TPC Benchmark C
Full Disclosure Report
IBM AS/400
AS/400e server model 740 with feature code 2070

June 3, 1999

Special Notices

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

- 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 May 28, 1999

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.

Abstract

This report documents the full disclosure information required by the TPC Benchmark™ C Standard Specification dated April 25, 1998 for measurements on the IBM AS/400e server model 740 with feature code 2070.

The software used on the AS/400 systems includes OS/400 Version 4, Release 3, Modification 0, DB2 for AS/400 Version 4, Release 3, Modification 0, OS/400 Version 4, Release 4, 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 AS/400e server model 740 with feature code 2070

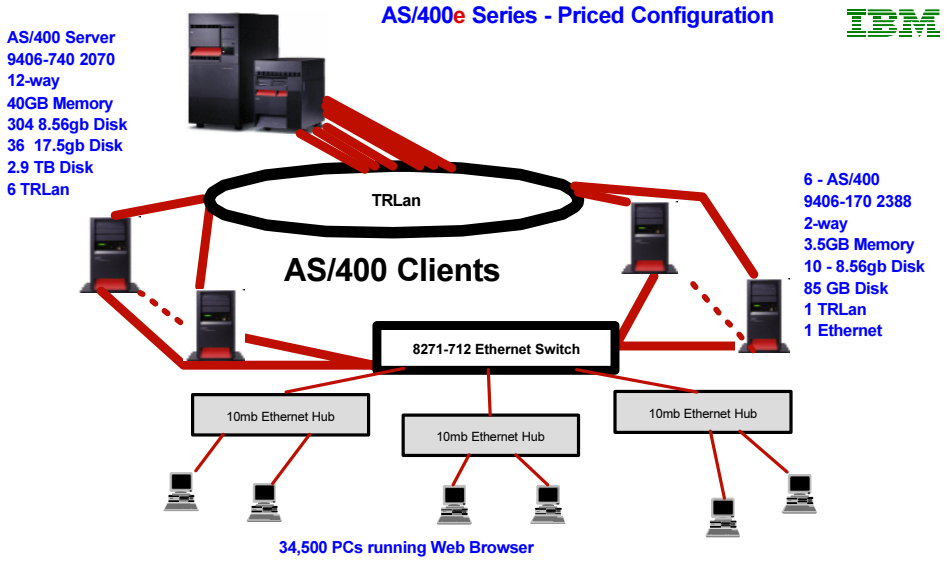
Company Name	System Name	Data Base Software	Operating System Software
IBM®	AS/400e server 740-2070 C/S	DB2 for AS/400 Version 4 Release 3	OS/400 Version 4 Release 3
Availability Date: June 1, 1999			

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 ($3,037,802/43,419.15)
$3,037,802	43,419.15 tpmC	$69.96 per tpmC

IBM	AS/400e server model 740 with feature code 2070	TPC-C Rev. 3.4 Report Date: June 3, 1999	
Total System Cost	**TPC-C Throughput**	**Price/Performance**	**Availability Date**
$3,037,802	**43,419.15**	**$69.96**	**June 1, 1999**

Processors	Database Manager	Operating System	Other Software	Number of Users
1 AS/400e 740-2070 (12-way) 6 AS/400e 9406-170 2388	DB2 for AS/400 Version 4 Release 3	OS/400 Version 4 Release 3	BEA TUXEDO 6.4 V4 R4 INT LNG ENV C OS/400 V4 R4	34,500

AS/400e server 740-2070 C/S Priced Configuration



System components:	Qty:	Description:
Server	1	
Processors	12	AS/400e server 740-2070
Memory		40 GB main memory
Disk controllers	20	6533 DASD IOP
	1	Integrated multifunction IOP
Disk drives	340	304 8589 MB DASD 36 17542 MB DASD
Total storage		2.9 TB
Clients (each):	6	
Processor	2	AS/400e server 9406-170 2388
Memory		3.5 GB main memory
Disk controller	1	Integrated multifunction IOP
Disk	10	8568 MB DASD

AS/400e server model 740-2070

TPC-C REV 3.4 EXECUTIVE SUMMARY

Report Date: June 3, 1999

AS/400 9406-740 Configuration

Item Description	Feature Number	Third Party Pricing	Unit Price	Qty.	Extended Price	Extended Maintenance	5-Year Price
Server Hardware:							
AS/400 9406 - Model 740 12-way Processor	2070		342,500	1	342,500	80,400	422,900
Token-Ring Adapter (9249)	Bundled			1	0		0
Battery Backup	Bundled			1	0		0
Twinax Work Station Controller (5540)	Bundled			1	0		0
2048 Optional MB Base Memory	8193		32,256	4	129,024		129,024
Optical Link	2688		2,000	6	12,000		12,000
Optical Bus Adapter	2695		1,000	1	1,000		1,000
2048 MB Main Storage	3193		36,864	16	589,824		589,824
Storage Expansion Unit	5057		5,000	1	5,000	10,560	15,560
Storage Expansion Unit	5058		5,000	10	50,000		50,000
1063 Mbps System Unit Expansion Tower	5073		14,900	1	14,900	105,600	120,500
1063 Mbps Storage Expansion Tower	5083		14,900	10	149,000	144,000	293,000
Token Ring Adapter	6149		1,200	6	7,200		7,200
RAID Disk Unit Controller	6533		9,900	20	198,000		198,000
2.5 GB 1/4-inch Tape Drive	6382		1,300	1	1,300		1,300
Lan/Wan/Workstation IOP	2629		2,600	6	15,600		15,600
Operations Console Cable	0328		125	1	125		125
V.24./EIA232 20 ft Cable	0330		125	1	125		125
17.2 GB Disk Unit	6714		3,000	36	108,000		108,000
8.58 GB Disk Unit	6713		1,650	303	499,950		499,950
Base 8.58 GB Disk Unit	8713		600	1	600		600
External V.34 modem	7852		525	1	525		525
Server Subtotal					*2,124,673*	*340,560*	*2,464,708*
Client Hardware:							
AS/400 9406-170 V4R4	2388		105,125	1	105,125	20,160	125,285
Twinax Work Station Controller (9720)	Bundled			1	0		0
Base Token Ring Adapter (9724)	Bundled			1	0		0
Tape Drive	6381		1,000	1	1,000		1,000
128 MB Main Storage	3002		1,280	2	2,560		2,560
256 MB Main Storage	3004		2,560	12	30,720		30,720
Opt Base 8.58 GB Disk Unit	8813		600	1	600		600
8.58 GB Disk Unit	6813		1,650	9	14,850		14,850
PCI Raid Disk Unit Ctlr	2740		4,000	1	4,000		4,000
PCI Lan/Wan/Workstation IOP	2809		1,800	1	1,800		1,800
PCI 100/10 Mbps Ethernet IOA	2838		900	1	900		900
32Mb IOP Memory (for IPCS)	2861		125	1	125		125
PCI Integrated Netfinity Server	2866		1,500	1	1,500		1,500
PCI System Exp Unit Base 16Mb	7101		3,000	1	3,000	2,640	5,640
External V.34 Modem	7852		525	1	525		525
Single Client Subtotal					*166,705*	*22,800*	*189,505*
Number of Clients				6			
Client Subtotal					*1,000,230*	*136,800*	*1,137,030*
Server Software:							
IBM Operating System/400 V4R3M0	Bundled			1	0		0
Client Software:							
IBM Operating System/400 V4R4M0	Bundled			6	0		0
BEA Tuxedo V6.4		1	3,000	6	18,000	14,400	32,400
ILE COBOL	5738-CB1		7,200	1	7,200		7,200
Application Development Toolkit	5716-PW1		7,850	1	7,850		7,850
DB2 Query Manager Dev. Toolkit	Bundled			1	0		0
ILE C	5716-CX2		7,200	1	7,200		7,200
Software Subtotal					*40,250*	*14,400*	*54,650*
User Connectivity:							
12-port 10/100 Ethernet Switch	8271-712		3,495	4	13,980	5,600	19,580
8-Port 10Mb Hub (10% spares)	Z99552	1	28	5449	150,120		150,120
8-port Token Ring connection unit (including 2 spares)	8226		545	4	2,180		2,180
User Connectivity Subtotal					*166,280*	*5,600*	*171,880*
5-year System Subtotal					*3,331,433*	*497,360*	*3,828,793*
Discounts:				**%Allowance**	**Volume**	**Discount**	
Revenue Allowance				20	3,181,313	(636,263)	
Mid-range System Option				17	497,360	(84,551)	
Extended Maintenance Option				17	412,809	(70,177)	

Notes:
Revenue Allowance is applied to hardware and software for the priced configuration.
EMO is a discount for prepayment of 5 years of maintenace costs.
MSRO is available to customers when agreement is reached for the customer to perform certain duties

Five-Year Cost of Ownership: $3,037,802

tpmC Rating: 43,419.15

$/tpmC: $69.96

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. Thank you.

Numerical Quantities Summary for IBM AS/400e server 740-2070

MQTH, computed Maximum Qualified Throughput			43,419.15
% throughput difference, reported & reproductivity			0.07%
Response Times (in seconds)	90%	Avg.	Max.
- New Order	0.400	0.284	7.594
- Payment	0.300	0.204	8.391
- Order-Status	0.400	0.300	1.375
- Delivery (interactive portion)	0.200	0.111	0.657
- Delivery (deferred portion)	9.13	4.13	26.56
- Stock-Level	1.000	0.471	7.922
- Menu	0.111	0.110	6.671
- Response time delayed for emulated components	0.0		
Transaction Mix (in percent of total transactions)			
- New Order			44.85%
- Payment			43.09%
- Order-Status			4.03%
- Delivery			4.02%
- Stock-Level			4.02%
Keying/Think Times (in seconds)	Min.	Avg.	Max.
- New Order 18.00/120.313	18.00/.015	18.00/12.054	
- Payment 3.00/120.313	3.00/.015	3.00/12.031	
- Order-Status 2.00/100.610	2.00/.015	2.00/10.060	
- Delivery	2.00/.015	2.00/5.045	2.00/50.406
Stock-Level	2.00/.015	2.00/5.096	2.00/50.407
Test Duration			
- Ramp-up time			82 minutes
- Measurement interval			20 minutes
- Number of checkpoints			0
- Checkpoint interval			0 minutes
- Number of transactions (all types) completed in Measurement Interval 1,936,158			

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 AS/400 systems according to the TPC Benchmark“ C Standard Specification. Measurements were done on the AS/400e server 740-2070.

