

TPC Benchmark™ C
Full Disclosure Report
IBM eServer iSeries 400
Model 840-2420-001



October 24, 2000



Special Notices

The following terms used in this publication are trademarks of **International Business Machines** Corporation in the United States and/or other countries:

- IBM ~
- iSeries 400
- Application System/400
- AS/400
- AS/400e
- DB2 for AS/400
- INT LNG ENV COBOL
- INT LNG ENV C
- IBM
- OS/400
- Structured Query Language/400 (SQL/400)

The following terms used in this publication are trademarks of other companies as follows:

TPC Benchmark Trademark of the Transaction Processing Performance Council
TUXEDO Trademark of BEA Systems, Inc.

Revised July 12, 2000

The information contained in this document is distributed on an AS IS basis without any warranty either expressed or implied. The use of this information or the implementation of any of these techniques is a customer's responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. While each item has been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environment do so at their own risk.

In this document, any references made to an IBM-licensed program are not intended to state or imply that only IBM's licensed program may be used; any functionally equivalent program may be used.

It is possible that this material may contain reference to, or information about, IBM products (machines and programs), programming, or services that are not announced in your country. Such references or information must not be construed to mean that IBM intends to announce such products, programming, or services in your country. All performance data contained in this publication was obtained in a controlled environment, and therefore the results which may be obtained in other operating environments may vary significantly. Users of this document should verify the applicable data in their specific environment.


© **International Business Machines Corporation 2000. All right reserved.**

Note: U.S. Government Users--Documentation related to restricted rights--Use, duplication, or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corporation.

Abstract

This report documents the full disclosure information required by the TPC Benchmark™ C Standard Specification dated October 25, 1999 for measurements on the IBM ~ iSeries 400 Model 840 with Feature Code 2420-001 running at 450 Mhz. The software used on the iSeries 400 Model 840-2420-001 systems includes OS/400 Version 4, Release 5, Modification 0, DB2 for AS/400 Version 4, Release 5, Modification 0, OS/400 Version 4, Release 5, Modification 0, INT LNG ENV COBOL OS/400 V4 R4, INT LNG ENV C OS/400 V4 R4, DB2 Query Manager and SQL Development Kit, BEA TUXEDO 6.4, and Application Development Tools.

IBM® server iSeries Model 840-2420-001

	IBM ^ iSeries Model 840-2420-001	DB2 for AS/400 Version 4 Release 5	Operating System Software OS/400 Version 4 Release 5
Availability Date: December 15, 2000			
Total System Cost	TPC-C Throughput	Price/Performance	
- Hardware - Software -5 Years Maintenance	Sustained maximum throughput of system running TPC-C expressed in transactions per minute	Total system cost/tpmC (\$/tpmC)	
\$9,642,918	163,775.80 tpmC	\$58.88 per tpmC	



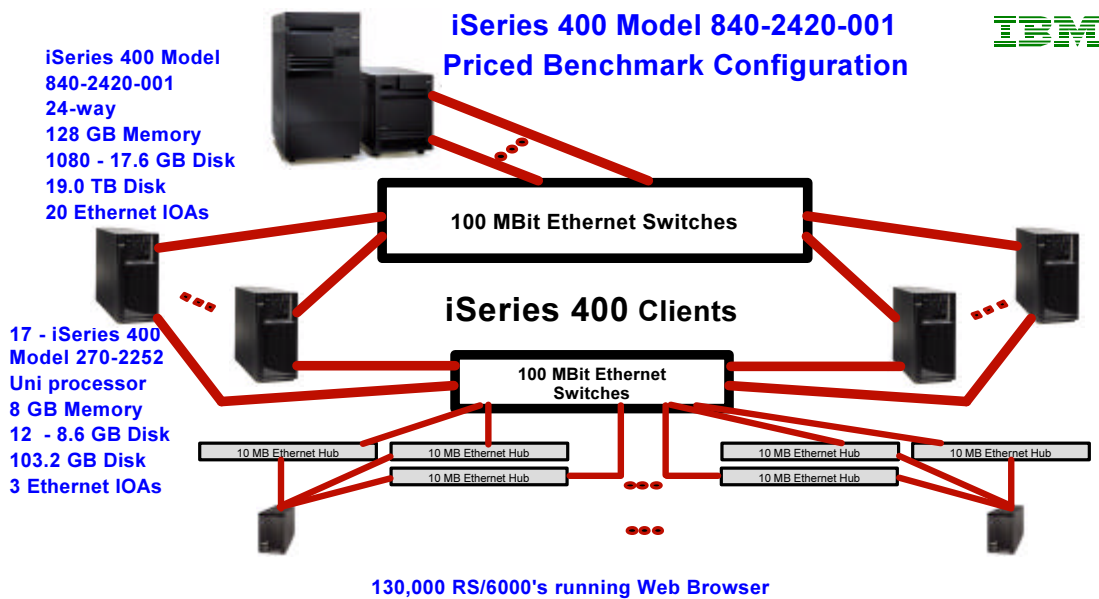
IBM e server iSeries 400 Model 840-2420-001

TPC-C Rev. 3.5

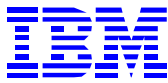
Report Date:
October 24, 2000

Total System Cost	TPC-C Throughput	Price/Performance	Availability Date	
\$9,642,918	163,775.80	\$58.88	December 15, 2000	
Processors	Database Manager	Operating System	Other Software	Number of Users
1 iSeries 400 840-2420-001 (24-way) 17 iSeries 400 9406-270 2252	DB2 for AS/400 Version 4 Release 5	OS/400 Version 4 Release 5	BEA TUXEDO 6.4 INT LNG ENV COBOL OS/400 V4 R4 INT LNG ENV C OS/400 V4 R4	130,000

iSeries 400 Model 840-2420-001 Priced Configuration



System components:	Qty:	Description:
Server	1	
Processors	24	iSeries 400 Model 840-2420-001
Memory		128 GB main memory
Disk controllers	52	RAID Disk Unit Controller (4748)
Disk drives	1080	17.6 GB DASD Total: 13.5 TB
Total storage		
Clients (each):	17	
Processor	1	iSeries 400 Model 270-2252
Memory		8 GB main memory
Disk controller	2	RAID Disk Unit Controller (4748)
Disk drives	12	8.6 GB DASD Total: 51.6 GB



