done
# TPCC_WH is needed by the RTEs
TPCC_WH='expr $L_rte_count \* $L_incr'

# Tell each RTE what warehouse range to use.
L_BW=1
TPCC_BWH_tpccdr1=$L_BW# base warehouse
L_BW='expr $L_BW + $L_incr'
TPCC_BWH_tpccdr2=$L_BW# base warehouse
L_BW='expr $L_BW + $L_incr'
TPCC_BWH_tpccdr3=$L_BW# base warehouse
L_BW='expr $L_BW + $L_incr'
TPCC_BWH_tpccdr4=$L_BW# base warehouse
L_BW='expr $L_BW + $L_incr'
TPCC_BWH_tpccdr5=$L_BW# base warehouse
L_BW='expr $L_BW + $L_incr'
TPCC_BWH_tpccdr6=$L_BW# base warehouse
L_BW='expr $L_BW + $L_incr'
TPCC_BWH_tpccdr7=$L_BW# base warehouse
L_BW='expr $L_BW + $L_incr'
TPCC_BWH_tpccdr8=$L_BW# base warehouse
L_BW='expr $L_BW + $L_incr'
TPCC_BWH_tpccdr9=$L_BW# base warehouse
L_BW='expr $L_BW + $L_incr'
```

```
TPCC_BWH_tpccdr10=$L_BW# base warehouse
L_BW=‘expr $L_BW + $L_incr‘
TPCC_BWH_tpcccl1=$L_BW# base warehouse
L_BW=‘expr $L_BW + $L_incr‘
TPCC_BWH_tpcccl2=$L_BW# base warehouse

TPCC_PORT=21001     # multi-threaded client
P=”:$TPCC_PORT”
TPCC_HOST_tpccdr1=”tpcccl4-p${P}”
TPCC_HOST_tpccdr2=”tpcccl4-p${P}”
TPCC_HOST_tpccdr3=”tpcccl5-p${P}”
TPCC_HOST_tpccdr4=”tpcccl5-p${P}”
TPCC_HOST_tpccdr5=”tpcccl6-p${P}”
TPCC_HOST_tpccdr6=”tpcccl6-p${P}”
TPCC_HOST_tpccdr7=”tpcccl7-p${P}”
TPCC_HOST_tpccdr8=”tpcccl7-p${P}”
TPCC_HOST_tpccdr9=”tpcccl8-p${P}”
TPCC_HOST_tpccdr10=”tpcccl8-p${P}”
TPCC_HOST_tpcccl1=”tpcccl9-p${P}”
TPCC_HOST_tpcccl2=”tpcccl9-p${P}”
export TPCC_BWH_tpccdr1 TPCC_HOST_tpccdr1
export TPCC_BWH_tpccdr2 TPCC_HOST_tpccdr2
export TPCC_BWH_tpccdr3 TPCC_HOST_tpccdr3
export TPCC_BWH_tpccdr4 TPCC_HOST_tpccdr4
export TPCC_BWH_tpccdr5 TPCC_HOST_tpccdr5
export TPCC_BWH_tpccdr6 TPCC_HOST_tpccdr6
export TPCC_BWH_tpccdr7 TPCC_HOST_tpccdr7
export TPCC_BWH_tpccdr8 TPCC_HOST_tpccdr8
export TPCC_BWH_tpccdr9 TPCC_HOST_tpccdr9
export TPCC_BWH_tpccdr10 TPCC_HOST_tpccdr10
export TPCC_BWH_tpcccl1 TPCC_HOST_tpcccl1
export TPCC_BWH_tpcccl2 TPCC_HOST_tpcccl2

#
# Run special code for ACID tests
# 0 - regular, 1 - ACID tests
TPCC_ACID=0
# Run special code for AUDIT tests
# 0 - regular, 1 - AUDIT tests
TPCC_AUDIT=0
```

Appendix E. 60-Day Space Calculations

60-Day space requirements

Warehouses	10000				TpmC	112,122.95
Table	Rows	Data KB	Index KB	Extra 5% KB	8hr Space	Total Space KB
Warehouse	10,000	40,004	76	2,004		42,084
District	100,000	40,004	84	2,004		42,092
Customer	300,000,000	200,400,004	14,405,424	10,740,271		225,545,699
History	300,000,000	16,804,096	0		2,790,067	16,804,096
New Order	90,000,000	1,440,004	4,296	72,215		1,516,515
Orders	300,000,000	8,589,256	25,568		2,507,348	8,614,824
OrderLine	3,000,000,000	184,693,684	595,796		33,194,184	185,289,480
Item	100,000	9,604	28	482		10,114