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 1
1.1 Application Code Disclosure 1
1.2 Benchmark Sponsor 1
1.3 Parameter Settings 1
1.4 Configuration Diagrams 1
1.5 AS/400e server 740-2070 C/S Benchmark Configuration 2
1.6 AS/400e server 740-2070 C/S Priced Configuration 3

2.0 Clause 1: Logical Database Design - Related Items 4
2.1 Table Definitions 4
2.2 Database Organization 4
2.3 Insert and/or Delete Operations 4
2.4 Horizontal or Vertical Partitioning 5

3.0 Clause 2: Transaction and Terminal Profiles - Related Items 6
3.1 Verification for the Random Number Generator 6
3.2 Input/Output Screens 6
3.3 Terminal Features 7
3.4 Presentation Managers 7
3.5 Home and Remote Order Lines 7
3.6 New-Order Rollback transactions 7
3.7 Number of Items per Order 7
3.8 Home and Remote Payment Transactions 7
3.9 Nonprimary Key Transactions 8
3.10 Skipped Delivery Transactions 8
3.11 Mix of Transaction Types 8
3.12 Queueing Mechanism of Delivery 8

4.0 Clause 3: Transaction and System Properties - Related Items 10
4.1 Atomicity Requirements 10
4.1.1 Atomicity of Completed Transaction 10
4.1.2 Atomicity of Aborted Transactions 10
4.2 Consistency Requirements 11
4.2.1 Consistency Condition 1 11
4.2.2 Consistency Condition 2 11
4.2.3 Consistency Condition 3 12
4.2.4 Consistency Condition 4 12

4.2.5 Consistency Condition 5 12
4.2.6 Consistency Condition 6 13
4.2.7 Consistency Condition 7 13
4.2.8 Consistency Condition 8 13
4.2.9 Consistency Condition 9 13
4.2.10 Consistency Condition 10 13
4.2.11 Consistency Condition 11 14
4.2.12 Consistency Condition 12 14
4.2.13 Consistency Tests 14
4.3 Isolation Requirements 14
4.3.1 Isolation Test 1 14
4.3.2 Isolation Test 2 15
4.3.3 Isolation Test 3 15
4.3.4 Isolation Test 4 15
4.3.5 Isolation Test 5 16
4.3.6 Isolation Test 6 16
4.3.7 Isolation Test 7 16
4.3.7.1 Isolation Test 8 16
4.3.7.2 Isolation Test 9 17
4.4 Durability Requirements 17
4.4.1 Permanent Unrecoverable Failure of any Single Durable Medium 18
4.4.1.1 Failure of Durable Medium of Journal Receiver and Instantaneous Interruption and Memory Failure 18
4.4.1.2 Failure of Durable Medium of Database 18

5.0 Clause 4: Scaling and Database Population - Related Items 20
5.1 Cardinality of Tables 20
5.2 Distribution of Tables and Logs 20
5.3 Database Model Implemented 20
5.4 Partitions/Replications Mapping 21

6.0 Clause 5: Performance Metrics and Response Time - Related Items 22
6.1 Response Times 22
6.2 Keying and Think Times 22
6.3 Client Substitution Table 23
6.4 Response Time Frequency Distribution 24
6.5 Performance Curve for Response Time versus Throughput 27
6.6 Think Time Frequency Distribution 27
6.7 Throughput Versus Elapsed Time 28
6.8 Work Performed During Steady State 28
6.9 Reproducibility 29
6.10 Measurement Interval 29

7.0 Clause 6: SUT, Driver, and Communication Definition-Related Items 30
7.1 RTE Availability 30
7.2 Functionality and Performance of Emulated Components 30
7.3 Network Bandwidth 30
7.4 Operator Intervention 30

8.0 Clause 7: Pricing - Related Items 31
8.1 Hardware and Programs Used 31
8.2 Five Year Cost of System Configuration 31
8.3 Statement of tpmC and Price/Performance 31
8.3.1 IBM AS/400e server 740-2070 Five-Year System Price Configuration 32

9.0 Clause 8: Audit - Related Items 33

Appendix A. System Parameters and User Profile 34
A.1 System Parameters 34
A.1.1 Transaction Subsystem Description 34
A.2 User Profile 34

Appendix B. Data Base File Definitions 35
B.1 AREFFIL: Reference File 35
B.2 CSTMRLFCRT: Customer Logical File 35
B.3 CSTMRLFNAM: Customer Names Logical File 35
B.4 CSTMR: Customer Logical File 35
B.5 CSTMRPF: Customer Physical File 35
B.6 DSTRCT: District File 36
B.7 HSTRY: History File 36
B.8 ITEM: Item Physical File 36
B.9 ITEMLF: Item Logical File 36
B.10 NEWORD: New Order Logical File 36
B.11 NEWORDLF: New Order Logical File 36
B.12 NEWORDPF: New Order Physical File 36
B.13 ORDERS: Orders Logical File 36
B.14 ORDERSLF: Orders Logical File 36
B.15 ORDERSPF: Orders Physical File 36
B.16 ORDLIN: Order Lines Logical File 36
B.17 ORDLINLF: Order Lines Logical File 37
B.18 ORDLINPF: Order Lines Physical File 37
B.19 STOCK: Stock Logical File 37
B.20 STOCKLF: Stock Logical File 37

Appendix C. Data Base Build Programs 38
C.1 Program Flow For Build of Server, Client, and Data Base 38
C.2 BLDTPCCPGM: Database Build Program 39
C.3 CRTBLDPGMS: Create Database Build Programs 39
C.4 NATBLD: Create Physical and Logical Files 39
C.5 CRTHASH: Create Hashes 40
C.6 DLTLOGICLS: Delete Logical Files 40
C.7 LOADTPCCF: Database Population Control Program 40
C.8 LOADW_D: Fill WRHS and DSTRCT files 41
C.9 LOADITEM: Fill ITEM and STOCK Files 42
C.10 LOADCST: Fill CSTMR and HSTRY Files 43
C.11 LOADORD: Fill ORDERS, ORDLIN, and NEWORD Files 45
C.12 SETUP: System Setup Program 47
C.13 ORDERSVIEW: Creates Orders View Logical Files 48
C.14 ORDLINVIEW: Creates Orders View Logical Files 48
C.15 NEWORDVIEW: Creates New Orders View Logical Files 48
C.16 CSTMRVIEW: Creates Customer View Logical Files 48
C.17 STRJRNTPCC: Start Journaling for all TPCC Physical Files 48
C.18 CRTENVPGMS: Create Environment Programs 48
C.19 C_CREATE: Create ITEM Hash 48
C.20 C_CREATE2: Create STOCK Hash 48
C.21 C_CREATE3: Create CSTMR Hash 49

Appendix D. Application Source Code 50
Client code

D.1 COMMON: Common functions used on the client & server 51
D.2 COMMON.H: Header file 51
D.3 DELVRYSRV: Tuxedo service program for Delivery transaction 51
D.4 FORMCHILDS: Prestart jobs program 51
D.5 FORMPARENT: Prestart jobs program 52
D.6 FORMSERVER: Main TPCC client program 54
D.7 GETDTTMSTR: Resolve time Program 56
D.8 IMPORT.H: Header file for scaninput program 57
D.9 NEWORDSRV: Tuxedo service program for neword transaction 58
D.10 ORDSTSSRV:Tuxedo service program for ordsts transaction 58
D.11 PAYMNTSRV:Tuxedo service program for payment transaction 58
D.12 POSTRESULT: Format and output program 59
D.13 SCANINPUT: Format input program 61
D.14 STKLVLSRV:Tuxedo service program for stklvl transaction 65
D.15 TUXCONFIG: Tuxedo configuration file 65
Server code

D.16 DELIVERY: Delivery Transaction Program 65

D.17 DLVFEND: Front end CL program for Delivery 67
D.18 DLVSRVR: Delivery server program 67
D.19 NOPAYOSMOD: Neword, Paymnt, Ordsts Transaction Program 67
D.20 NPOFEND: Front end CL program for nopayosmod 72
D.21 NPOSRVR: Neworder/Payment/Order Server program 72
D.22 STKFEND: Front end CL program for Stock Level 72
D.23 STKLVL: Stock Level Transaction program 73
D.24 STKSRVR: Stock Level server program 74
D.25 STRSRVR: Start server program 74

Appendix E. RTE Scripts 75
E.1 RTE Parameters 75

Appendix F. 180-Day DASD Requirements 76
F.1 IBM AS/400e server 740-2070--180-Day DASD Requirements 76
F.2 IBM AS/400e server 740-2070--Journal DASD Requirements 76

Appendix G. Third Party Quotes 77
G.1 Tuxedo Quote 77
G.1 Ethernet hubs quote 78

Appendix H. Auditor Letter 80

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 AS/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.*

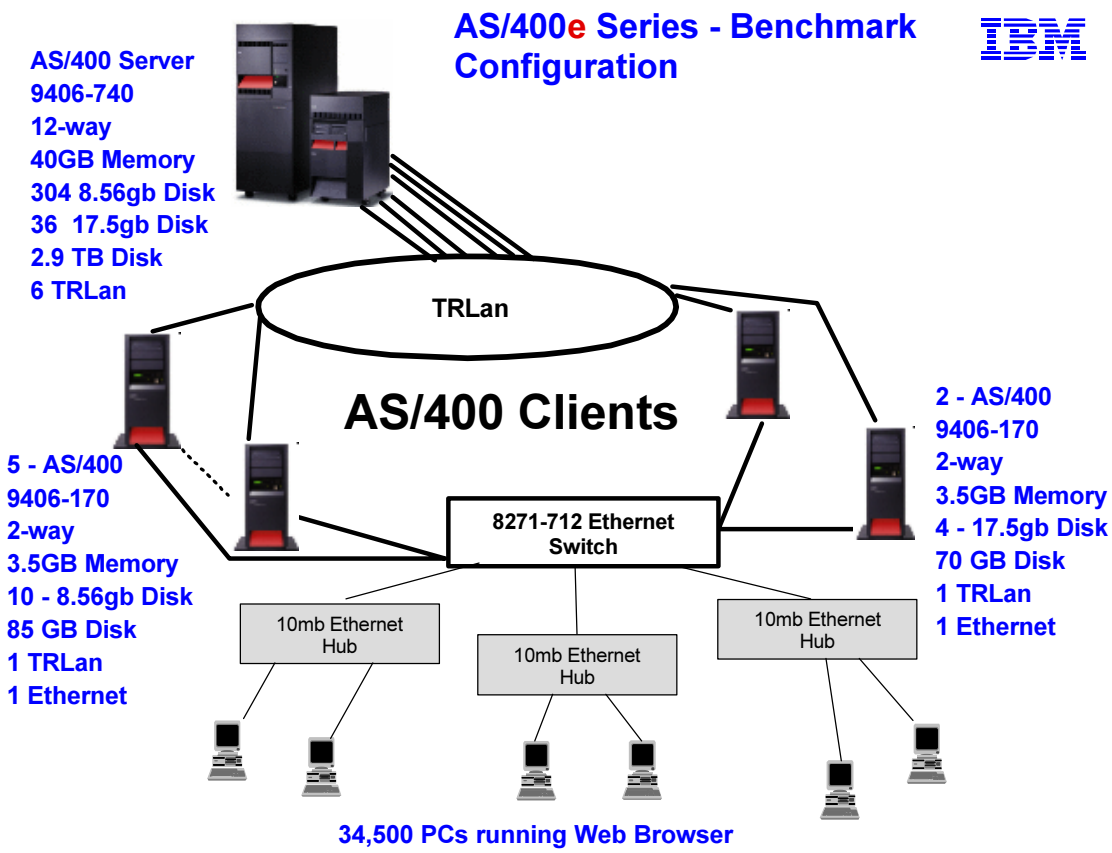
Appendix A contains the system, database, and application parameters changed from their default values used in these TPC Benchmark™ C tests.