IBM e server iSeries 400 Model 840-2420-001

TPC-C REV 3.5
EXECUTIVE SUMMARY
Report Date: October 24, 2000

iSeries 400 Model 840-2420-001 Configuration

Item Description	Feature Number	Third Party Pricing	Unit Price	Quantity	Extended Price	Extended Maintenance	5-Year Price
Server Hardware:							
iSeries 400 Model 840 24-way Base System Unit	9406-840		640,000	1	640,000	84,816	724,816
RPQ 847108	847108		710,000	1	710,000		710,000
Processor - ISTAR 7S 450 Mhz 24-way, Cache 8x4 MB	2420-001		550,000	1	550,000	92,928	642,928
Interactive Card	1540		0	1	0		0
6m HSL Cable - Base I/O Tower	1461		550	26	14,300		14,300
8192 MB (RIVER - 256 MB technology)	3196		147,456	16	2,359,296		2,359,296
Programmable Regulator (req'd with 3196)	2730		750	2	1,500		1,500
PCI Ultra Magnetic Media Controller	2749		1,300	1	1,300		1,300
PCI IOP with 64 MB memory	2843		1,925	22	42,350		42,350
17.6 GB 10K RPM Disk Unit	4318		2,520	1080	2,721,600		2,721,600
CD-ROM	4425		415	1	415		415
PCI Two Line WAS IOA (ECS) req'd with 5540	4745		425	1	425		425
PCI Twinaxial IOA - Req'd with 5540	4746		750	1	750		750
PCI RAID Disk Unit Controller with 26 MB write cache	4748		6,000	71	426,000		426,000
PCI 100/10 mbps Ethernet IOA	4838		900	20	18,000		18,000
PCI100/16/4 MBPS Token Ring IOA	2744		840	2	1,680		1,680
PCI Expansion Tower / Mantis	5074		17,900	23	411,700	381,984	793,684
PCI Expansion Tower	5101		9,000	24	216,000	476,928	692,928
25 GB 1/4-Inch Tape Drive	4486		6,000	1	6,000		6,000
V.24/EIA232 50ft PCI Cable - System Console	0348		125	1	125		125
Server Subtotal					8,122,141	1,036,656	9,158,097
Client Hardware:							
iSeries 400 Model 270 Base System Unit	9406-270		4,000	1	4,000	4,416	8,416
V.24/EIA 232 20ft PCI Cable for System Console	0348		125	1	125		125
6m HSL Cable - Base I/O Tower	1461		550	2	1,100		1,100
Processor Card	2252		16,000	1	16,000	7,824	23,824
PCI IOP with 32 MB memory	2842		1,800	3	5,400		5,400
Main Store - memory riser card - 16 slots	2884		2,200	1	2,200		2,200
512 MB DIMMS Main Storage	3025		2,048	16	32,768		32,768
8.6 GB 10K RPM Disk Unit	4317		1,400	12	16,800		16,800
CD-ROM	4525		415	1	415		415
4 GB 1/4-Inch Tape Drive	4582		1,300	1	1,300		1,300
PCI Expansion Tower	5075		6,000	1	6,000	5,856	11,856
PCI Two Line WAS IOA (ECS) req'd with 5540	4745		425	1	425		425
PCI Twinaxial IOA - Req'd with 5540	4746		750	1	750		750
PCI RAID Disk Unit Controller with 26 MB write cache	4748		6,000	2	12,000		12,000
PCI 100/10 mbps Ethernet IOA	4838		900	2	1,800		1,800
Single Client Subtotal					101,083	18,096	119,179
Number of Clients				17			
Client Subtotal					1,718,411	307,632	2,026,043
Server Software:							
IBM Operating System/400 V4R5M0	Bundled			1	0	41,164	41,164
DB2 Query Manager Dev. Toolkit	QU1 and ST1		28,800	1	28,800		28,800
Client Software:							
IBM Operating System/400 V4R5M0	Bundled		0	17	0	329,654	329,654
BEA Tuxedo V6.4		1	3,000	17	51,000	40,800	91,800
ILE COBOL	5769CB1		2,400	1	2,400		2,400
Application Development Toolkit	5769PW1		3,020	1	3,020		3,020
DB2 Query Mgr and SQL Development kit	5769ST1		1,600	1	1,600		1,600
ILE C	5769CX2		2,400	1	2,400		2,400
Software Subtotal					89,220	411,618	500,838
User Connectivity:							
12-port 10/100 Ethernet Switch		2	975	21	20,480		20,480
8-Port 10Mb Hub (10% spares)		2	29	23976	684,515		684,515
User Connectivity Subtotal					704,995	0	704,995
5-year System Subtotal					10,634,067	1,755,906	12,389,973
Discounts:							
				<u>%Allowance</u>	<u>Volume</u>	<u>Discount</u>	
Revenue Allowance				22	9,878,072	(2,173,176)	
5-year Term Maintenance Contract Discount				8	1,344,288	(107,543)	
5-year Maintenance Prepay Discount				15.05	1,236,745	(186,130)	
Software Support Discount For Secondary Systems				85	329,654	(280,206)	
					Purchase	Maintenance	Total
5-year System Total					8,460,891	1,182,027	9,642,918
tpmC							163775.80
\$/tpmC							58.88
Note: All pricing is from IBM except items noted in the 3rd Party Pricing column. 1 - BEA Systems, Inc.; 2 - Venture Computer Systems.							
Notes: Server Hardware requires no charge RPQ 847109 Revenue Allowance is applied to hardware and software for the priced configuration. 5-Year Term Maintenance Contract Discount is given when 5-year contract is signed. 5-Year Maintenance Prepay Discount is given when 5-year maintenance costs are prepaid. Software Support Discount For Secondary Systems is given for maintenance costs on Multiple systems in a single I/S shop. Results audited by François Raab of InfoSizing Inc.					Five-Year Cost of Ownership: \$9,642,918		
					tpmC Rating: 163,775.80		
					\$/tpmC: \$58.88		
Prices used in TPC benchmarks reflect the 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.							

Numerical Quantities Summary for IBM ^

iSeries 400 Model 840-2420-001

MQTH, computed Maximum Qualified Throughput 163,775.80
 % throughput difference, reported & repeated 0.99%

Response Times (in seconds)	<u>90%</u>	<u>Avg.</u>	<u>Max.</u>
- New Order	0.5	0.32	5.01
- Payment	0.3	0.20	5.21
- Order-Status	0.5	0.31	4.43
- Delivery (interactive portion)	0.2	0.11	0.21
- Delivery (deferred portion)	2.1	1.70	6.18
- Stock-Level	1.7	0.82	9.22
- Menu	0.1	0.10	0.32
- Response time delayed for emulated components	0.1		

Transaction Mix (in percent of total transactions)	
- New Order	44.88%
- Payment	43.05%
- Order-Status	4.03%
- Delivery	4.03%
- Stock-Level	4.02%

Keying/Think Times (in seconds)	<u>Min.</u>	<u>Avg.</u>	<u>Max.</u>
- New Order	18.00/0.00	18.00/12.02	18.14/120.24
- Payment	3.00/0.00	3.00/12.01	3.12/120.23
- Order-Status	2.00/0.00	2.00/10.01	2.16/100.22
- Delivery	2.00/0.00	2.00/5.04	2.07/50.21
- Stock-Level	2.00/0.00	2.00/5.02	2.06/50.21

Test Duration	
- Ramp-up time	85 minutes
- Measurement interval	20 minutes
- Number of checkpoints	N/A ¹
- Checkpoint interval	N/A ¹
- Number of transactions (all types) completed in Measurement Interval	7,298,931

1. Transparent file synchronization occurred at least five times during the measurement interval.

For additional information on the IBM ~ iSeries 400 see: <http://www.ibm.com/eserver/iseries>

Preface

TPC Benchmark™ C Standard Specification was developed by the Transaction Processing Performance Council (TPC) and released August 13, 1992.

This is the full disclosure report for benchmark testing of the IBM ~ iSeries 400 Model 840-2420-001 system according to the TPC Benchmark™ C Standard Specification. Measurements were done on the IBM ~ iSeries 400 Model 840-2420-001 450 MHz. TPC Benchmark™ C 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 with 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).
- Nonuniform distribution of data access through primary and secondary keys.
- Data bases consisting of many tables with a wide variety of sizes, attributes, and relationships.
- Contention on data access and update.

The 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 work loads or environments. Extrapolations to any other environment are not recommended.

Benchmark results are highly dependent upon work load, specific application requirements, systems 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 subject 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.

Table of Contents

1.0 General Items	14
1.1 Application Code Disclosure	14
1.2 Benchmark Sponsor	14
1.3 Parameter Settings	14
1.4 Configuration Diagrams	14
1.5 IBM ~ iSeries 400 Model 840-2420-001 Benchmark Configuration	15

2.0 Clause 1: Logical Database Design - Related

Items	16
2.1 Table Definitions	16
2.2 Database Organization	16
2.3 Insert and/or Delete Operations	16
2.4 Horizontal or Vertical Partitioning	17

3.0 Clause 2: Transaction and Terminal Profiles -

Related Items	18
3.1 Verification for the Random Number Generator	18
3.2 Input/Output Screens	18
3.3 Terminal Features	19
3.4 Presentation Managers	19
3.5 Home and Remote Order Lines	19
3.6 New-Order Rollback transactions	19
3.7 Number of Items per Order	19
3.8 Home and Remote Payment Transactions	19
3.9 Nonprimary Key Transactions	20
3.10 Skipped Delivery Transactions	20
3.11 Mix of Transaction Types	20
3.12 Queuing Mechanism of Delivery	20
Table 1. Numerical Quantities for Transaction and Terminal Profiles.....	21

4.0 Clause 3: Transaction and System Properties -

Related Items	22
4.1 Atomicity Requirements	22
4.1.1 Atomicity of Completed Transaction	22
4.1.2 Atomicity of Aborted Transactions.....	22
4.2 Consistency Requirements	23
4.2.1 Consistency Condition 1	23

4.2.2 Consistency Condition 2	23
4.2.3 Consistency Condition 3	24
4.2.4 Consistency Condition 4	24
4.2.5 Consistency Condition 5	24
4.2.6 Consistency Condition 6	25
4.2.7 Consistency Condition 7	25
4.2.8 Consistency Condition 8	25
4.2.9 Consistency Condition 9	25
4.2.10 Consistency Condition 10	25
4.2.11 Consistency Condition 11	26
4.2.12 Consistency Condition 12	26
4.2.13 Consistency Tests	26
4.3 Isolation Requirements	26
4.3.1 Isolation Test 1	26
4.3.2 Isolation Test 2	27
4.3.3 Isolation Test 3	27
4.3.4 Isolation Test 4	27
4.3.5 Isolation Test 5	28
4.3.6 Isolation Test 6	28
4.3.7 Isolation Test 7	28
4.3.7.1 Isolation Test 8	29
4.3.7.2 Isolation Test 9	29
4.4 Durability Requirements	30
4.4.1 Permanent Unrecoverable Failure of any Single Durable Medium	30
4.4.1.1 Failure of Durable Medium of Journal Receiver and Instantaneous Interruption and Memory Failure	30
4.4.1.2 Failure of Durable Medium of Database	30
 5.0 Clause 4: Scaling and Database Population -	
Related Items	32
5.1 Cardinality of Tables	32
5.2 Distribution of Tables and Logs	32
5.3 Database Model Implemented	32
5.4 Partitions/Replications Mapping	33
 6.0 Clause 5: Performance Metrics and Response	
Time - Related Items	34
6.1 Response Times	34
6.2 Keying and Think Times	34
6.3 Response Time Frequency Distribution	35
6.4 Performance Curve for Response Time versus Throughput	38