Stock	1,000,000,000	308,000,004	839,248	15,441,963		324,281,215
Total		720,016,660	15,870,520	26,258,939	38,491,599	762,146,119
	MB					
Dynamic Space	205,163	Sum of Data for Order, Orderline and History				
Static Space	539,120	Sum of Data+Index+5%-Dynamic Space				
Free Space		Total Allocated Spac - (Dynamic + Static Space)				
Daily Growth	36,806	(Dynamic Space/(W*62.5))*tpmc				
Daily Spread	-	(Free Space -1.5*Daily Growth) Zero Assumed				
60 Day Space MB	2,747,456					
60 Day Space GB	2,683.06	GB				
Log Size	0.00	MB				
KB Per New Order	3.73	KB				
8 hr log MB	196,087	MB				
8 hr log GB	191.4917	GB				
		Disks	Space			
Space Usage	GB Needed	Measured	Available			
60 Day Space DB	2,683.06	405	4,656			
Total DB		405	4656.10			
8-hr log	191.49	11	239			
OS, Swap, and filer overhead		20				
Total Storage	2,874.55	GB	4,895.02			

	The file groups are reported in 4K pages from the sysfile table.					
	Customer	Misc	Stock	Order line	Orders	History
Warehouse		42,084				
District		42,092				
Customer	225,545,699					
History						19,594,163
New Order		1,516,515				
Orders					11,122,172	
OrderLine				218,483,664		
Item		10,114				
Stock			324,281,215			
Total	225,545,699	1,610,805	324,281,215	218,483,664	11,122,172	19,594,163
Total pages	**56,463,360**	**899,072**	**85,585,920**	**54,681,600**	**4,270,080**	**5,550,336**
Total KB	225,853,440	3,596,288	342,343,680	218,726,400	17,080,320	22,201,344
	OK	OK	OK	OK	OK	OK
KB needed	(307,741)	(1,985,483)	(18,062,465)	(242,736)	(5,958,148)	(2,607,181)

Appendix F. Price Quotations

This appendix lists price quotations for the following components included in this FDR:

BEA Tuxedo

February 28, 2002

David Yu
Business Development Manager
Network Appliance
495 East Java Drive
Sunnyvale, CA 94089
(408) 822-6318 (Voice)
(408) 822-4415 (Fax)

Dear Mr. Yu

Per your request I am enclosing the pricing information regarding TUXEDO 7.1 that you requested. This pricing applies to Tuxedo 6.4, 6.5, 7.1, and 8.0. Please note that Tuxedo 8.0 is our most recent version of Tuxedo. Core functionality services pricing is appropriate for your activities. As per the table below systems are classified as either a Tier 1, 2, 3, 4 or 5 systems depending on the performance and CPU capacity of the system. Note: A Dell Power Edge 1550 Server is classified as a Tier 1 server per the table below. This quote is valid for 60 days from the date of this letter.

Tuxedo Core Functionality Services (CFS) Program Product Pricing and Description

TUX-CFS provides a basic level of middleware support for distributed computing, and is b used by organizations with substantial resources and knowledge for advanced distributed computing implementations.