1.4 Configuration Diagrams

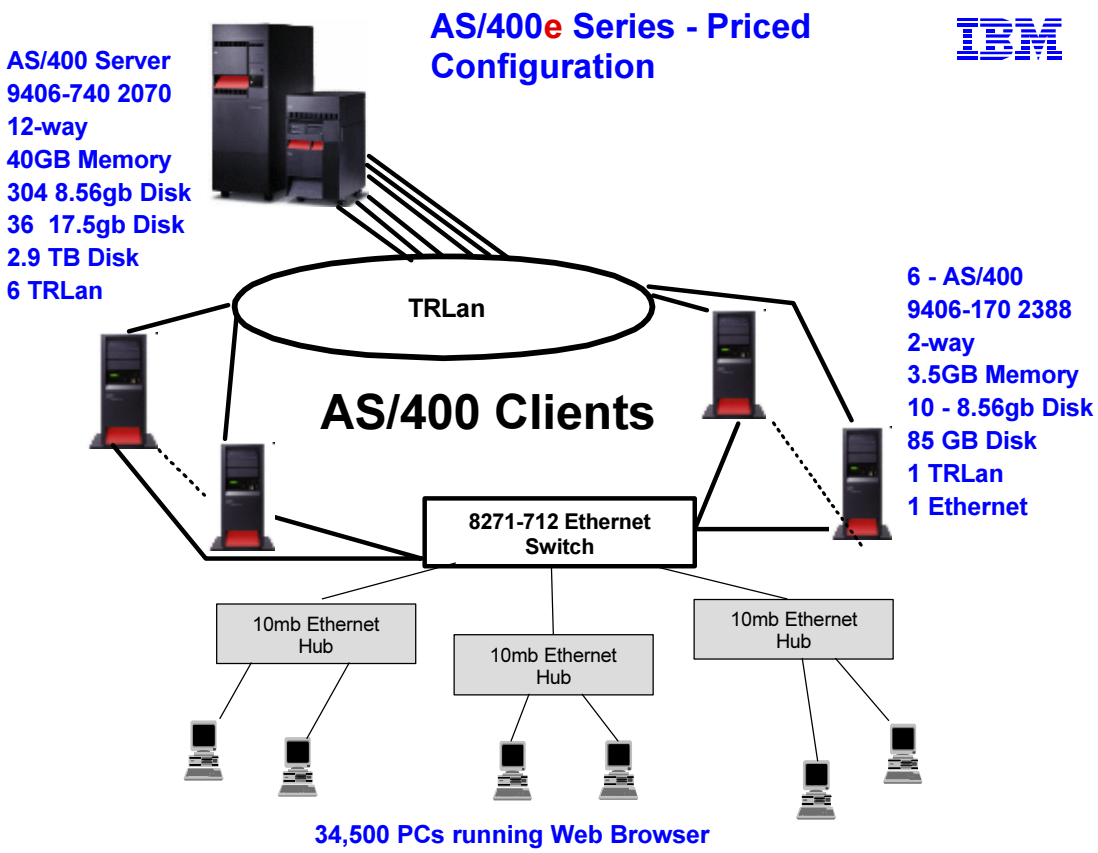
Diagrams of both measured and priced configurations must be provided, accompanied by a description of the differences. This includes, but is not limited to:

- *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 (eg, 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 (eg, DBMS, client processes, transaction monitors, software drivers, etc).*

1.5 AS/400e server 740-2070 Benchmark Configuration



1.6 AS/400e server 740-2070 C/S Priced Configuration



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 2,147,483,646 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.

For the benchmark test of the AS/400e server 740-2070, IBM did not implement horizontal or vertical partitioning upon any of the benchmark files.

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 PC desktop computer systems. All processing of the input/output screens was handled by the IBM AS/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 impleme nt the intelligent terminals is provided in Appendix A. All data manipulation was handled by the IBM AS/400.

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 10 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 10 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 10 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 10 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 10 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 10 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 10 shows the mix percentage for each of the transaction types executed by the SUT.

3.12 Queueing Mechanism of Delivery

The queueing 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 CICS transaction monitor.

Table 1. Numerical Quantities for Transaction and Terminal Profiles	
New Order	AS/400e server 740-2070
Percentage of Home order lines	99.01%
Percentage of Remote order lines	0.99%
Rolled Back Transactions	0.99%
Number of Items per order	10
Payment	
Percentage of Home transactions	84.86%
Percentage of Remote transactions	15.04%
Nonprimary Key Access	
Percentage of Payment using C_LAST	60.01%
Percentage of Order-Status using C_LAST	59.66%
Delivery	
Delivery transactions skipped	0
Transaction Mix	
New-Order	44.85%
Payment	43.09%
Order-Status	4.03%
Stock-Level	4.02%
Delivery	4.02%

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 selecta 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 selecta 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 = 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 = max(O_ID) = 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 = {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:

sum(O_OL_CNT) = {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 = 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 = 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 = sum(OL_AMOUNT) - 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(*) from ORDER) - (count(*)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: Starta 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: Performa 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 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: Starta New-Order transaction using the necessary inputs. The transaction is delayed just prior to the Commit.
3. Second terminal: Starta 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: Starta New-Order transaction using the necessary inputs to causea roll back (invalid item number). The transaction is delayed just prior to the rollback.
2. Second terminal: Starta 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: Starta Payment transaction fora 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: Starta Delivery transaction. The transaction is delayed just prior to the rollback.
2. Second terminal: Starta Payment transaction fora 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: Executea New Order transaction including itemsx and y.

2. First terminal:A New-Order transaction is started that contains itemx twice and itemy 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 itemsx 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 itemsx andy 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 the 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 the 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 AS/400 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 User ASP is protected by device parity protection (RAID-5) and contains the journal receiver (recovery log).

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 froma single disk unit in the User ASP was disconnected. Since the User 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. Processing of transactions continued without performance degradation.
4. Processing was allowed to continue for an additional 10 minutes.
5. An instantaneous power failure was then 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:

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. The DSTRCT database table was saved.
3. A full scale test was started on the SUT. The test was allowed to run for 10 minutes.
4. An unprotected disk in the System ASP was disconnected. This caused the system to halt.
5. The disk was reinstalled and the system restarted.

6. Restore the selected table saved in Step 2. This is equivalent to having losta disk unit and having to restore the system from a backup tape.
7. Step1 is performed returning SUM_2. SUM_2 is equal to SUM_1 returned in Step 1.
8. The OS/400 command APYJRNCHG (Apply Journal Changes) is executed applying all of the changes to the restored table for transactions that occurred after the table was saved.
9. Step1 is performed returning SUM_3.
10. The difference between SUM_3 and SUM_1 is verified to match the total number of successful New-Order transactions completed during the measurement.

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	AS/400e server 740-2070
WAREHOUSE	3,600
CUSTOMER	108,000,000
NEW-ORDER	32,400,000
DISTRICT	36,000
STOCK	360,000,000
ORDERS	108,000,000
ORDER-LINE	1,080,018,233
HISTORY	108,000,000
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 IBM AS/400 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 304-Disk ASP was used for system code, application code, and the database. A separate, fully mirrored, 36-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 IBM AS/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.

IBM did not implement horizontal or vertical partitioning for these TPC-C tests beyond the normal transparent system partitioning.

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 AS/400e server 740-2070.

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 AS/400e server 740-2070.

Table 3. AS/400e server 740-2070 Response, Keying, and Think Times

Response Times	NewOrder	Payment	Order Status	Delivery (int/def)	Stock Level	Menus
90%	0.40	0.30	0.40	0.200/9.13	1.00	0.11
Average	0.28	0.20	0.30	0.111/4.13	0.47	0.11
Maximum	7.59	8.39	1.38	0.657/26.56	7.92	6.67
			Think Times			
Minimum	0.01	0.01	0.01	0.01	0.01	N/A
Average	12.05	12.03	10.06	5.04	5.10	N/A
Maximum	120.31	120.31	100.61	50.41	50.41	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	18.00	3.00	2.00	2.00	2.00	N/A

6.3 Client Substitution Table

The following table lists the new orders/user by each priced and non-priced clients, and also the average new orders/users for the set of priced clients and for the set of non-priced (substitute) clients.

Client Substitution table

System	CPU	Users	tpmC	Measurement New Orders	New Orders / user
			Priced Clients		
A	9406-170	6,000	7547.85	150,957	25.16
B	9406-170	6,000	7552.55	151,051	25.18
C	9406-170	6,000	7553.50	151,070	25.18
D	9406-170	6,000	7550.15	151,003	25.17
E	9406-170	4,500	5672.45	113,449	25.21
				Average --	25.18
			Non-Priced Clients		
F	9406-170	3,000	3770.05	75,401	25.13
G	9406-170	3,000	3772.60	75,452	25.15
				Average --	25.14
		34,500			

6.4 Response Time Frequency Distribution

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

Note: Transaction counts are shown at the lower end of the interval ie, a count of 200 at 0 indicates that there were 200 transactions with a response time between 0 and the next interval.