6.5 Think Time Frequency Distribution	38
6.6 Throughput Versus Elapsed Time	39
6.7 Work Performed During Steady State	39
6.8 Reproducibility	40
6.9 Measurement Interval	40

7.0 Clause 6: SUT, Driver, and Communication

Definition-Related Items	41
7.1 RTE Availability	41
7.2 Functionality and Performance of Emulated Components	41
7.3 Network Bandwidth	41
7.4 Operator Intervention	41

8.0 Clause 7: Pricing - Related Items

8.1 Hardware and Programs Used	42
8.2 Five Year Cost of System Configuration	42
8.3 Statement of tpmC and Price/Performance	43
8.3.1 IBM ~ iSeries 400 Model 840-2420-001 Five-Year System Price Configuration	44

9.0 Clause 8: Audit - Related Items

Appendix A. System Parameters and User Profile

A.1 System Parameters	46
A.2 Transaction Subsystem Description	47

Appendix B. Database File Definitions

B.1 AREFFIL: Reference File	48
B.2 CSTMRLFCRT: Customer Logical File	48
B.3 CSTMRLFNAM: Customer Names Logical File	48
B.4 CSTMR: Customer Logical File	48
B.5 CSTMRPF: Customer Physical File	48
B.6 DSTRCTLF: District Logical File	48
B.7 DSTRCT: District Physical File	49
B.8 HSTRY: History Logical File	49
B.9 HSTRYLF: History Logical File	49
B.10 ITEM: Item Physical File	49
B.11 NEWORD: New Order Logical File	49
B.12 NEWORDLF: New Order Logical File	49
B.13 NEWORDPF: New Order Physical File	49
B.14 ORDERS: Orders Logical File	49

B.15 ORDERSLF: Orders Logical File	49
B.16 ORDERSPF: Orders Physical File	49
B.17 ORDLIN: Order Lines Logical File	49
B.18 ORDLINLF: Order Lines Logical File	50
B.19 ORDLINPF: Order Lines Physical File	50
B.20 STOCK: Stock Logical File	50
B.21 STOCKPF: Stock Logical File	50
B.22 WRHSLF: Warehouse Logical File	50
B.23 WRHS: Warehouse Physical File	50

Appendix C. Database Build Programs	51
Program Flow For Build of Server, Client, and Database	51
C.1 CHKIXSPACE.C:	52
C.2 CHKWHSPACE.C:	52
C.3 COMMON.C:	52
C.4 COMMON.H:	53
C.5 COMMONTOOL.H:	53
C.6 COMMONTGCC.C:	53
C.7 COMMONTGCC.H:	55
C.8 CRT1VIEW.CL:	56
C.9 CRT2VIEW.CL:	56
C.10 CRTBLDSPCE.C:	56
C.11 CRTDBDTA.CL:	57
C.12 CRTDBIX.CL:	58
C.13 CRTDIST.CL:	58
C.14 CRTHASH.C:	59
C.15 CRTHISFILE.CL:	59
C.16 CRTHSTRYLF.CL:	59
C.17 CRTINDEX.CL:	59
C.18 CRTITEM.CL:	60
C.19 CRTORDFILE.CL:	60
C.20 CRTTPCCDB.CL:	61
C.21 CRTTPCCJRN.CL:	62
C.22 CRTWHFILE.CL:	63
C.23 CSTMRVIEW.CL:	63
C.24 FILCUSFILE.CL:	63
C.25 FILLCUS.C:	64
C.26 FILLDATA.H:	65
C.27 FILLDIST.C:	66
C.28 FILLHIST.C:	66
C.29 FILLITEM.C:	68
C.30 FILLORD.C:	69

C.31 FILLPARTS.CL:	70
C.32 FILLSTOCK.C:	72
C.33 FILLWH.C:	73
C.34 FILSTKFILE.CL:	74
C.35 MASTCUSSTK.C:	74
C.36 MASTORD.C:	78
C.37 CRTSTOCKLF.CL:	79
C.38 STRTPCCJRN.CL:	79
C.39 TPCCSPACE.C:	80
C.40 TPCCSPACE.H:	81
C.41 TPCCSPACEI.H:	82

Appendix D. Application Source Code	83
Program Flow	83
CLIENT CODE:	84
D.1 COMMON.C:	84
D.2 COMMON.H:	84
D.3 DELVRYSRV.C:	85
D.4 FORMCHILDS.C:	86
D.5 FORMPARENT.C:	86
D.6 FORMSERVER.C:	87
D.7 GETDTTMSTR.C:	91
D.8 IMPORT.H:	92
D.9 NEWORDSRV.C:	93
D.10 ORDSTSSRV.C:	93
D.11 PAYMNTSRV.C:	93
D.12 POSTRESULT.C:	94
D.13 SCANINPUT.C:	96
D.14 STKLVLRSRV.C:	99
D.15 TUXCONFIG File:	99
SERVER CODE:	99
D.16 COMMON.H	99
D.17 DELIVERY.CBL:	100
D.18 DLVRSRVR.C:	101
D.19 GETTIME.C:	102
D.20 NEWORDER.CBL:	102
D.21 NEWSRVR.C:	105
D.22 ORDSRVR.C:	105
D.23 ORDSTS.CBL:	106
D.24 PAYMENT.CBL:	107
D.25 PAYSRVR.C:	110
D.26 STKLVL.CBL:	110

D.27 STKSRVR.C:	111
D.28 TOOMANY.CBL:	112
D.29 TPCCUSER.H:	112
Appendix E. RTE Scripts	113
E.1 RTE Parameters	113
E.2 TRANSACTIONS.H	115
E.3 TRANSACTIONS.C	115
Appendix F. 180-Day DASD Requirements	121
F.1 IBM ~ iSeries 400 Model 840-2420-001 -- 180-Day DASD Requirements	121
F.2 IBM ~ iSeries 400 Model 840-2420-001 -- Journal DASD Requirements	122
Appendix G. Third Party Quotes	123
BEA Tuxedo Quote	123
Venture Computer Systems - Ethernet Switches/Hubs Quote	125
Appendix H. Auditor Letter	126

1.0 General Items

1.1 Application Code Disclosure

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 and output functions.

Appendix D contains the iSeries 400 application code for the five TPC Benchmark™ C transactions and the terminal functions.

1.2 Benchmark Sponsor

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

This benchmark was sponsored by **International Business Machines** Corporation.

1.3 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 tuning options.*
- *Recovery/commit options.*
- *Consistency/locking options.*
- *Operating system and application configuration parameters.*

Appendix A contains the system, database, and application parameters changed from their default values used in these TPC Benchmark™ C tests.