TUX-CFS prices are server only and are based on the overall performance characteristics the server and uses the same five tier computer classification as TUXEDO 6.4, 6.5, 7.1, a 8.0. Prices range from $3,000 for Tier 1 to $250,000 for Tier 5. Under this pricing option EVERY system running TUX-CFS at the user site must have a TUXEDO license installed pay the appropriate per server license fees. Note that a 5% discount will apply to total list license purchases less than $100,000 (for instance 10 Tier 1 servers; PowerEdge 1550 – 3,000 = $30,000 - would be eligible for a 5% discount). Support is not discountable.

Very Truly Yours,

Robert J. Gieringer

Rob Gieringer,
Worldwide Pricing Manager

BEA Tux/CFS Unlimited User License Fees Per Server

Unlimited User License fees per server	Number of Users	Dollar Amount	Maintenance (5 x 8) per year	Maintenance (7 x 24) per year
Tier 1 -- PC Servers with 1 or 2 CPUs, entry level RISC Uni-processor workstations and servers	Unlimited	$3,000.00	$480.00	$690.00
Tier 2 - PC Servers with 3 or 4 CPUs, Midrange RISC Uni-processor servers and workstations with up to 2 CPUs	Unlimited	$12,000.00	$1,920.00	$2,760.00
Tier 3 - Midrange Multiprocessors, up to 8 CPUs per system capacity	Unlimited	$30,000.00	$4,800.00	$6,900.00
Tier 4 - Large (more than 8, less than 32 CPUs)	Unlimited	$100,000.00	$16,000.00	$23,000.00
Tier 5 - Massively Parallel Systems, > 32 processors	Unlimited	$250,000.00	$40,000.00	$57,500.00

Dell PowerEdge 1550 - Tier 1

Cisco Catalyst 3524 switch

Charlie Manca
Regional Manager, Silicon Valley Named Accounts
Cisco Systems, Inc.
408.853.2493 voicemail
800.365.4578 pager
650.743.3493 cell

25 February 2002

Product	Description	Quantity	Price	Lead Time
WS-C3524-XL-EN	Catalyst 3524 XL Enterprise Edition	1	$1,976.70	11
CAB-AC	Power Cord,110V	1	$0.00	
CON-OSP-WS-C3524	24x7x4 Onsite Svc,WS-C3524	1	$1,436.40	
Total Price:			$3,413.10	

Total Lead Time: 11 days

This quote is valid until 3 April 2002.

Dell PowerEdge 1550

DELL

QUOTATION

E Com

QUOTE #: 77353523

Customer #: 5613271

Quote Date: 2/28/02

Date: 2/28/02 5:21:57 PM

Customer Name: NETWORK APPLIANCES

TOTAL QUOTE AMOUNT:	**$27,999.51**
Product Subtotal:	**$25,115.60**
Tax:	**$2,133.91**
Shipping & Handling:	**$750.00**
Shipping Method:	**Ground**

Total Number of System Groups: 1

GROUP:	QUANTITY:	SYSTEM PRICE: $2,511.56	GROUP TOTAL: $25,115.60
1	10		

Base Unit:	**PowerEdge 1550,1GHz/133,256K Cache,P3,Base (220-9993)**
Processor:	**2nd Processor,1GHz,256K,P3 for Dell PowerEdge/PowerApp (311-5512)**
Memory:	**1GB SDRAM,133MHz,2 X 512MB DIMMs (311-0855)**
Keyboard:	**No Keyboard Option (310-3281)**
Monitor:	**No Monitor Option (320-0058)**
Hard Drive:	**18GB 10K RPM Ultra 160 SCSI Hard Drive (340-2468)**
Floppy Disk Drive:	**1.44M,3.5 in,Third Height, Floppy Drive (340-2474)**
Operating System:	**Red Hat Linux 7.2,with Documentation and Media Factory Installed (420-0322)**
Operating System:	**Dell OpenManage Kit,32-Bit (310-1261)**
Mouse:	**Mouse Option None (310-0024)**

NIC:	**Dual On-Board NICS ONLY (430-8991)**