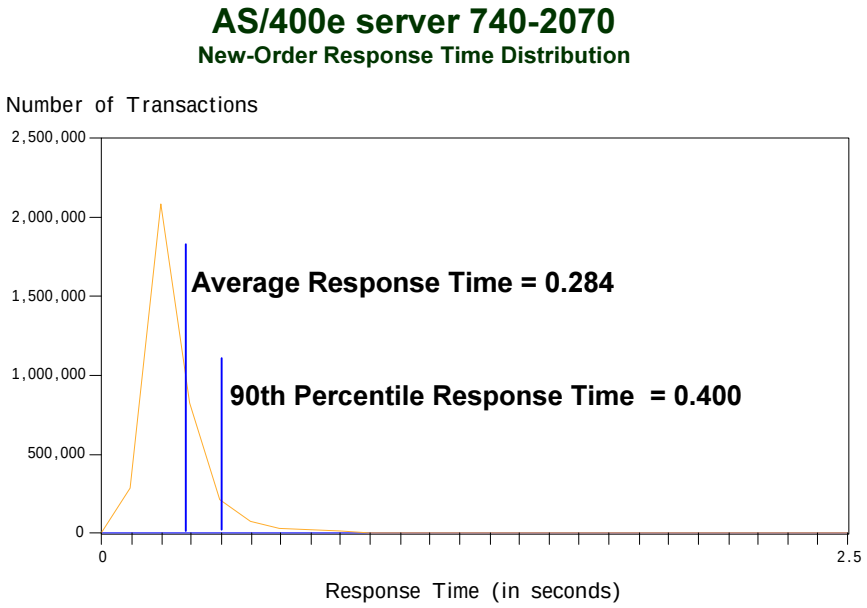


Figure 1. AS/400e server 740-2070 New-Order Response Time Distribution

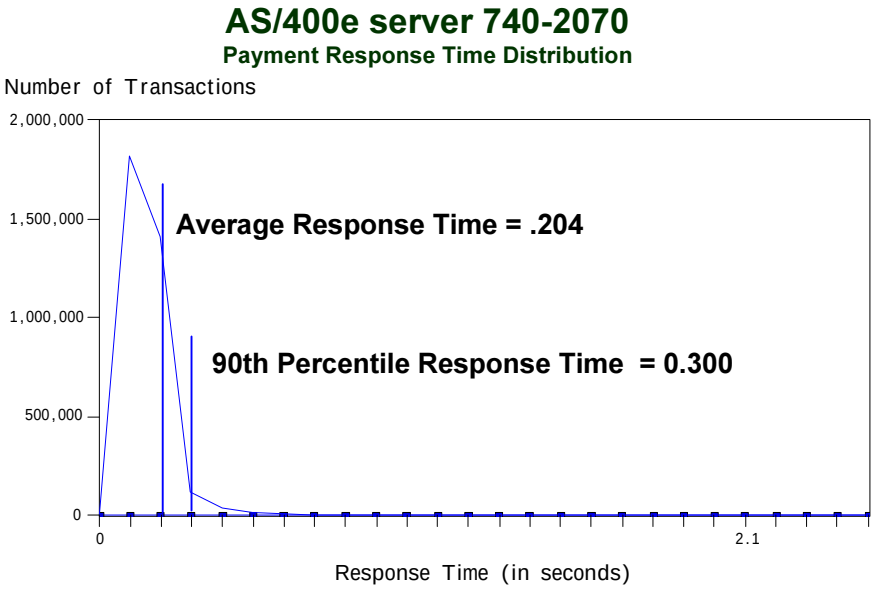


Figure 2. AS/400e server 740-2070 Payment Response Time Distribution

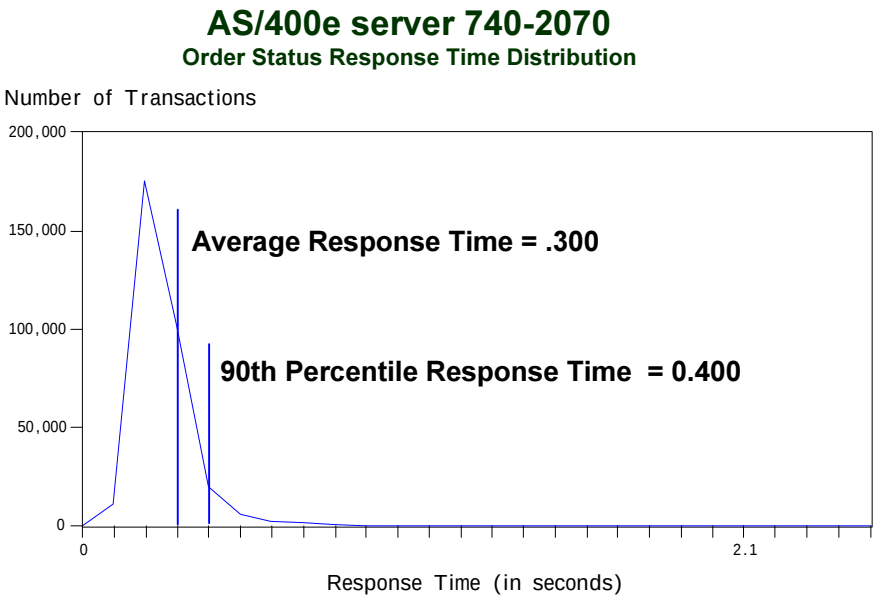


Figure 3. AS/400e server 740-2070 Order-Status Response Time Distribution

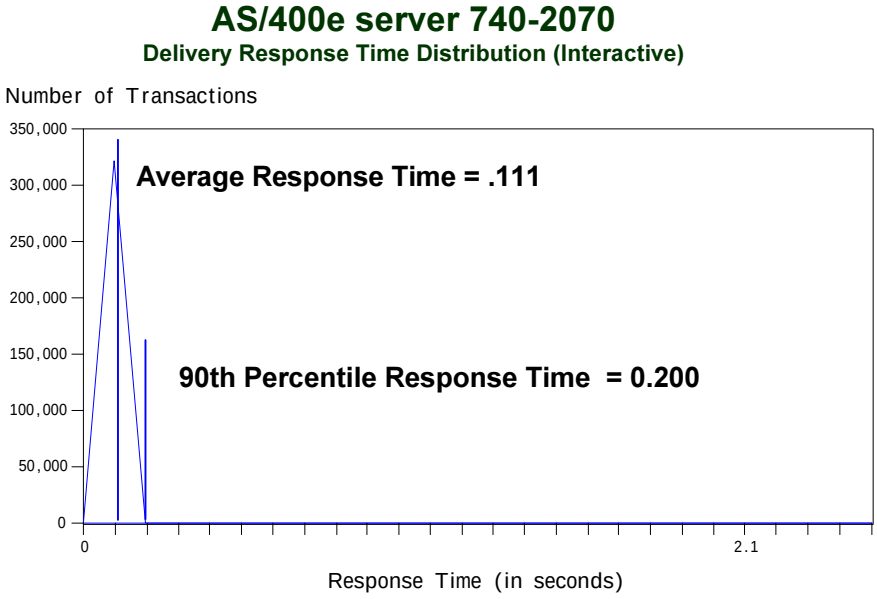


Figure 4. AS/400e server 740-2070 Delivery (Interactive) Response Time Distribution

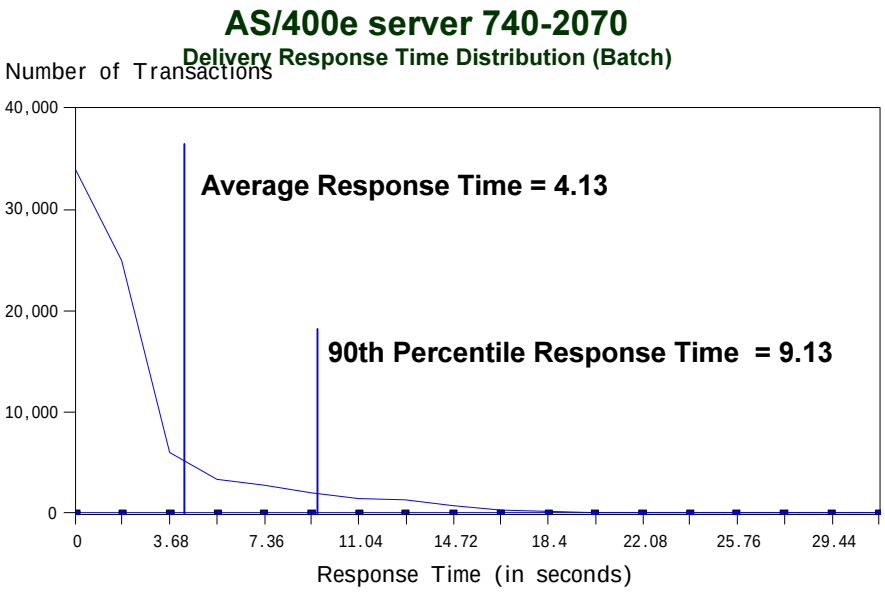


Figure 5. AS/400e server 740-2070 Delivery (Batch) Response Time Distribution

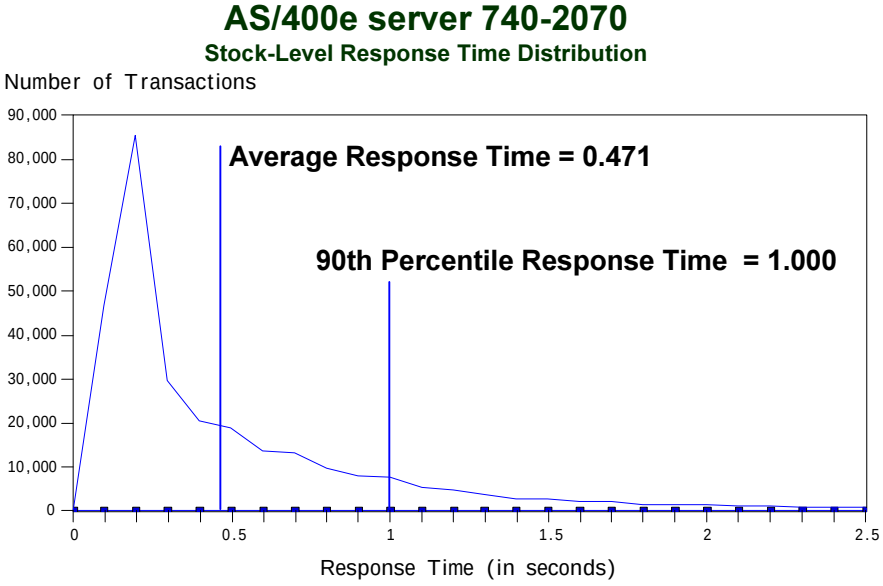


Figure 6. AS/400e server 740-2070 Stock-Level Response Time Distribution

6.5 Performance Curve for Response Time versus Throughput

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

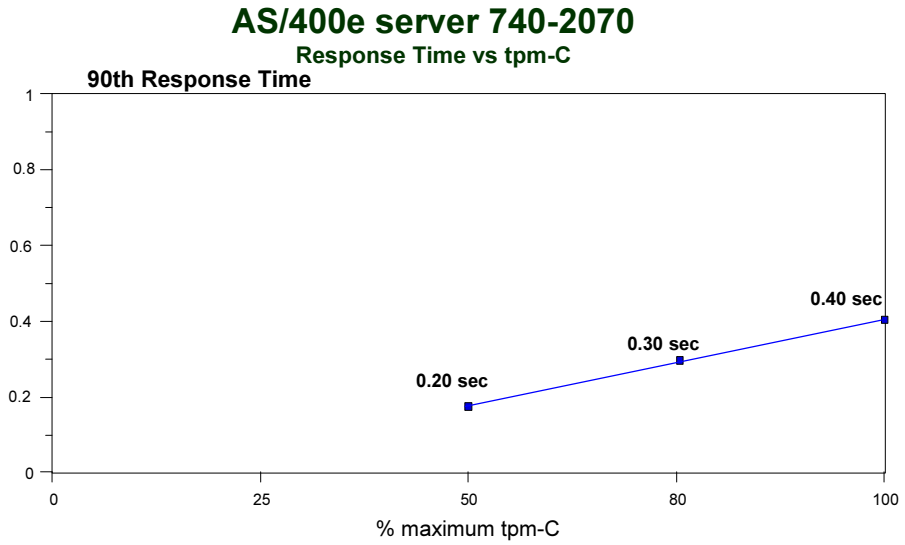


Figure 7. AS/400e server 740-2070 New-Order Response Time Versus Throughput

6.6 Think Time Frequency Distribution

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

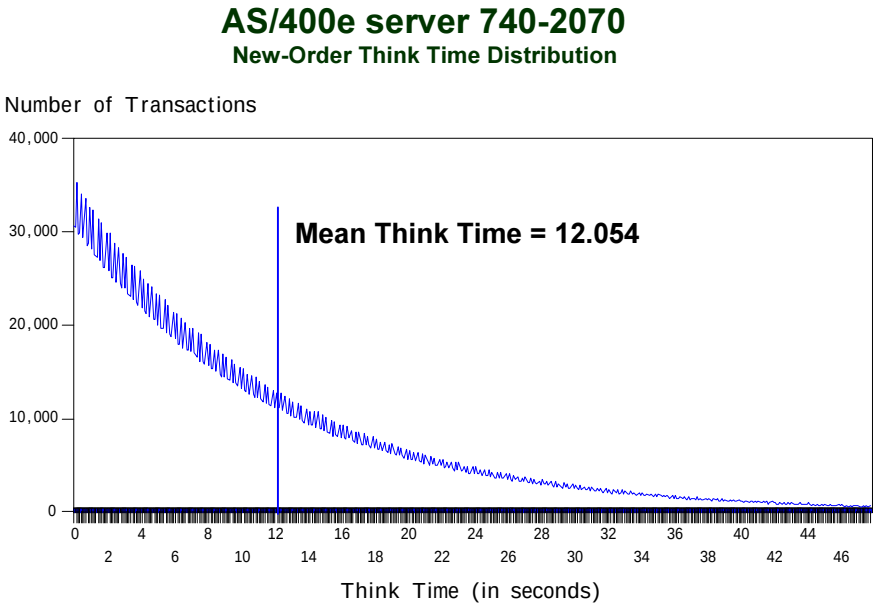


Figure 8. AS/400e server 740-2070 New-Order Think Time Distribution

6.7 Throughput Versus Elapsed Time

A graph of throughput versus elapsed time must be reported for each the New-Order transaction.