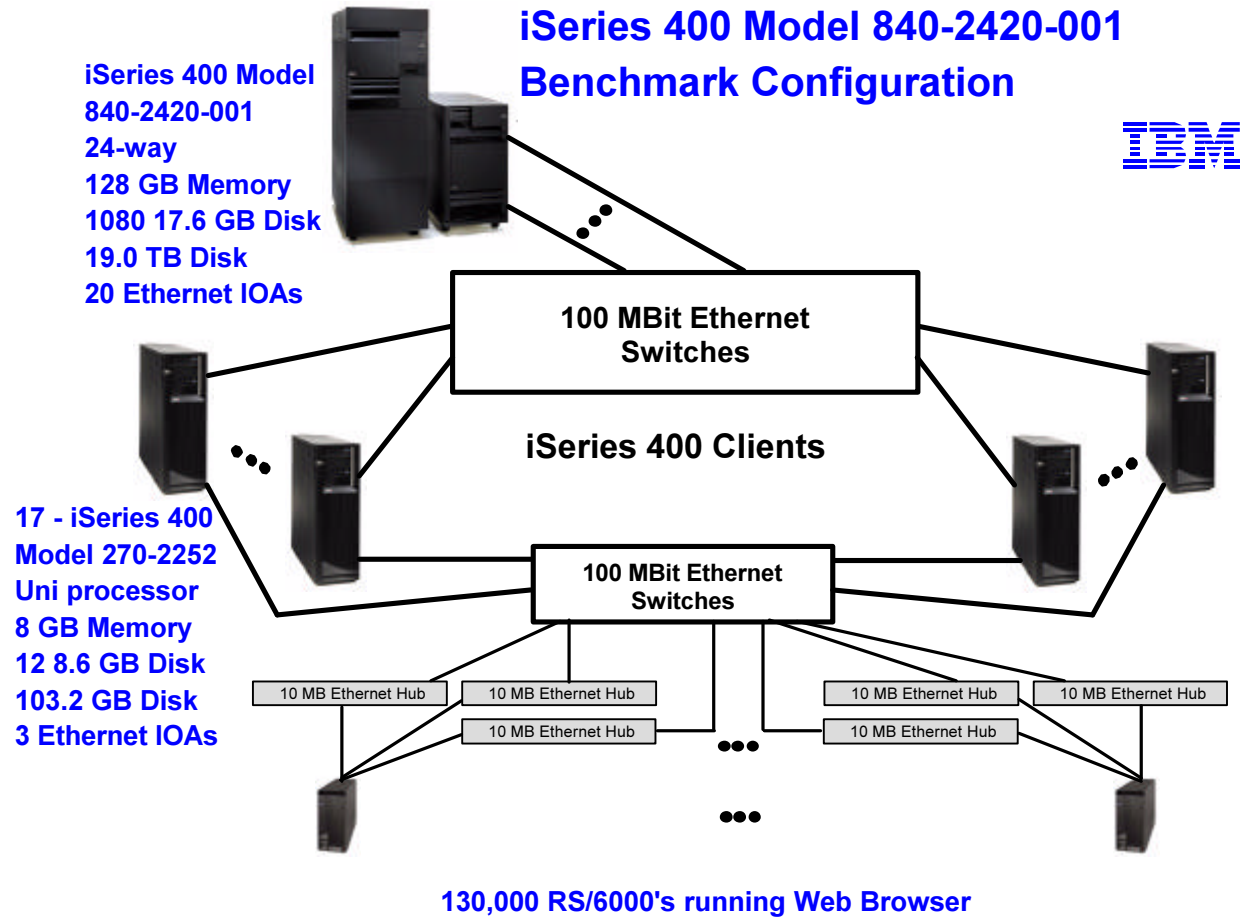
1.4 Configuration Diagrams

Diagrams of both measured and priced configurations must be provided, accompanied by a description of the differences. This includes, but is not limited to:

- *Number and type of processors.*
- *Size of allocated memory, and any specific mapping/partitioning of memory unique to the test.*
- *Number and type of disk units (and controllers if applicable).*
- *Number of channels or bus connections to disk units, including the protocol type.*
- *Number of LAN (e.g., Ethernet) connections, including routers, workstations, terminals, etc., that were physically used in the test or are incorporated into the pricing structure (see Clause 8.1.8).*
- *Type and run-time execution location of software components (e.g., DBMS, client processes, transaction monitors, software drivers, etc.).*

1.5 IBM ^ Configuration

iSeries 400 Model 840-2420-001 Benchmark



2.0 Clause 1: Logical Database Design - Related Items

2.1 Table Definitions

Listing must be provided for all table definition statements and all other statements used to set up the database.

The listings for all file definitions to create the database files are available in Appendix B and the programs used for loading the database (minimal population) are provided in Appendix C.

2.2 Database Organization

The physical organization of table and indices, within the database, must be disclosed.

Physical space for each file (table) is allocated by OS/400 as the file is filled. Although the initial build and the application transactions add records (rows) to several files in the same transaction, each file's extent will reside in separate areas on physical disk. OS/400 will spread the extents for each file across all available disk units to ensure that multiple access requests to the same file may be handled simultaneously. Records are added contiguously within extents, crossing page boundaries where necessary.

Files are created in sequential order according to their primary key.

Indices are generated concurrently with data for Warehouse, District, and Item tables. All other indices are generated after the database is populated. The available space within the index is included in the space reported in this disclosure.

2.3 Insert and/or 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.

During the course of the testing, records were inserted into the ITEM file while users were executing the defined TPC-C transactions.

All of the files used by TPC-C transactions were created with the following attributes:

Authority	*PUBLIC	(Any user can view and modify the files).
ALWUPD	*YES	(Allow update and insert of records).
ALWDLT	*YES	(Allow delete of records).
ALWWRT	*YES	(Allow write of records).

Static files were created with *NOMAX specified on the number of records allowed. This limit is therefore set by the operating system at 2,147,483,648 records. Dynamic files were created with an initial size that is slightly larger than initial database requirement, and extent definitions that would allow expansion, as needed.

2.4 Horizontal or Vertical Partitioning

While there are few restrictions placed upon horizontal or vertical partitioning of tables and rows in the TPC-C benchmark, any such partitioning must be disclosed.

Horizontal partitioning was implemented on all the files except ITEM. The partitioning was done based on the Warehouse ID key field. The following eight tables: Warehouse, District, Customer, History, Neworder, Orders, Orderline, and Stock were split such that records for Warehouse ID's 1 through 6,800 were in the first partition and 6,801 through 13,600 were in the second partition.

3.0 Clause 2: Transaction and Terminal Profiles - Related Items

3.1 Verification for the Random Number Generator

The method of verification for the random number generation must be disclosed.

The `srandom()`, `getpid()`, and `gettimeofday()` functions are used to produce unique random seeds for each driver. The drivers use these seeds to seed the `srand()`, `srandom()`, and `srand48()` functions. Random numbers are produced using wrappers around the standard system random number generators.

The negative exponential distribution uses the following function to generate the distribution. This function has the property of producing a negative exponential curve with a specified average and a maximum value 4 times the average.

```
const double RANDOM_4_Z=0.89837799236185
const double RANDOM_4_K=0.97249842407114

double neg_exp_4(double average {
    return - average * (1/RANDOM_4_Z * log (1 - RANDOM_4_K * drand48()));
})
```

The random functions used by the driver system and the database generation program were verified. The `C_LAST` column was queried to verify the random values produced by the database generation program. After a measurement, the `HISTORY`, `ORDER`, and `ORDER_LINE` tables were queried to verify the randomness of values generated by the driver. The rows were counted and grouped by customer and item numbers.

Here is an example of one SQL query used to verify the random number generation functions:

- `create table TEMP (W_ID int, D_ID, C_LAST char(16), CNTR int);`
- `insert into TEMP select C_W_ID, C_D_ID, C_LAST, COUNT(*) from CUSTOMER group by C_W_ID, C_D_ID, C_LAST;`
- `select CNTR, COUNT(*) from TEMP group by CNTR order by 1;`

3.2 Input/Output Screens

The actual layouts of the terminal input/out screens must be disclosed. (8.1.3.2)

The screen layouts are based on those in Clauses 2.4.3, 2.5.3, 2.6.3, 2.7.3 and 2.8.3 of the TPC Benchmark C Standard Specification. .

3.3 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 must for the demonstration in 8.1.3.3 must be disclosed and commercially available (including supporting software and maintenance). (8.1.3.3)

The auditor verified terminal features by direct experimentation. The benchmarked configuration uses a browser and HTML scripts as the terminal interface

The following numbered items correspond directly to the seven items listed under Clause 2.2.2.4 with a description of how the requirement was met.

3.4 Presentation Managers

Any usage of presentation managers or intelligent terminals must be explained. (8.1.3.4)

The terminals emulated in the priced configuration are IBM RS/6000 desktop computer systems. All processing of the input/output screens was handled by the IBM iSeries 400 clients. The screen input/output was managed via HTML strings that comply with the HTML Version 2.0 specification. A listing of the code used to implement the intelligent terminals is provided in Appendix A. All data manipulation was handled by the IBM iSeries 400 clients.

3.5 Home and Remote Order Lines

The percentage of home and remote order lines in the New-Order transactions must be disclosed.

Table 1 on page 21 shows the percentage of home and remote transactions that occurred during the measurement period for the New-Order transactions.

3.6 New-Order Rollback transactions

The percentage of New-Order transactions that were rolled back as a result of an illegal item number must be disclosed.

Table 1 on page 21 shows the percentage of New-Order transactions that were rolled back due to an illegal item being entered.

3.7 Number of Items per Order

The number of items per orders entered by New-Order transactions must be disclosed.

Table 1 on page 21 shows the average number of items ordered per New-Order transaction.

3.8 Home and Remote Payment Transactions

The percentage of Home and Remote Payment transactions must be disclosed.

Table 1 on page 21 shows the percentage of Home and Remote transactions that occurred during the measurement period for the Payment transactions.

3.9 Nonprimary Key Transactions

The percentage of Payment and Order-Status transactions that used nonprimary key (C_LAST) access to the database must be disclosed.

Table 1 on page 21 shows the percentage of nonprimary key access to the database by the Payment and Order-Status transactions.

3.10 Skipped Delivery Transactions

The percentage of Delivery transactions that were skipped as a result of an insufficient number of rows in the NEW-ORDER table must be disclosed.

Table 1 on page 21 shows the percentage of Delivery transactions missed due to a shortage of supply in the NEW-ORDER table.

3.11 Mix of Transaction Types

The mix (ie, percentages) of transaction types seen by the SUT must be disclosed.

Table 1 on page 21 shows the mix percentage for each of the transaction types executed by the SUT.