CD-ROM or DVD-ROM Drive:	**24X IDE Internal CD-ROM,Black,for Dell PowerEdge (313-0317)**
Option 1:	**Hard Drive Configuration #1,MSSCSI (340-2470)**
Option 2:	**2 Post Shelf Kit,Rack,for DellPowerEdge/PowerApp (310-0519)**
Service:	**Type 2 Contract - Same Day 4-Hour 7x24 Parts and Labor On-Site Response, Initial Year (900-6350)**
Service:	**Type 2 Contract - Same Day 4-Hour 7x24 Parts & Labor On-Site Response 2YR Extended (900-6352)**
Installation:	**On-Site Installation Declined (900-9997)**

COMMENTS

EXPIRES 4/3

FINANCE APPROVED 2/28

TOTAL QUOTE AMOUNT:	**$27,999.51**
Product Subtotal:	**$25,115.60**
Tax:	**$2,133.91**
Shipping & Handling:	**$750.00**
Shipping Method:	**Ground**

Total Number of System Groups: **1**

SALES REP:	**PATTY ROSADO**	**PHONE:**	**800-274-3355**
Email Address:	**Patty_Rosado@Dell.com**	**Phone Ext:**	**42452**

For your convenience, your sales representative, quote number and customer number have been included to provide you with faster service when you are ready to place your order. For easier ordering, ask your Sales Rep to add this quote to your Premier Page.
This quote is subject to the terms of the agreement signed by you and Dell, or absent such agreement, is subject to the applicable Dell terms and conditions agreement.
Prices and tax rates are valid in the U.S. only and are subject to change.
All product and pricing information is based on latest information available. Subject to change without notice or obligation.
LCD panels in Dell products contain mercury, please dispose of properly.
Please contact Dell Financial Services' Asset Recovery Services group for EPA compliant disposal options at US_DFS_AssetRecovery@dell.com. Minimum quantities may apply.

Cables

QUOTATION

See Reverse for Conditions of Sale

Date 02/28/02
Quote No. 00830
Customer No. 714567

David Yu
Network Appliance Inc

Anixter Inc.
Corporate Headquarters
4711 Golf Road
Skokie, IL 60076

DAVID YU 022802
APRIL 30, 02

PRICE GOOD THRU

Item	Quantity	Anixter Catalog Number and Description	Unit	Unit Price	Extended Price
01	8	N/S (COMPU-LINK FGRJ-4848-06-40) FGRJ-4848-06-40FT. 40FT. MTRJ TO MTRJ MULTIMODE DUPLEX BLUE FIBER JU	EA	44.25	$354.00
02	11	N/S (COMPU-LINK T3A3ABQ506BA026) T3A3A-BQ506-BA-02-6FT 6 FT. RJ45 TO RJ45 BLUE W/ BLACK BOOTS	EA	3.25	$35.75

Quote Total: $389.75

TERMS NET30, subject to credit approval.
F.O.B. SHIP.PT., PPD/CHARGE
SHIPMENT
NOTE NOTE
by Anixter.

Prices will be those in effect at time of shipment unless otherwise stated.

Please refer all inquiries to:
LYNN SCHMIDT
Anixter Inc.
Structured Cabling Systems
5720 Stoneridge Drive Ste 2
Pleasanton CA 94588
Phone: 925/469-8500
Fax: 925/469-8910

UPS

Alphanumeric SYSTEMS · INC

Network Appliance

Kimberly C. Munn
Alphanumeric Systems, Inc.
Phone: 919-376-4540
Fax: 919-872-1440
kmunn@alphanumeric.com

2/28/02

Item	Part	Description	Quantity	Unit Price	Extended Price
1	SU1400RM2U	SMARTUPS 1400VA RM 2U LINEINT UPS	1	$ 705.00	$705.00

Subtotal	$705.00
Tax 6.5%	$ 46.80
Shipping	$ 15.00
Quote Total	$ 766.80

Purchase Orders should be made to:
Alphanumeric Systems, Inc.