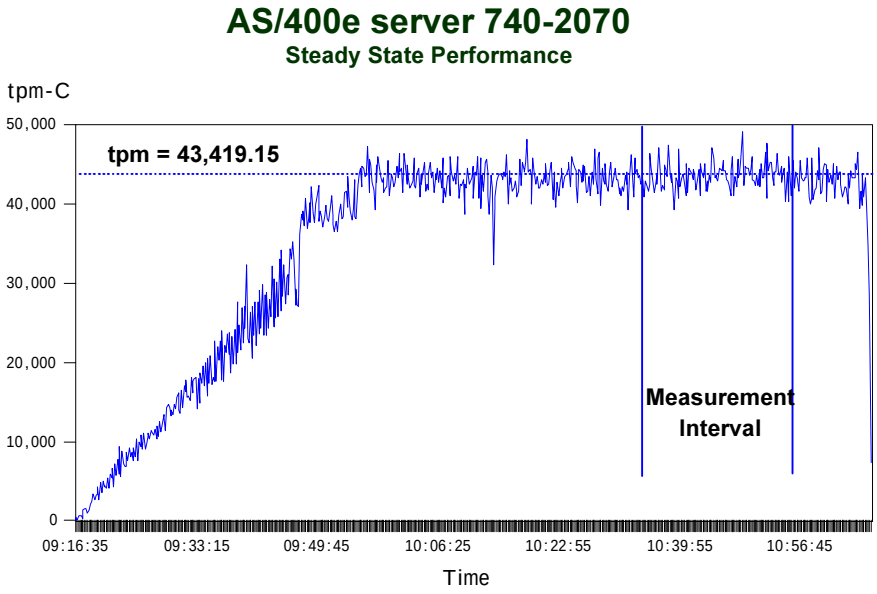


Figure 9. AS/400e server 740-2070 New-Order Throughput Versus Elapsed Time

6.8 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.

For each of the TPC Benchmark™ C transaction types the following steps are executed:

- At the database transaction start, a “start of commit cycle” entry is recorded in the journal.
- For each of the files updated by a transaction:
  - The database record being updated is locked by the transaction preventing further updates or reads until the journal records are written.
  - A before image of the record is written to the journal receiver.
  - An after image of the record is written to the journal receiver.
- At the end of the database transaction, the records are committed in the journal and all locks on the database records by the transaction are released.

Recording a block of journal entries does not correspond directly to a disk write, since the AS/400 journal management function has the ability to block journal writes from one or more jobs into a single physical I/O.

The AS/400 integrated relational database does not require an overt checkpointing system to ensure that data is written to disk. AS/400 standard journal management function ensures that data in disk files is synchronized with that in memory in a transparent, nondisruptive fashion.

As database I/Os are committed, the journal entries associated with the change are written prior to the completion of the commitment function. The database I/Os are issued as asynchronous I/Os which may be delayed by other requests to use the same data. To ensure that all data updates are completed in a reasonable period of time, AS/400 journal management ensures that unwritten pages from all tables being journaled are forced to disk at least once for every 50,000 entries to the journal. That is, for the 9 tables in TPC-C one file is forced to disk every 5,555 journal entries, so that all 9 are synchronized within the period of time it takes to log 50,000 journal entries.

The AS/400e server 740-2070 system operating at 43,419.15 tpm-C completes 50,000 entries to the journal in approximately 0.86 seconds. A minimum measurement interval of 20 minutes includes approximately 1036 complete cycles.

6.9 Reproducibility

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

A repeatability measurement was taken on the AS/400e server 740-2070 for the same length of time as the measured run. The repeatability interval was taken from the same measurement as the reported interval and the intervals were separated by 1minute and 53 seconds. The repeatability measurement was 43,387.05 tpmC.

6.10 Measurement Interval

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

The measurement interval for the tpmC reported was for 20 minutes in duration. The reproducibility measurement was also for a 20-minute interval.

7.0 Clause 6: SUT, Driver, and Communication Definition-Related Items

7.1 RTE Availability

If the RTE is commercially available, then its inputs must be specified. Otherwise, a description must be supplied of what inputs to the RTE had been used.

Appendix E contains the scripts used in the Remote Terminal Emulator testing.

7.2 Functionality and Performance of 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.

In the benchmark configuration the Remote Terminal Emulator (RTE) communicates with the client system over ethernet. The communications mechanism used in the benchmarked and priced configurations are the same.

7.3 Network Bandwidth

The bandwidth of the network(s) used in the tested/priced configuration must be disclosed.

RTE network used 10 Mb/second ethernet. The tested configuration of front-end AS/400s and RTEs was physically connected on a 10/100 ethernet switched network. The tested configuration of front-end AS/400s and server AS/400 was physically connected on a token ring network.

The priced configuration observed the physical limitation of 1024 physical devices connected to a single ethernet segment. The priced configuration observed the physical limitation of 260 physical devices connected to a single token ring.

7.4 Operator Intervention

If the configuration requires operator intervention, the mechanism and the frequency of this intervention must be disclosed.

The AS/400 configurations reported do not require any operator intervention to sustain the reported throughput during the 8-hour period.

8.0 Clause 7: Pricing - Related Items

8.1 Hardware and Programs Used

A detailed list of the 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 date. If package-pricing is used, contents of the package must be disclosed. Pricing source(s) and effective date(s) must also be reported.

The detailed list of all hardware and programs for the priced configuration is listed in the pricing sheets (please refer to Section 8.2 for details) for each system reported. The prices for all products and features are provided by IBM and available the same day as product or feature availability. All products are currently orderable for delivery on or before the published availability date.

8.2 Five Year Cost of System Configuration

The total 5-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 price sheet for the AS/400e server 740-2070 and associated client systems is contained on the following pages. The discounts used are disclosed below.

Revenue Allowance

This allowance of 20% was based on total hardware and software purchase price of the configuration.

Midrange Service Option (MRSO)

This discount is available for customers when agreement is reached for the customer to perform specified service duties (consult marketing representative for details). For the priced configuration, the MSAD discount is 17%.

Extended Maintenance Option (EMO)

This is a discount for prepayment of maintenance costs. A discount of 17% is available for this configuration based on payment for five years maintenance at time of purchase. This discount is applied to the balance after the Midrange Service Option is applied.

8.3 Statement of tpmC and Price/Performance

A statement of the measure tpmC, as well as the respective calculations for 5-year pricing, price/performance (price/tpmC), and the availability date must be disclosed.

The IBM AS/400e server 740-2070 was measured at 43,419.15 tpmC with a 5-year system price of $3,037,802. The respective price-performance for the AS/400e server 740-2070 is $69.96 per tpmC. The AS/400e server 740-2070 priced configuration is currently orderable for delivery on or after June 1, 1999.

8.3.1 IBM AS/400e server 740-2070 Five-Year System Price Configuration

AS/400e server model 740-2070

TPC-C REV 3.4

Report Date: June 3, 1999

AS/400 9406-740 Configuration

Item Description	Feature Number	Third Party Pricing	Unit Price	Qty.	Extended Price	Extended Maintenance	5-Year Price
Server Hardware:							
AS/400 9406 - Model 740 12-way Processor	2070		342,500	1	342,500	80,400	422,900
Token-Ring Adapter (9249)	Bundled			1	0		0
Battery Backup	Bundled			1	0		0
Twinax Work Station Controller (5540)	Bundled			1	0		0
2048 Optional MB Base Memory	8193		32,256	4	129,024		129,024
Optical Link	2688		2,000	6	12,000		12,000
Optical Bus Adapter	2695		1,000	1	1,000		1,000
2048 MB Main Storage	3193		36,864	16	589,824		589,824
Storage Expansion Unit	5057		5,000	1	5,000	10,560	15,560
Storage Expansion Unit	5058		5,000	10	50,000		50,000
1063 Mbps System Unit Expansion Tower	5073		14,900	1	14,900	105,600	120,500
1063 Mbps Storage Expansion Tower	5083		14,900	10	149,000	144,000	293,000
Token Ring Adapter	6149		1,200	6	7,200		7,200
RAID Disk Unit Controller	6533		9,900	20	198,000		198,000
2.5 GB 1/4-inch Tape Drive	6382		1,300	1	1,300		1,300
Lan/Wan/Workstation IOP	2629		2,600	6	15,600		15,600
Operations Console Cable	0328		125	1	125		125
V.24./EIA232 20 ft Cable	0330		125	1	125		125
17.2 GB Disk Unit	6714		3,000	36	108,000		108,000
8.58 GB Disk Unit	6713		1,650	303	499,950		499,950
Base 8.58 GB Disk Unit	8713		600	1	600		600
External V.34 modem	7852		525	1	525		525
Server Subtotal					*2,124,673*	*340,560*	*2,464,708*
Client Hardware:							
AS/400 9406-170 V4R4	2388		105,125	1	105,125	20,160	125,285
Twinax Work Station Controller (9720)	Bundled			1	0		0
Base Token Ring Adapter (9724)	Bundled			1	0		0
Tape Drive	6381		1,000	1	1,000		1,000
128 MB Main Storage	3002		1,280	2	2,560		2,560
256 MB Main Storage	3004		2,560	12	30,720		30,720
Opt Base 8.58 GB Disk Unit	8813		600	1	600		600
8.58 GB Disk Unit	6813		1,650	9	14,850		14,850
PCI Raid Disk Unit Ctlr	2740		4,000	1	4,000		4,000
PCI Lan/Wan/Workstation IOP	2809		1,800	1	1,800		1,800
PCI 100/10 Mbps Ethernet IOA	2838		900	1	900		900
32Mb IOP Memory (for IPCS)	2861		125	1	125		125
PCI Integrated Netfinity Server	2866		1,500	1	1,500		1,500
PCI System Exp Unit Base 16Mb	7101		3,000	1	3,000	2,640	5,640
External V.34 Modem	7852		525	1	525		525
Single Client Subtotal					*166,705*	*22,800*	*189,505*
Number of Clients				6			
Client Subtotal					*1,000,230*	*136,800*	*1,137,030*
Server Software:							
IBM Operating System/400 V4R3M0	Bundled			1	0		0
Client Software:							
IBM Operating System/400 V4R4M0	Bundled			6	0		0
BEA Tuxedo V6.4		1	3,000	6	18,000	14,400	32,400
ILE COBOL	5738-CB1		7,200	1	7,200		7,200
Application Development Toolkit	5716-PW1		7,850	1	7,850		7,850
DB2 Query Manager Dev. Toolkit	Bundled			1	0		0
ILE C	5716-CX2		7,200	1	7,200		7,200
Software Subtotal					*40,250*	*14,400*	*54,650*
User Connectivity:							
12-port 10/100 Ethernet Switch	8271-712		3,495	4	13,980	5,600	19,580
8-Port 10Mb Hub (10% spares)	Z99552	1	28	5449	150,120		150,120
8-port Token Ring connection unit (including 2 spares)	8226		545	4	2,180		2,180
User Connectivity Subtotal					*166,280*	*5,600*	*171,880*
5-year System Subtotal					*3,331,433*	*497,360*	*3,828,793*
Discounts:				**%Allowance**	**Volume**	**Discount**	
Revenue Allowance				20	3,181,313	(636,263)	
Mid-range System Option				17	497,360	(84,551)	
Extended Maintenance Option				17	412,809	(70,177)	

Notes:
Revenue Allowance is applied to hardware and software for the priced configuration.
EMO is a discount for prepayment of 5 years of maintenace costs.
MSRO is available to customers when agreement is reached for the customer to perform certain duties

Five-Year Cost of Ownership: $3,037,802

tpmC Rating: 43,419.15

$/tpmC: $69.96

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. Thank you.