3.12 Queuing Mechanism of Delivery

The queuing mechanism used to defer execution of the Delivery transaction must be disclosed.

Deferred queuing of the Delivery transaction is handled within standard support from the Tuxedo transaction monitor.

Table 1. Numerical Quantities for Transaction and Terminal Profiles

New Order	IBM ~ iSeries 400 Model 840-2420-001
Percentage of Home order lines	99.00%
Percentage of Remote order lines	1.00%
Rolled Back Transactions	0.99%
Number of Items per order	10
Payment	
Percentage of Home transactions	85.03%
Percentage of Remote transactions	14.97%
Nonprimary Key Access	
Percentage of Payment using C_LAST	59.99%
Percentage of Order-Status using C_LAST	60.04%
Delivery	
Delivery transactions skipped	0
Transaction Mix	
New-Order	44.88%
Payment	43.05%
Order-Status	4.03%
Stock-Level	4.02%
Delivery	4.03%

4.0 Clause 3: Transaction and System Properties - Related Items

The results of the ACID test must be disclosed along with a description of how the ACID requirements were met.

4.1 Atomicity Requirements

The system under test must guarantee that database 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.

4.1.1 Atomicity of Completed Transaction

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

The following steps were performed to verify the atomicity of completed transactions:

1. Randomly select a Customer, District, and Warehouse, and query the Customer, District, and Warehouse tables.
2. Execute a Payment transaction for the Customer, District, and Warehouse used in Step1 above. Commit the transaction.
3. Repeat the query performed in Step1 to demonstrate that the appropriate changes have been made.

4.1.2 Atomicity of Aborted Transactions

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

The following steps were performed to verify the atomicity of the aborted Payment transaction:

1. Randomly select a Customer, District, and Warehouse, and query the Customer, District, and Warehouse tables.
2. Execute a Payment transaction for the Customer, District, and Warehouse used in Step1 above. Roll-back the transaction.
3. Repeat the query performed in Step1 to verify that no changes have been made to the database.

4.2 Consistency Requirements

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

4.2.1 Consistency Condition 1

Entries in the WAREHOUSE and DISTRICT tables must satisfy the relationship:

$$W_YTD = \text{sum}(D_YTD)$$

for each warehouse defined by ($W_ID = D_W_ID$)

The following SQL queries were executed before and after transactions were run to show that the database was always in a consistent state.

```
Select WID, WYTD from WRHS
Select DWID, sum(DYTD) from DSTRCT group by DWID
```

The results of these two queries were then compared to verify consistency.

4.2.2 Consistency Condition 2

Entries in the DISTRICT, ORDER, and NEW-ORDER tables must satisfy the relationship:

$$D_NEXT_O_ID - 1 = \text{max}(O_ID) = \text{max}(NO_O_ID)$$

for each district defined by ($D_W_ID = O_W_ID = NO_W_ID$) and ($D_ID = O_D_ID = NO_D_ID$). This condition does not apply to the NEW-ORDER table for any districts which have no outstanding new orders.

The following SQL queries were executed before and after transactions were run to show that the database was always in a consistent state.

```
Create View QRYTEMP1(OWID, ODID, MAXOID)
  As Select OWID, ODID, MAX(OID) from ORDERS group by OWID, ODID
Create View QRYTEMP2(NOWID, NODID, MAXNOOID)
  As Select NOWID, NODID, MAX(NOOID) from NEWORDS group by NOWID, NODID
Select DWID, DID, (DNXTOR-1), MAXOID, MAXNOOID
  from DSTRCT, QRYTEMP1, QRYTEMP2 where DWID = OWID and DWID=NOWID
  and DID =ODID and DID = NODID and (((DNXTOR-1) <> MAXOID)
  or ((DNXTOR-1) <> MAXNOOID) or (MAXOID <> MAXNOOID))
```

If any records are produced by these queries, the database would be inconsistent. No records were produced by these queries.

4.2.3 Consistency Condition 3

Entries in NEW-ORDER table must satisfy the relationship:

$$\max(NO_O_ID) - \min(NO_O_ID) + 1 = \{\text{number of rows in the NEW-ORDER table for this district}\}$$

for each district defined by NO_W_ID and NO_D_ID. This condition does not apply to any districts which have no outstanding new orders.

The following SQL queries were executed before and after transactions were run to show that the database was always in a consistent state.

```
Create View QRYTEMP3(NOWID, NODID, MAXNOOID, MINNOOID, MAXMIN, COUNTO ID)
  As Select NOWID, NODID, MAX(NOOID), MIN(NOOID), (MAX(NOOID) -
  MIN(NOOID) +1), COUNT(*) from NEWORD group by NOWID, NODID
Select * from QRYTEMP3 where MAXMIN <> COUNTOID
```

If any records are produced by these queries, the database would be inconsistent. No records were produced by these queries.

4.2.4 Consistency Condition 4

Entries in the ORDER and ORDER-LINE tables must satisfy the relationship:

$$\text{sum}(O_OL_CNT) = \{\text{number of rows in the ORDER-LINE table for this district}\}$$

for each district defined by (O_W_ID = OL_W_ID) and (O_D_ID = OL_D_ID).

The following SQL queries were executed before and after transactions were run to show that the database was always in a consistent state.

```
Create View QRYTEMP4(OLWID, OLDID, COUNTORD)
  As Select OLWID, OLDID, COUNT(*)
  From ORDERLINE Group by OLWID, OLDID
Create View QRYTEMP5(OWID, ODID, SUMOLINES)
  As Select OWID, ODID, SUM(OLINES)
  From ORDERS Group by OWID, ODID
Select OWID, ODID, SUMOLINES, COUNTORD
  From QRYTEMP4, QRYTEMP5 Where OWID = OLWID and ODID = OLDID
  and SUMOLINES <> COUNTORD Order by OWID, ODID
```

If any records are produced by these queries, the database would be inconsistent. No records were produced by these queries.

4.2.5 Consistency Condition 5

For any row in the ORDER table, O_CARRIER_ID is set to a null value if and only if there is a corresponding row in the NEW-ORDER table defined by (O_W_ID, O_D_ID, O_ID) = (NO_W_ID, NO_D_ID, NO_O_ID).

This consistency test completed successfully.

4.2.6 Consistency Condition 6

For any row in the ORDER table, O_OL_CNT must equal the number of rows in the ORDER-LINE table for the corresponding order defined by $(O_W_ID, O_D_ID, O_ID) = (OL_W_ID, OL_D_ID, OL_O_ID)$.

This consistency test completed successfully.

4.2.7 Consistency Condition 7

For any row in the ORDER-LINE table, $OL_DELIVERY_D$ is set to a null date/time if and only if the corresponding row in the ORDER table defined by $(O_W_ID, O_D_ID, O_ID) = (OL_W_ID, OL_D_ID, OL_O_ID)$ has $(O_CARRIER_ID)$ set to a null value.

This consistency test completed successfully.

4.2.8 Consistency Condition 8

Entries in the WAREHOUSE and HISTORY tables must satisfy the relationship:

$$W_YTD = \text{sum}(H_AMOUNT)$$

for each warehouse defined by $(W_ID = H_W_ID)$.

This consistency test completed successfully.

4.2.9 Consistency Condition 9

Entries in the DISTRICT and HISTORY tables must satisfy the relationship:

$$D_YTD = \text{sum}(H_AMOUNT)$$

for each district defined by $(D_W_ID, D_ID = H_W_ID, H_D_ID)$.

This consistency test completed successfully.

4.2.10 Consistency Condition 10

Entries in the CUSTOMER, HISTORY, ORDER, and ORDER-LINE tables must satisfy the relationship:

$$C_BALANCE = \text{sum}(OL_AMOUNT) - \text{sum}(H_AMOUNT)$$

where:

H_AMOUNT is selected by $(C_W_ID, C_D_ID, C_ID) = (H_C_W_ID, H_C_D_ID, H_C_ID)$

and:

OL_AMOUNT is selected by:

$(OL_W_ID, OL_D_ID, OL_O_ID) = (O_W_ID, O_D_ID, O_ID)$ and

$(O_W_ID, O_D_ID, O_C_ID) = (C_W_ID, C_D_ID, C_ID)$ and

$(OL_DELIVERY_D)$ is not a null value

This consistency test completed successfully.

4.2.11 Consistency Condition 11

Entries in the CUSTOMER, ORDER, and NEW-ORDER tables must satisfy the relationship:

$$(count(*) \text{ from } ORDER) - (count(*) \text{ from } NEW-ORDER) = sum(C_DELIVERY_CNT)$$

for each district defined by (O_W_ID, O_D_ID) = (NO_W_ID, NO_D_ID) = (C_W_ID, C_D_ID).

This consistency test completed successfully.

4.2.12 Consistency Condition 12

Entries in the CUSTOMER, and ORDER-LINE table must satisfy the relationship:

$$C_BALANCE + C_YTD_PAYMENT = sum(OL_AMOUNT)$$

for any randomly selected customers and where OL_DELIVERY_ID is not set to a null date/time.