3801 Wake Forest Road
Raleigh, NC 27609
Attention: Kimberly C. Munn, Inside Sales Representative

Accepted _____

Date _____

This quote will be valid to April 5, 2002.

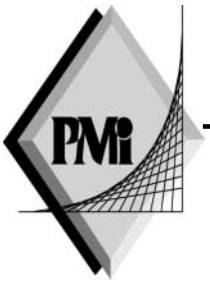
Alphanumeric Hours of Operation and Billing

Standard Operating Hours Monday – Friday 8:30 AM – 5:30 PM., Excluding Holidays.
After Hours/Rates Monday – Friday 5:30 PM – 8:30 AM (Billed at 150% of Standard Rate)
Weekend Hours/Rates 5:30 PM Friday – 8:30 AM Monday (Billed at 200% of Standard Rate, 2 Hr. minimum)
Holiday Rates ASI Schedule (Billed at 300% of Standard Rate, 2 Hr. minimum)
Travel Time (Billed at half time)

Terms

1 ASI will invoice for Hardware and Software upon delivery
2 ASI will invoice for Installation Services as performed.
3 ASI will invoice for Training, Support Hours, and Annual Maintenance in advance.
4 Terms for all invoices are Net 30 days.

Appendix G. Attestation Letter



PERFORMANCE METRICS INC.
TPC Certified Auditors

March 1, 2002

Mr. Stephen Daniel
Database Performance Architect
Network Appliance
627 Davis Dr. Suite 200
Durham, NC 27713

I have verified on site and by remote the TPC Benchmark™ C client/server for the following configuration:

Platform: Fujitsu PRIMEPOWER 850
Database Manager: Sybase ASE 12.5.0.1
Operating System: Solaris 8
Transaction Monitor: BEA Tuxedo 7.1

Servers: Fujitsu PRIMEPOWER 850 with:				
CPU's	Memory	Disks (total)	90% Response	TpmC
16 SPARC64 @675Mhz	Main: 64 GB Cache: 128 KB	2 @ 36GB internal 420 @ 18GB 14 @ 36 GB	0.55	112,286.46
10 Clients: Dell PowerEdge 1550 each with:				
2 Pentium III @1 Ghz	Main: 1 GB Cache: 256K	1 @ 18GB	Na	Na

In my opinion, these performance results were produced in compliance with the TPC requirements for the benchmark. The following attributes of the benchmark were given special attention:

- The transactions were correctly implemented.
- The database files were properly sized and populated.

2229 Benita Dr. Suite 101, Rancho Cordova, CA 95670 Page 1
(916) 635-2822 fax: (916) 858-0109 email: Lorna@PerfMetrics.com

- The database was properly scaled with 10,000 warehouses of which only 9,000 were active during the measured interval.
- The ACID properties were successfully demonstrated.
- The ACID properties for data and log loss were demonstrated on a subset of the SUT configured with a database properly populated for 1000 warehouses and using 10,000 users to drive the load.
- Input data was generated according to the specified percentages.
- Eight hours of mirrored log space was present on the tested system.
- Eight hours of growth space for the dynamic tables was present on the tested system.
- The data for the 60 day space calculation was verified.
- The controller cache was disabled on the log disk controllers.
- The steady state portion of the test was 120 minutes.
- One checkpoint was taken before the measured interval.
- Four checkpoints were taken during the measured interval.
- The system pricing was checked for major components and maintenance.
- Third party quotes were verified for compliance.

Auditor Notes: None.

Sincerely,

Lorna Livingtree
Auditor

2229 Benita Dr. Suite 101, Rancho Cordova, CA 95670
(916) 635-2822 fax: (916) 858-0109 email: Lorna@PerfMetrics.com