9.0 Clause 8: Audit - Related Items

If the benchmark has been independently audited, then the auditor's name, address, phone number, and a brief audit summary report indicating compliance must be included in the Full Disclosure Report. A statement should be included, specifying when the complete audit report will become available and who to contact in order to obtain a copy.

These TPC Benchmark™ C results have been audited by Francis Raab of Information Paradigm. The attestation letter is included at the end of this report.

Appendix A. System Parameters and User Profile

A.1 System Parameters

Table 4 shows the system parameters changed for these TPC-C measurements.

Table 4. System Parameters

System Parameters	AS/400e server 740-2070
QMCHPOOL	2,857,504
*BASE	39,143,040
PAGING OPTION	USRDFN
QBASACTLVL	30,000
QPFRADJ	0
EDTRCYAP	*OFF

A.1.1 Transaction Subsystem Description

All transactions and associated user processes ran in the *BASE memory pool.

Appendix B: Database File Definitions

B.1 AREFFIL: Reference File

```
A*   OCO Source Materials
A* The Source code for this program is not published or otherwise
A* divested of its trade secrets, irrespective of what has been
A* deposited with the U.S. Copyright office
A* (C) Copyright IBM Corp. 1993, 1996, 1998

A*B:H2.AREFFIL:  Reference File
          R REFRCD                    TEXT('TPCC Reference File')
A*This file contains the field definition for most of the other
A*files used in the TPC-C benchmark.  For example to define the
A*warehouse ID field, both the warehouse file (WRHS) and the
A*district file (DSTRCT) reference WRHSID in this file.  This
A*helps when field definitions need to change, the change only
A*needs to be made in one place.
A*
A*UPDATED:10/23/93
A*Changes BALANC and CRDLMT to 12 digit fields as required
A*by the TPC-C V1.1 Specification
A*UPDATED:03/04/96
A*Changes BALANC and CRDLMT to 13 digit fields for effieiency.
A*Added ITMIMAGE perf v3.0 spec.  Converted many fields to binary.
A* wrhsid, distid, custid, itemid, ordid, oline, ytd2, qty2, qty4
A*
A*UPDATED:03/31/98
A*
A*
A*Code Identifiers
A* Note Binary nubers are 4B for two byte shorts and 9B
A* for long integer 32 byte numbers comforms to SQL defintions
A*
A            WRHSID         4B         TEXT('Warehouse ID')
A                                      COLHDG('W/H ID')
A            DISTID         4B         TEXT('District ID')
A                                      COLHDG('District')
A                                      EDTCDE(1)
A            CUSTID         9B         TEXT('Customer ID')
A                                      COLHDG('Customer')
A            ITEMID         9B         TEXT('Item ID')
A                                      COLHDG('Item')
A            ITMIMAGE       9B         TEXT('ImageID')
A                                      COLHDG('Image#')
A            ORDID          9B         TEXT('Order ID')
A                                      COLHDG('Order #')
A                                      EDTCDE(4)
A*                                     EDTWRD('       0')
A            OLINE          3P         TEXT('Order Line Number')
A                                      COLHDG('Order' 'Line #')
A*Name, Address and Descriptor Information
A            LOCNAM        10          TEXT('Name of W/H or District')
A                                      COLHDG('Location' 'Name')
A            FNAME         16          TEXT('First Name')
A                                      COLHDG('First' 'Name')
A            MINIT          2          TEXT('Middle Initial')
A                                      COLHDG('Middle' 'Init')
A            LNAME         16          TEXT('Last Name')
A                                      COLHDG('Last' 'Name')
A            ADDR1         20          TEXT('Address Line 1')
A                                      COLHDG('Address' 'Line 1')
A            ADDR2         20          TEXT('Address Line 2')
A                                      COLHDG('Address' 'Line 2')
A            CITY          20          TEXT('City')
A            STATE          2          TEXT('State')
A            ZIPCD          9A         TEXT('Zip Code')
A                                      COLHDG('Zip' 'Code')
A            PHONE         16          TEXT('Phone Number')
A            ITMNAM        24          TEXT('Item Name')
A                                      COLHDG('Item' 'Name')
A            CARRID         2          TEXT('Carrier ID')
A                                      COLHDG('Carrier' 'Number')
A            LOCAL          1  0       TEXT('Flag to indicate local')
A*Financial and Inventory Numbers
A            TAX            5  4       TEXT('Tax Percentage')
A                                      EDTWRD('0.    ')
A            DSCNT          5  4       TEXT('Discount Percentage')
A                                      COLHDG('Discount')
A                                      EDTWRD('0.    ')
A*UPDATED:03/01/96


  A            BALANC        13  2      TEXT('Balance Information')
A                                      COLHDG('Balance')
A                                      EDTWRD('        $0  .  -')
A            CREDIT         2          TEXT('Credit Status-GC or BC')
A                                      COLHDG('Credit' 'Status')
A*UPDATED:03/01/96
A            CRDLMT        13  2       TEXT('Credit Limit')
A                                      COLHDG('Credit' 'Limit')
A                                      EDTWRD('        $0  .  -')
A            AMOUNT7        7P 2       TEXT('Amount')
A                                      EDTWRD('  $0  .  -'  )
A            YTD           13  2       TEXT('YTD Amount')
A                                      EDTWRD('        $0  .  -'  )
A            YTD2           9B         TEXT('YTD Amount')
A                                      EDTCDE(4)
A            QTY2           3P 0       TEXT('Quantity')
A                                      COLHDG('Quantity')
A                                      EDTCDE(1)
A            QTY4           3P 0       TEXT('Quantity')
A                                      COLHDG('Quantity')
A                                      EDTCDE(1)
A            QTY3           3  0       TEXT('Quantity')
A                                      COLHDG('Quantity')
A                                      EDTCDE(1)
A            PRICE          5  2       TEXT('Price')
A                                      EDTWRD('$0  .  -')
A*Date and Time Information
A            TIMEDATE       8          TEXT('Internal Time Date')
A            DATE           8S         TEXT('Date')
A                                      EDTWRD('  -  -    ')
A            TIME           6S         TEXT('Time')
A                                      EDTWRD('  :  :  ')
A*Pad Information
A            CDATA        500          TEXT('Customer Data')
A                                      COLHDG('Cust' 'Filler')

A            HDATA         24          TEXT('History Information')
A                                      COLHDG('Hist' 'Filler')

A            IDATA         50          TEXT('Item Data')
A                                      COLHDG('Item' 'Filler')
A            DISTINFO      24          TEXT('Dist Info')
A                                      COLHDG('Dist' 'Info')
A*TPCC - Plus Information
A            CPHNTC        10          COLHDG('Phonetic Search')
A            DSPDID         2  0       TEXT('District ID')
A                                      COLHDG('District')
A                                      EDTCDE(4)
A            DSPOID         8  0       TEXT('Order ID')
A                                      COLHDG('Order #')
A                                      EDTCDE(4)
A            DSPOLN         2  0       TEXT('Order Line Number')
A                                      COLHDG('Order' 'Line #')
A            AMOUNT6        6  2       TEXT('Amount')
A                                      EDTWRD('$0   .  -')
A            DSPQTY         3S 0       TEXT('Quantity')
A                                      COLHDG('Quantity')
A                                      EDTCDE(1)
```

B.2 CSTMRLFCRT: Customer Logical File

```
A          R CSRCD                     PFILE(CSTMRPF)
A          K CWID
A          K CDID
A          K CLAST
A          K CFIRST
```

B.3 CSTMRLFNAM: Customer Names Logical File

```
A          R CSRCD                     PFILE(CSTMRPF)
A          K CWID
A          K CDID
A          K CLAST
```

B.4 CSTMR: Customer Logical File

```
A*This is the file definition for the customer data base file.
A*All fields that are stored in the customer file are defined
A*here, the actual field definitions are in AREFFIL (the
A*reference file) and are pulled into this file when it is
A*compiled (CRTPF command executed against this source).
A*The UNIQUE keyword guarantees that there will be not duplicate
A*key values inserted into the file.
A*
A                                      UNIQUE
A          R CSRCD                     PFILE(CSTMRPF)
A                                      TEXT('CUSTOMER MASTER FILE - TPCC')
A          K CID
A          K CDID
A          K CWID
```

B.5 CSTMRPF: Customer Physical File

```
  A                                    REF(AREFFIL)
A*This is the file definition for the customer data base file.
A*All fields that are stored in the customer file are defined
A*here, the actual field definitions are in AREFFIL (the
A*reference file) and are pulled into this file when it is
A*compiled (CRTPF command executed against this source).
A          R CSRCD                     TEXT('CUSTOMER MASTER FILE - TPCC')
A            CID       R               REFFLD(CUSTID)
A            CDID      R               REFFLD(DISTID)
A            CWID      R               REFFLD(WRHSID)
A            CFIRST    R               REFFLD(FNAME)
A            CINIT     R               REFFLD(MINIT)
A            CLAST     R               REFFLD(LNAME)
A            CLTOD     R               REFFLD(TIMEDATE)
A                                      COLHDG('Date of' 'Last Order')
A                                      TEXT('Date of DB Build')
A            CADDR1    R               REFFLD(ADDR1)
A            CCREDT    R               REFFLD(CREDIT)
A                                      COLHDG('Credit' 'Status')
A                                      TEXT('Credit Status')
A            CADDR2    R               REFFLD(ADDR2)
A            CDCT      R               REFFLD(DSCNT)
A            CCITY     R               REFFLD(CITY)
A            CSTATE    R               REFFLD(STATE)
A            CZIP      R               REFFLD(ZIPCD)
A            CPHONE    R               REFFLD(PHONE)
A            CBAL      R               REFFLD(BALANC)
A                                      TEXT('Customer Balance')
A            CCRDLM    R               REFFLD(CRDLMT)
A                                      TEXT('Credit Limit')
A            CYTD      R               REFFLD(YTD)
A                                      TEXT('Customer YTD')
A            CPAYCNT   R               REFFLD(QTY4)
```

```
A                                       TEXT('Customer Payments')
A            CDELCNT   R                REFFLD(QTY4)
A                                       TEXT('Customer Deliveries')
A*           CLTIME    R                REFFLD(TIME)
A*                                      COLHDG('Time of' 'Last Order')
A*                                      TEXT('Time of DB Build')
A            CDATA     R
```