This consistency test completed successfully.

All 12 consistency tests were completed successfully.

4.2.13 Consistency Tests

Verify that the database is initially consistent by verifying that it meets the consistency conditions defined in Clauses 3.3.2.1 to 3.3.2.4. Describe the steps used to do this in sufficient detail so that the steps are independently repeatable.

The queries defined in 4.2.1 through 4.2.4 were run after initial database build and prior to executing any transactions. All queries showed that the database was in a consistent state.

After executing transactions at full load for approximately 10 minutes, the queries defined in 4.2.1 through 4.2.4 were run again. All queries showed that the database was still in a consistent state.

4.3 Isolation Requirements

Operations of concurrent database 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.

4.3.1 Isolation Test 1

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

The following steps were performed to satisfy the test of isolation for Order-Status and New-Order transactions:

1. First terminal: Start a New-Order transaction with the necessary inputs. The transaction is delayed to pause the program execution.
2. Second terminal: Start an Order-Status transaction for the same customer as used in the New-Order transaction.
3. Second terminal: The Order-Status transaction attempts to read the CUSTOMER file but is locked out by the New-Order transaction waiting to complete.
4. First terminal: The New-Order transaction is released and the Commit is executed releasing the record. With the CUSTOMER record now released, the Order-Status transaction can now complete.
5. Second terminal: Verify that the Order-Status transaction completes after the New-Order transaction. and that the results displayed for the Order-Status transaction match the input for the New-Order transaction.

4.3.2 Isolation Test 2

This test demonstrates isolation for read-write conflicts of Order-Status and New-Order transactions when the New-Order transaction is rolled back.

The following steps were performed to satisfy the test of isolation for Order-Status and a rolled back New-Order transaction.

1. First terminal: Perform a New-Order transaction for the same customer in the Order-Status transaction in Isolation Test 1; include an invalid item number in the order. The transaction is delayed just prior to the rollback.
2. Second terminal: Start an Order-Status transaction for the same customer used in the New-Order transaction. The Order-Status transaction attempts to read the CUSTOMER file but is locked by the New-Order transaction.
3. First terminal: Roll back the New-Order transaction. With the CUSTOMER record now released, the Order-Status transaction completes.
4. Verify the results from the Order-Status transaction matches those in Isolation Test 1.

4.3.3 Isolation Test 3

This test demonstrates isolation for write-write conflicts of two New-Order transactions.

1. The following steps were performed to verify isolation of two New-Order transactions:
2. First terminal: Start a New-Order transaction using the necessary inputs. The transaction is delayed just prior to the Commit.
3. Second terminal: Start a second New-Order transaction for the same customer used by the first terminal. This transaction is forced to wait while the first terminal holds a lock on the DISTRICT record requested by the second terminal.
4. First terminal: The New-Order transaction is allowed to complete and Commit the transaction. With the DISTRICT record released, the second terminal New-Order transaction will complete.
5. Verify the order number from the second terminal New-Order transaction is one greater than the order number from the first terminal.

4.3.4 Isolation Test 4

This test demonstrates isolation for write-write conflicts of two New-Order transactions when one transaction is rolled back.

The following steps were performed to verify isolation of two New-Order transactions after one is rolled back:

1. First terminal: Start a New-Order transaction using the necessary inputs to cause a roll back (invalid item number). The transaction is delayed just prior to the rollback.
2. Second terminal: Start a second New-Order transaction for the same customer used by the first terminal. This transaction is forced to wait while the first terminal holds a lock on the DISTRICT record requested by the second terminal.
3. First terminal: The New-Order transaction is allowed to complete, and the transaction is rolled back due to the invalid item number.
4. Second terminal: With the DISTRICT record released, the second terminal New-Order transaction will complete normally.
5. Verify the order number from the second terminal New-Order transaction is equal to the next order number before either New-Order transaction was started.

4.3.5 Isolation Test 5

This test demonstrates isolation for write-write conflicts of Payment and Delivery transactions.

The following steps were performed to successfully conduct this test:

1. First terminal: A Delivery transaction is started. The transaction is delayed just prior to the Commit.
2. Second terminal: Start a Payment transaction for a customer that will have an order delivered in this transaction started in Step 1. The Payment transaction is forced to wait while the Delivery transaction holds a lock on the CUSTOMER record.
3. The Delivery transaction completes.
4. Second terminal: With the CUSTOMER record released, the Payment transaction is now able to complete.

4.3.6 Isolation Test 6

This test demonstrates isolation for write-write conflicts of Payment and Delivery transactions when the Delivery transaction is rolled back.

The following steps were performed to successfully conduct this test:

1. First terminal: Start a Delivery transaction. The transaction is delayed just prior to the rollback.
2. Second terminal: Start a Payment transaction for a customer that will have an order delivered in this transaction started in Step 1. The Payment transaction is forced to wait while the Delivery transaction holds a lock on the CUSTOMER record.
3. The Delivery transaction rolls back.
4. Second terminal: With the CUSTOMER record released, the Payment transaction is now able to complete.

4.3.7 Isolation Test 7

This test demonstrates repeatable reads for the New-Order transaction while an interactive transaction updates the price of an item.

The following steps were performed to successfully conduct this test:

1. First terminal: Execute a New Order transaction including items x and y.

2. First terminal: A New-Order transaction is started that contains item x twice and item y once. This transaction is stopped after reading the price of item x from the item file the first time.
3. Second terminal: Using interactive SQL, an update transaction is started for items x and y, increasing their price by 10%. Case A, transaction 3 stalls, occurs.
4. First terminal: The New-Order transaction is allowed to complete. It is verified that the prices for items x and y are the same throughout the entire transaction and that they match the results of Step 1.
5. Second terminal: After the New-Order transaction completes, the update transaction completes and is committed.
6. First terminal: Step1 is repeated, noting that the prices of items x and y now match those set in Step 3.

4.3.7.1 Isolation Test 8

This test demonstrates isolation for phantom protection between a Delivery and a New-Order transaction.

The following steps were performed to successfully conduct this test:

1. First terminal: All rows for a randomly selected district and warehouse were removed from the NEW-ORDER table.
2. First terminal: A Delivery transaction for the selected warehouse was started.
3. First terminal: The Delivery transaction was stopped immediately after reading the NEW-ORDER table for the selected district. No qualifying row was found.
4. Second terminal: A New-Order transaction was started for the same warehouse and district. Case A, Transaction 2 stalled.
5. First terminal: Repeated read of the NEW-ORDER table for the selected district.
6. Again no qualifying row was found.
7. First terminal: The Delivery transaction was allowed to complete and was COMMITTED.
8. Second terminal: The NEW-ORDER transaction completed successfully.

4.3.7.2 Isolation Test 9

This test demonstrates isolation for phantom protection between an Order-Status and a New-Order transaction.

The following steps were performed to successfully conduct this test:

1. First terminal: An Order-Status transaction for a selected customer was started.
2. First terminal: The Order-Status transaction was stopped immediately after reading the ORDER table for the selected customer. The most recent order for that customer was found.
3. Second terminal: A NEW-ORDER transaction was started for the same customer. Case A, Transaction 2 stalled.
4. First terminal: Repeated read of the ORDER table for the selected customer.
5. Verified the order found was the same as in step 3.
6. First terminal: The Order-Status transaction was allowed to complete and was COMMITTED.
7. Second terminal: The NEW-ORDER transaction completed successfully.

4.4 Durability Requirements

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

4.4.1 Permanent Unrecoverable Failure of any Single Durable Medium

Permanent unrecoverable failure of any single durable medium containing TPC-C database tables or recovery log data.

The iSeries 400 Model 840-2420-001 implementation of the TPC Benchmark™ C divides the configured disk space into two available auxiliary storage pools (ASP): System ASP and User ASP. The system ASP contains the operating system, integrated relational database, the application libraries, and the TPC-C tables. The System ASP is protected by device parity protection (RAID-5) and the User ASP, which contains the journal receiver (recovery log), is mirrored.

4.4.1.1 Failure of Durable Medium of Journal Receiver and Instantaneous Interruption and Memory Failure

The following steps were performed to successfully complete the test of the Durability of the journal receiver:

1. The current count of the total number of orders was determined by the sum of D_NEXT_O_ID of all rows in the DISTRICT table giving SUM_1.
2. A full scale test was started on the SUT. The test was allowed to run for 10 minutes before creating the failure.
3. The signal cable from a single disk unit in the User ASP was disconnected. Since the User ASP is protected by mirroring, the system continued to process transactions. Detection of the device failure caused a diagnostic message to be issued. Processing of transactions continued without performance degradation.
4. Processing was allowed to continue for an additional 10 minutes.
5. An instantaneous power failure was simulated by issuing an immediate power-off at the service panel.
6. The system was then powered on and IPLed.
7. The number of New-Order transactions executed by the SUT is verified against the number of successful transactions logged by the RTE.
8. Step1 above was performed again retrieving the new total of orders processed, SUM_2. The difference between SUM_2 and SUM_1 was compared to the number of transactions reported by RTE.

4.4.1.2 Failure of Durable Medium of Database

The following steps were performed to successfully perform the Durability test of failure of a disk unit with database tables:

Note: This test was combined with the tests in 4.4.1.1, because the ASP with the disk units containing the database tables is protected by device parity protection (RAID5).

1. The database tables reside on the system ASP.
2. After a disk unit in user ASP was disconnected as mentioned in step 3 in 4.4.1.1, and we let the processing continue for about 10 minutes as mentioned in step 4 in 4.4.1.1, a disk unit in the system ASP was disconnected.

Since the system ASP is protected by device parity protection, (RAID 5) the system continued to process transactions. Detection of the device failure caused a diagnostic message to be issued. This also resulted in parity protection being suspended for the parity set that that disk unit was in. Processing of transactions continued without performance degradation.

3. After the system was powered off, the disk unit was plugged in.
4. After the system was powered back on, the disk was reinstalled and the parity protection was resumed again for that parity set.

5.0 Clause 4: Scaling and Database Population - Related Items

5.1 Cardinality of Tables

The cardinality (ie, the number of rows) of each table, as it existed at the start of the benchmark run, must be disclosed.

Table 2 portrays the TPC Benchmark™ C defined tables and the number of rows for each table as they were built initially.

Table 2. Initial Database Build (# of rows per table)

TPC Benchmark C Tables	IBM ~ iSeries 400 Model 840-2420-001
WAREHOUSE	13,600
CUSTOMER	408,000,000
NEW-ORDER	122,400,000
DISTRICT	136,000
STOCK	1,360,000,000
ORDERS	408,000,000
ORDER-LINE	4,080,062,172
HISTORY	408,000,000
ITEM	100,000

5.2 Distribution of Tables and Logs

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

The iSeries 400 system utilizes a Single-Level Storage concept where OS/400 views all drives in an Auxiliary Storage Pool (ASP) as a single virtual drive. This technique spreads information across all available drives in an ASP, attempting to maintain equivalent percentages of free storage. For this Benchmark, a 930-Disk RAID-5 ASP was used for system code, application code, and the database. A separate, fully mirrored, 150-Disk ASP was used for log data.

5.3 Database Model Implemented

A statement must be provided that describes the database model implemented by the DBMS used.

The type of database implemented in all iSeries 400 systems is an integrated relational database. The database is integrated into the OS/400 operating system.

5.4 Partitions/Replications Mapping

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

Horizontal partitioning was implemented on all the files except ITEM. The partitioning was done based on the Warehouse ID key field. The following eight tables: Warehouse, District, Customer, History, Neworder, Orders, Orderline, and Stock were split such that records for Warehouse ID's 1 through 6,800 were in the first partition and 6,801 through 13,600 were in the second partition.

6.0 Clause 5: Performance Metrics and Response Time - Related Items

6.1 Response Times

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

Table 3 lists the response times and the ninetieth percentiles for each of the transaction types for the IBM ~ iSeries 400 Model 840-2420-001 system.

6.2 Keying and Think Times

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

Table 3 lists the keying and think times from the measured TPC-C tests for the IBM iSeries 400 Model 840-2420-001.

Table 3. IBM ~ iSeries 400 Model 840-2420-001 - Response, Keying, and Think Times

Response Times	NewOrder	Payment	Order Status	Delivery (int/def)	Stock Level	Menus
90%	0.5	0.3	0.5	0.2/6.8	1.7	0.1
Average	0.32	0.20	0.31	0.10/3.17	0.82	0.10
Maximum	5.01	5.21	4.43	0.21/21.37	9.22	0.32
			Think Times			
Minimum	0.00	0.00	0.00	0.00	0.00	N/A
Average	12.02	12.01	10.01	5.04	5.02	N/A
Maximum	120.24	120.23	100.22	50.21	50.21	N/A
			Keying Times			
Minimum	18.00	3.00	2.00	2.00	2.00	N/A
Average	18.00	3.00	2.00	2.00	2.00	N/A
Maximum	18.14	3.12	2.16	2.07	2.06	N/A

6.3 Response Time Frequency Distribution

Response time frequency distribution curves must be reported for each transaction type.

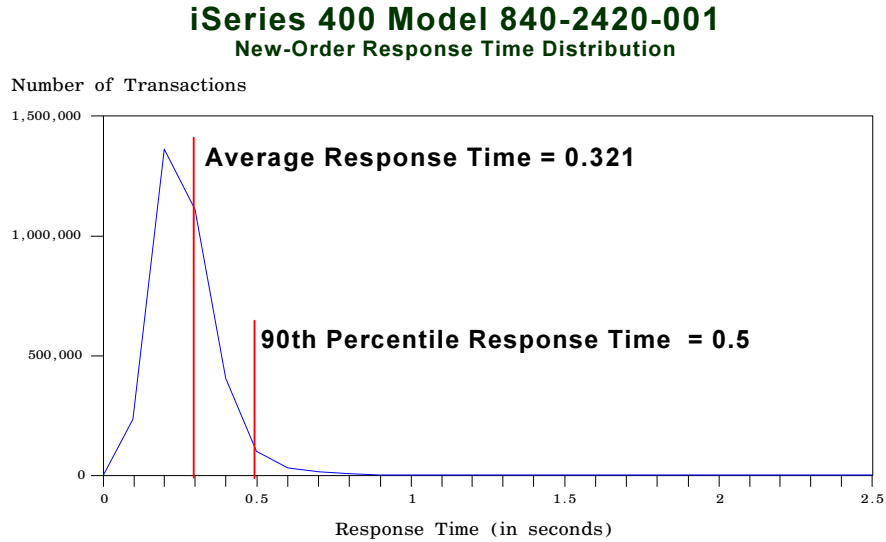


Figure 1. iSeries 400 Model 840-2420-001 New-Order Response Time Distribution

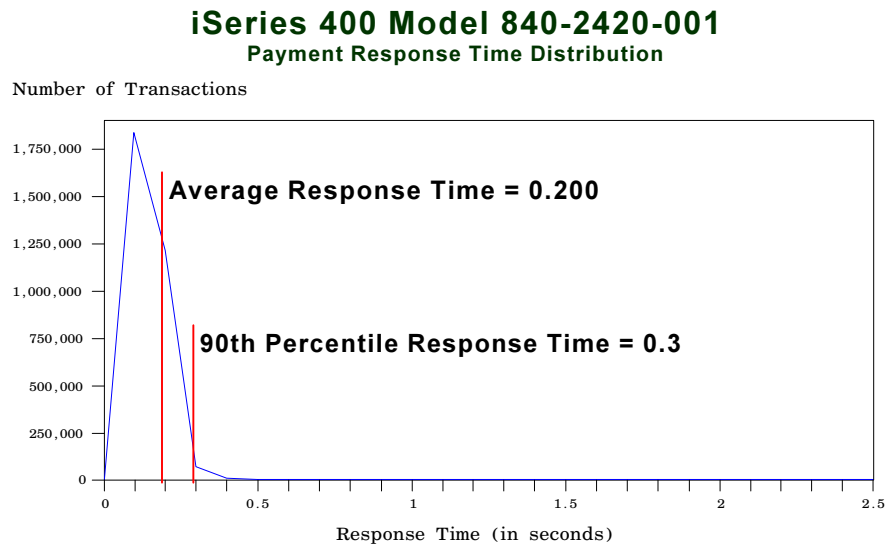


Figure 2. iSeries 400 Model 840-2420-001 Payment Response Time Distribution

iSeries 400 Model 840-2420-001 Order Status Time Distribution

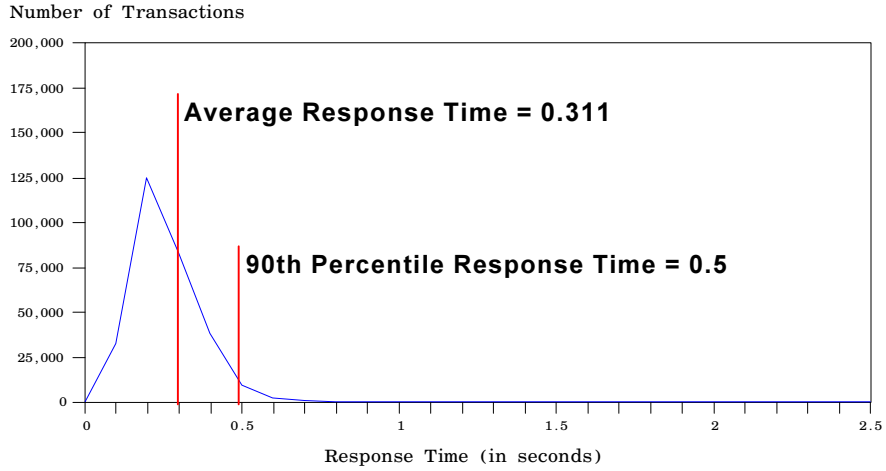


Figure 3. iSeries 400 Model 840-2420-001 Order Status Response Time Distribution

iSeries 400 Model 840-2420-001 Delivery Response Time Distribution (Interactive)