B.6 DSTRCT: District File

```
                                        REF(AREFFIL)
     A*This is the file definition for the district data base file.
     A*All fields that are stored in the district file are defined
     A*here, the actual field definitions are in AREFFIL (the
     A*reference file) and are pulled into this file when it is
     A*compiled (CRTPF command executed against this source).
     A*The UNIQUE keyword guarantees that there will be not duplicate
     A*key values inserted into the file.
     A*
     A                                  UNIQUE
     A          R DSRCD                 TEXT('District Master File - TPCC')
     A            DID       R          REFFLD(DISTID)
     A            DWID      R          REFFLD(WRHSID)
     A            DNAME     R          REFFLD(LOCNAM)
     A                                  COLHDG('District' 'Name')
     A            DADDR1    R          REFFLD(ADDR1)
     A            DADDR2    R          REFFLD(ADDR2)
     A            DCITY     R          REFFLD(CITY)
     A            DSTATE    R          REFFLD(STATE)
     A            DZIP      R          REFFLD(ZIPCD)
     A            DTAX      R          REFFLD(TAX)
     A            DYTD      R          REFFLD(YTD)
     A                                  COLHDG('YTD' 'Balance')
     A                                  TEXT('YTD Balance')
     A            DNXTOR    R          REFFLD(ORDID)
     A                                  COLHDG('Next' 'Order #')
     A                                  TEXT('Next Order Number')
     A          K DID
     A          K DWID
```

HSTRY: History File

```
                                        REF(AREFFIL)
     A          R HSRCD                 TEXT('History File for TPCC')
     A*This is the file definition for the history data base file.
     A*All fields that are stored in the history file are defined
     A*here, the actual field definitions are in AREFFIL (the
     A*reference file) and are pulled into this file when it is
     A*compiled (CRTPF command executed against this source).
     A*The UNIQUE keyword is not needed in this file because there
     A*are no key fields for this file.
     A*
     A            HDID      R          REFFLD(DISTID)
     A            HWID      R          REFFLD(WRHSID)
     A            HCID      R          REFFLD(CUSTID)
     A            HCDID     R          REFFLD(DISTID)
     A            HCWID     R          REFFLD(WRHSID)
     A            HTOD      R          REFFLD(TIMEDATE)
     A                                  COLHDG('Payment' 'Time Date')
     A*           HTIME     R          REFFLD(TIME)
     A*                                 COLHDG('Payment' 'Time')
     A            HAMT      R          REFFLD(AMOUNT7)
     A                                  COLHDG('Payment' 'Amount')
     A                                  TEXT('Payment Amount')
     A            HDATA     R          REFFLD(HDATA)
```

ITEM: Item Physical File

```
     A                                  REF(AREFFIL)
     A*This is the file definition for the item data base file.
     A*All fields that are stored in the item file are defined
     A*here, the actual field definitions are in AREFFIL (the
     A*reference file) and are pulled into this file when it is
     A*compiled (CRTPF command executed against this source).
     A*The UNIQUE keyword guarantees that there will be not duplicate
     A*key values inserted into the file.
     A*
     A                                  UNIQUE
     A          R ITRCD                 TEXT('Item File for TPCC')
     A            IID       R          REFFLD(ITEMID)
     A            IMAGEID   R          REFFLD(ITMIMAGE)
     A            INAME     R          REFFLD(ITMNAM)
     A            IPRICE    R          REFFLD(PRICE)
     A            IDATA     R
     A          K IID
```

B.7 ITEMLF: Item Logical File

```
     A          R ITRCD   PFILE(ITEM)
     A          K IID
```

B.8 NEWORD: New Order Logical File

```
     A*This is the file definition for the new order data base file.
     A*All fields that are stored in the new order file are defined
     A*here, the actual field definitions are in AREFFIL (the
     A*reference file) and are pulled into this file when it is
     A*compiled (CRTPF command executed against this source).
     A*The UNIQUE keyword guarantees that there will not be duplicate
     A*key values inserted into the file.
     A*
     A                                  UNIQUE
     A          R NORCD                 PFILE(NEWORDPF)
                                        TEXT('New Orders File - TPCC')
     A          K NOWID
     A          K NODID
     A          K NOOID
```

B.9 NEWORDLF: New Order Logical File

```
     A          R NORCD                 PFILE(NEWORDPF)
     A          K NOWID
     A          K NODID
```

B.10 NEWORDPF: New Order Physical File

```
                                        REF(AREFFIL)
     A*This is the file definition for the new order data base file.
     A*All fields that are stored in the new order file are defined
     A*here, the actual field definitions are in AREFFIL (the
     A*reference file) and are pulled into this file when it is
     A*compiled (CRTPF command executed against this source).
     A          R NORCD                 TEXT('New Orders File - TPCC')
     A            NOOID     R          REFFLD(ORDID)
     A            NODID     R          REFFLD(DISTID)
     A            NOWID     R          REFFLD(WRHSID)
```

B.11 ORDERS: Orders Logical File

```
     A*B:H2.ORDERS:  Orders File
     A*This is the file definition for the orders data base file.
     A*All fields that are stored in the orders file are defined
     A*here, the actual field definitions are in AREFFIL (the
     A*reference file) and are pulled into this file when it is
     A*compiled (CRTPF command executed against this source).
     A          R ORRCD                 PFILE(ORDERSPF)
     A            OWID
     A            ODID
     A            OCID                  EDTWRD('0       ')
     A            OID
     A            OENTTOD
     A*           OENTTM
     A            OCARID
     A            OLINES
     A            OLOCAL
     A          K OWID
     A          K ODID
     A          K OCID
     A          K OID
```

B.12 ORDERSLF: Orders Logical File

```
     A                                  UNIQUE
     A          R ORRCD                 PFILE(ORDERSPF)
     A          K OWID
     A          K ODID
     A          K OID
```

B.13 ORDERSPF: Orders Physical File

```
                                        REF(AREFFIL)
     A*This is the file definition for the orders data base file.
     A*All fields that are stored in the orders file are defined
     A*here, the actual field definitions are in AREFFIL (the
     A*reference file) and are pulled into this file when it is
     A*compiled (CRTPF command executed against this source).
     A          R ORRCD                 TEXT('Orders Master File for TPCC')
     A            OWID      R          REFFLD(WRHSID)
     A            ODID      R          REFFLD(DISTID)
     A            OCID      R          REFFLD(CUSTID)
     A            OID       R          REFFLD(ORDID)
     A            OENTTOD   R          REFFLD(TIMEDATE)
     A                                  COLHDG('Order' 'Date')
     A                                  TEXT('Order Entry Time Date')
     A*           OENTTM    R          REFFLD(TIME)
     A*                                 COLHDG('Order' 'Time')
     A*                                 TEXT('Order Entry Time')
     A            OCARID    R          REFFLD(CARRID)
     A            OLINES    R          REFFLD(OLINE)
     A                                  COLHDG('Number of' 'Order Lines')
     A                                  TEXT('Number of Lines in Order')
     A            OLOCAL    R          REFFLD(LOCAL)
     A                                  TEXT('Flag to indicate local order')
```

B.14 ORDLIN: Order Lines Logical File

```
     A*This is the file definition for the orders data base file.
     A*All fields that are stored in the orders file are defined
     A*here, the actual field definitions are in AREFFIL (the
     A*reference file) and are pulled into this file when it is
```

```
A*compiled (CRTPF command executed against this source).
A          R ORRCD                     PFILE(ORDERSPF)
A            OWID
A            ODID
A            OCID                      EDTWRD('0       ')
A            OID
A            OENTTOD
A*           OENTTM
A            OCARID
A            OLINES
A            OLOCAL
A          K OWID
A          K ODID
A          K OCID
A          K OID
```

B.15 ORDLINLF: Order Lines Logical File

```
A          R OLRCD                     PFILE(ORDLINPF)
A          K OLWID
A          K OLDID
A          K OLOID
```

B.16 ORDERLINPF: Order Lines Physical File

```
A                                        REF(AREFFIL)
A          R OLRCD                     TEXT('Order Line File for TPCC')
A            OLOID     R               REFFLD(ORDID)
A            OLDID     R               REFFLD(DISTID)
A                                      TEXT('Customers District')
A            OLWID     R               REFFLD(WRHSID)
A                                      COLHDG('Customers' 'Warehouse')
A                                      TEXT('Customers Warehouse')
A            OLNBR     R               REFFLD(OLINE)
A                                      TEXT('Line Number of Order')
A            OLSPWH    R               REFFLD(WRHSID)
A                                      COLHDG('Supply' 'Warehouse')
A                                      TEXT('Supply Warehouse ID')
A            OLIID     R               REFFLD(ITEMID)
A                                      TEXT('Item Ordered')
A            OLQTY     R               REFFLD(QTY2)
A                                      COLHDG('Qty' 'Ordered')
A                                      TEXT('Quantity Ordered')
A            OLAMNT    R               REFFLD(AMOUNT7)
A                                      TEXT('Order Line Amount')
A            OLDLVTOD  R               REFFLD(TIMEDATE)
A                                      COLHDG('Delivery' 'Time Date')
A                                      TEXT('Delivery Time Date')
A*           OLDLVT    R               REFFLD(TIME)
A*                                     COLHDG('Delivery' 'Time')
A*                                     TEXT('Delivery Time')
A            OLDSTI    R               REFFLD(DISTINFO)
A                                      TEXT('Dist Information')
```

B.17 STOCK: Stock Logical File

```
A                                      UNIQUE
A          R STRCD                     PFILE(STOCKPF)
A          K STWID
A          K STIID
```

B.18 STOCKLF: Stock Logical File

```
A          R STRCD                     TEXT('Stock File for TPCC')
A                                      PFILE(STOCKPF)
A          K STWID
A          K STIID
```

B.19 STOCKPF: Stock Physical File

```
A*B:H2.STOCKPF:  Stock File
A                                      REF(AREFFIL)
A          R STRCD                     TEXT('Stock File for TPCC')
A            STWID     R               REFFLD(WRHSID)
A            STIID     R               REFFLD(ITEMID)
A                                      TEXT('Warhouse Nbr of Stock Item')
A            STQTY     R               REFFLD(QTY4)
A                                      COLHDG('Qty in' 'Stock')
A                                      TEXT('Quantity on hand')
A            STDI01    R               REFFLD(DISTINFO)
A                                      TEXT('Dist Info 1')
A            STDI02    R               REFFLD(DISTINFO)
A                                      TEXT('Dist Info 2')
A            STDI03    R               REFFLD(DISTINFO)
A                                      TEXT('Dist Info 3')
A            STDI04    R               REFFLD(DISTINFO)
A                                      TEXT('Dist Info 4')
A            STDI05    R               REFFLD(DISTINFO)
A                                      TEXT('Dist Info 5')
A            STDI06    R               REFFLD(DISTINFO)
A                                      TEXT('Dist Info 6')
A            STDI07    R               REFFLD(DISTINFO)
A                                      TEXT('Dist Info 7')
A            STDI08    R               REFFLD(DISTINFO)
A                                      TEXT('Dist Info 8')
A            STDI09    R               REFFLD(DISTINFO)
A                                      TEXT('Dist Info 9')
A            STDI10    R               REFFLD(DISTINFO)
A                                      TEXT('Dist Info 10')
A            STYTD     R               REFFLD(YTD2)
A                                      TEXT('Qty ordered YTD')
A            STORDRS   R               REFFLD(QTY4)
A                                      TEXT('# times ordered')
A            STREMORD  R               REFFLD(QTY4)
A                                      TEXT('# times ordered remotely')
A            STDATA    R               REFFLD(IDATA)
```

B.20 WRHS: Warehouse Physical File

```
                                       REF(AREFFIL)
A                                      UNIQUE
A          R WRRCD                     TEXT('Warehouse Master File - TPCC')
A            WID       R               REFFLD(WRHSID)
A            WNAME     R               REFFLD(LOCNAM)
A                                      COLHDG('W/H' 'Name')
A            WADDR1    R               REFFLD(ADDR1)
A            WADDR2    R               REFFLD(ADDR2)
A            WCITY     R               REFFLD(CITY)
A            WSTATE    R               REFFLD(STATE)
A            WZIP      R               REFFLD(ZIPCD)
A            WTAX      R               REFFLD(TAX)
A            WYTD      R               REFFLD(YTD)
A                                      COLHDG('YTD' 'Balance')
A                                      TEXT('Warehouse YTD Balance')
A          K WID
```

Appendix C. Database Build Programs

Program Flow For Build of Server, Client, and Database

This list shows the order of invocation of the database build programs. Each level of indentation is a call. If a program is indented more than the previous program, it was called by the previous program. If it is at the same margin as the previous program (or as earlier program), it was called by the same program as the previous (or earlier) program. To make determining the level of call easier, the level number will follow the program enclosed in parenthesis.