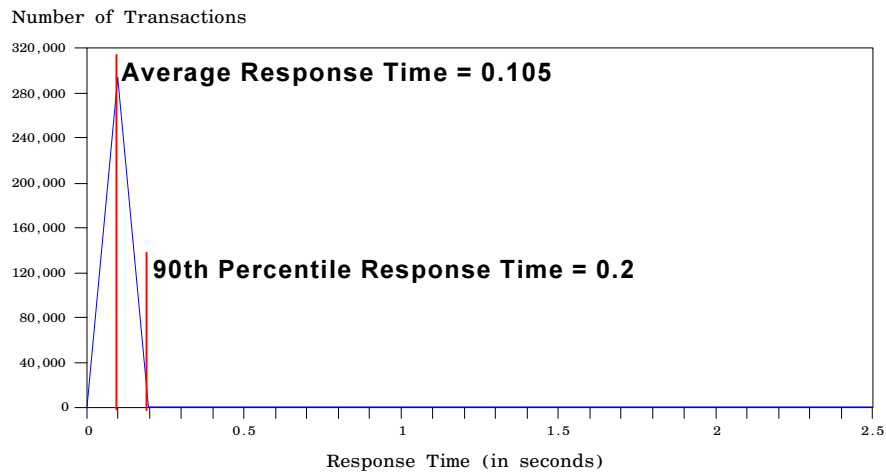


Figure 4. iSeries 400 Model 840-2420-001 Delivery (Interactive) Response Time Distribution

iSeries 400 Model 840-2420-001 Delivery Response Time Distribution (Batch)

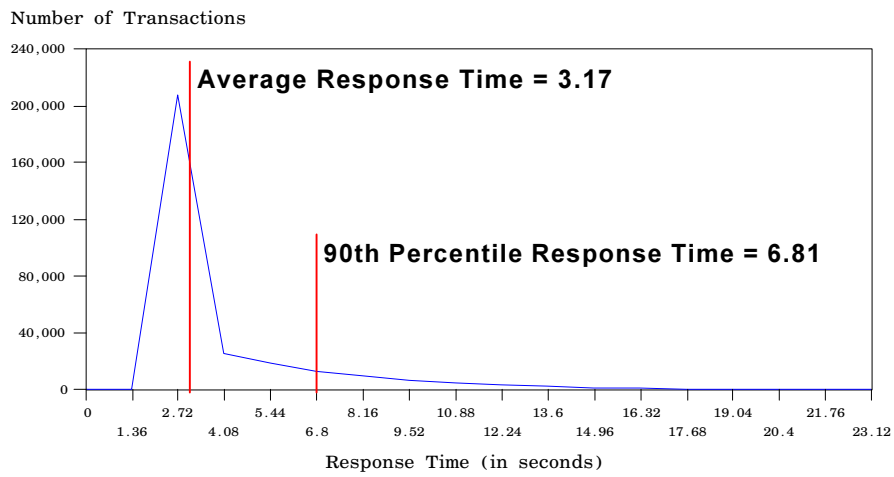


Figure 5. iSeries 400 Model 840-2420-001 Delivery (Batch) Response Time Distribution

iSeries 400 Model 840-2420-001 Stock-Level Time Distribution

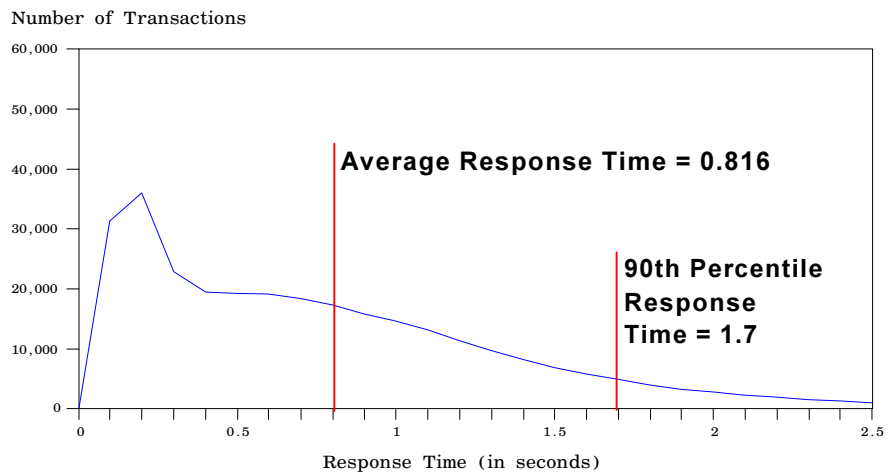


Figure 6. iSeries 400 Model 840-2420-001 Stock-Level Response Time Distribution

6.4 Performance Curve for Response Time versus Throughput

The performance curve for response times versus throughput must be reported for the New-Order transaction.

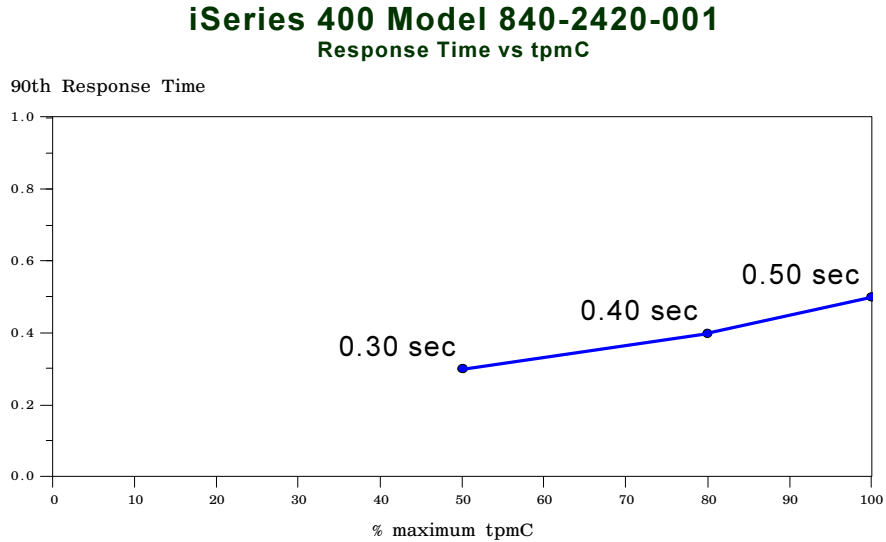


Figure 7. iSeries 400 Model 840-2420-001 New-Order Response Time Versus Throughput

6.5 Think Time Frequency Distribution

Think time frequency distribution curves must be reported for each transaction type.

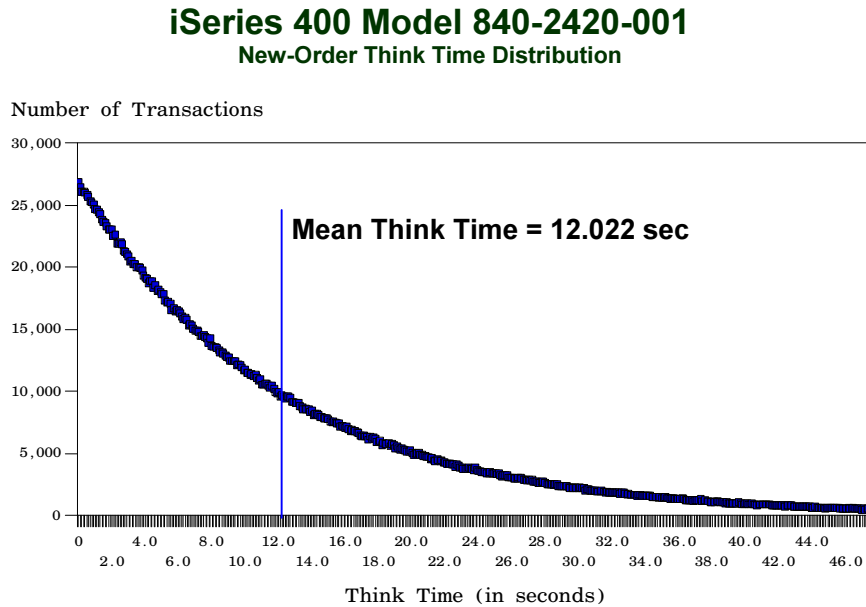


Figure 8. iSeries 400 Model 840-2420-001 New-Order Think Time Distribution