BLDTPCCPGM (2)
- CRTBLDPGMS (3)
  - NATBLD (4)
  - CRTHASH (4)
    - C_CREATE (5)
    - C_CREATE2 (5)
    - C_CREATE3 (5)
- DLTLOGICLS (3)
- LOADTPCCF (3)
  - LOADW_D (4)
  - LOADITEM (4)
  - LOADCST (4)
  - LOADORD (4)
- SETUP (3)
  - ORDERSVIEW (4)
  - ORDLINVIEW (4)
  - NEWORDVIEW (4)
  - CSTMRVIEW (4)
  - STRJRNTPCC (4)
- CRTENVPGMS (3)

BLDTPCCPGM: Database Build Programs

```
            PGM        PARM(&DB &ENV &CRTSAVFILE &DB2)
          /* This program brings in the number of warehouses and  */
          /* the options to compile the build programs and the    */
          /* environment programs.  The data library will get     */
          /* then get built, if it already exists, it is deleted  */
          /* and rebuilt.                                         */

          DCL        VAR(&WRHS) TYPE(*DEC) LEN(4 0)
          DCL        VAR(&DB) TYPE(*CHAR) LEN(1)
          DCL        VAR(&DB2) TYPE(*CHAR) LEN(1)
          DCL        VAR(&ENV) TYPE(*CHAR) LEN(1)
          DCL        VAR(&APPLIB) TYPE(*CHAR) LEN(10)
          DCL        VAR(&DATLIB) TYPE(*CHAR) LEN(10)
          DCL        VAR(&DB2SETPGM) TYPE(*CHAR) LEN(10)
          DCL        VAR(&LIBRARY) TYPE(*CHAR) LEN(10)
          DCL        VAR(&LIBASP) TYPE(*CHAR) LEN(1)
          DCL        VAR(&BLDLIB) TYPE(*CHAR) LEN(10)
          DCL        VAR(&CRTSAVFILE) TYPE(*CHAR) LEN(1)
          DCL        VAR(&SAVLIB) TYPE(*CHAR) LEN(10)
          DCL        VAR(&JRNLIB) TYPE(*CHAR) LEN(10)
          DCL        VAR(&WRHSCHAR) TYPE(*CHAR) LEN(4)
          DCL        VAR(&STRWHCHAR) TYPE(*CHAR) LEN(4)
          DCL        VAR(&ENDWHCHAR) TYPE(*CHAR) LEN(4)
          DCL        VAR(&ENDWRHS) TYPE(*CHAR) LEN(4)
          DCL        VAR(&STRWRHS) TYPE(*CHAR) LEN(4)
          DCL        VAR(&STRWH) TYPE(*DEC) LEN(4 0) VALUE(1)
          DCL        VAR(&ENDWH) TYPE(*DEC) LEN(4 0)
          DCL        VAR(&JRNASP) TYPE(*CHAR) LEN(1)
          DCL        VAR(&DATE) TYPE(*CHAR) LEN(6)
          DCL        VAR(&PRVLIBVAL) TYPE(*CHAR) LEN(10) /* +
                       Previous TPC-C Library Names */

          MONMSG     MSGID(CPF2556)

SETRPY:   ADDRPYLE   SEQNBR(111) MSGID(CPA7025) RPY(I)
          MONMSG     MSGID(CPF2555) EXEC(DO)
          RMVRPYLE   SEQNBR(111)
          RMVRPYLE   SEQNBR(112)
          RMVRPYLE   SEQNBR(113)
          RMVRPYLE   SEQNBR(114)
          RMVRPYLE   SEQNBR(115)
          ADDRPYLE   SEQNBR(111) MSGID(CPA7025) RPY(I)
          ENDDO
          ADDRPYLE   SEQNBR(112) MSGID(CPA1332) RPY(I)
          ADDRPYLE   SEQNBR(113) MSGID(CPF2110) RPY(I)
          ADDRPYLE   SEQNBR(114) MSGID(CPF1054) RPY(I)
          ADDRPYLE   SEQNBR(115) MSGID(CPF2105) RPY(I)
          CHGJOB     JOB(*) INQMSGRPY(*SYSRPYL)


GETPARMS: RTVDTAARA  DTAARA(TPCCINFO/DATALIB) RTNVAR(&DATLIB)
          RTVDTAARA  DTAARA(TPCCINFO/LIBASP) RTNVAR(&LIBASP)
          RTVDTAARA  DTAARA(TPCCINFO/APPLIB) RTNVAR(&APPLIB)
          RTVDTAARA  DTAARA(TPCCINFO/MNTLIB) RTNVAR(&BLDLIB)
          RTVDTAARA  DTAARA(TPCCINFO/SAVELIB) RTNVAR(&SAVLIB)
          RTVDTAARA  DTAARA(TPCCINFO/JRNLIB) RTNVAR(&JRNLIB)
          RTVDTAARA  DTAARA(TPCCINFO/JRNASP) RTNVAR(&JRNASP)
          RTVDTAARA  DTAARA(TPCCINFO/ENDWRHS) RTNVAR(&ENDWRHS)
          RTVDTAARA  DTAARA(TPCCINFO/STRWRHS) RTNVAR(&STRWRHS)
          CHGVAR     VAR(&WRHS) VALUE(&ENDWRHS)
          CHGVAR     VAR(&STRWH) VALUE(&STRWRHS)
          RTVSYSVAL  SYSVAL(QDATE) RTNVAR(&DATE)

          CHGVAR     VAR(&WRHSCHAR) VALUE(&WRHS)
          CHGVAR     VAR(&ENDWH) VALUE(&WRHS)
          CHGVAR     VAR(&STRWHCHAR) VALUE(&STRWH)
          CHGVAR     VAR(&ENDWHCHAR) VALUE(&ENDWH)

          MONMSG     MSGID(CPF0000)
          ENDSBS     SBS(TPCCBLDSBS) OPTION(*IMMED)
          MONMSG     MSGID(CPF1054)
          DLTSBSD    SBSD(&BLDLIB/TPCCBLDSBS)
          MONMSG     MSGID(CPF2105)
          DLTJOBD    JOBD(&BLDLIB/TPCBLDJOBD)
          MONMSG     MSGID(CPF2105)
          CRTSBSD    SBSD(&BLDLIB/TPCCBLDSBS) POOLS((1 *BASE)) +
                       TEXT('TPCCBLD Jobs Subsystem')
          DLTCLS     CLS(&BLDLIB/TPCCBLDCLS)
          MONMSG     MSGID(CPF2105)
          CRTCLS     CLS(&BLDLIB/TPCCBLDCLS) RUNPTY(21) +
                       TIMESLICE(2000) PURGE(*NO)
          DLTJOBQ    JOBQ(&BLDLIB/TPCBLDJOBQ)
          MONMSG     MSGID(CPF2105)
          CRTJOBQ    JOBQ(&BLDLIB/TPCBLDJOBQ)
          ADDJOBQE   SBSD(&BLDLIB/TPCCBLDSBS) +
                       JOBQ(&BLDLIB/TPCBLDJOBQ) MAXACT(*NOMAX)
          CRTJOBD    JOBD(&BLDLIB/TPCBLDJOBD) +
                       JOBQ(&BLDLIB/TPCBLDJOBQ) RTGDTA(TPCCBLD) +
                       INLLIBL(&BLDLIB QTEMP QGPL) LOG(4 +
                       0 *SECLVL) INQMSGRPY(*DFT)
          ADDRTGE    SBSD(&BLDLIB/TPCCBLDSBS) SEQNBR(99) +
                       CMPVAL(*ANY) PGM(QCMD) +
                       CLS(&BLDLIB/TPCCBLDCLS)
          STRSBS     SBSD(&BLDLIB/TPCCBLDSBS)
CHKDBLIB: IF (&DB *EQ 'Y') THEN(DO)
          CHKOBJ     OBJ(&DATLIB) OBJTYPE(*LIB)
          MONMSG     MSGID(CPF9801) EXEC(CRTLIB LIB(&DATLIB) +
                       ASP(&LIBASP))
          ENDJRNPF   FILE(*ALL) JRN(&DATLIB/TPCCJRN)
          MONMSG     MSGID(CPF0000)
          IF (&STRWRHS *EQ '1') THEN(DO)
          CLRLIB     LIB(&DATLIB)
          CHGLIB     LIB(&DATLIB) TEXT('Data library ' *CAT &DATE)
          ENDDO
          IF (&STRWRHS *NE '1') THEN(GOTO EXTENDDB)
          ENDDO
CHKOBJLIB: ADDLIBLE   LIB(&APPLIB)
          MONMSG     MSGID(CPF2103)
          MONMSG     MSGID(CPF2110) EXEC(DO)
          CRTLIB     LIB(&APPLIB) TEXT('Application Library ' +
                       *CAT &DATE)
            ADDLIBLE   LIB(&APPLIB)
          ENDDO

          SNDPGMMSG  MSGID(CPF9898) MSGF(QCPFMSG) +
                       MSGDTA('Starting Build') +
                       TOPGMQ(*EXT) MSGTYPE(*STATUS)


          /* Check for DB Build */

DBCHECK:  IF         COND(&DB *NE 'Y') THEN(GOTO CMDLBL(SKIPDB))
          CRTCLPGM   PGM(&BLDLIB/CRTBLDPGMS) +
                       SRCFILE(&BLDLIB/QCLSRC) +
                       SRCMBR(CRTBLDPGMS) OPTION(*NOSOURCE)

          CALL       PGM(&BLDLIB/CRTBLDPGMS) PARM(&STRWH &ENDWH +
                       &BLDLIB &DATLIB)

          CALL       PGM(&BLDLIB/CRTHASHES)

          GOTO       CMDLBL(SBMTPCCF)

EXTENDDB: ADDLIBLE   LIB(&BLDLIB)
          MONMSG     MSGID(CPF2103)

/* DELETE THE LOGICALS TO SPEED UP THE BUILD                        */

          CRTCLPGM   PGM(&BLDLIB/DLTLOGICLS) +
                       SRCFILE(&BLDLIB/QCLSRC) OPTION(*NOSOURCE)

          CALL       PGM(&BLDLIB/DLTLOGICLS) PARM(&DATLIB)

/*                FILL DATA FILES                                   */
 SBMTPCCF: CALL       PGM(&BLDLIB/LOADTPCCF) PARM(&DATLIB &BLDLIB +
                       &STRWHCHAR &WRHSCHAR)

/*   INITIALIZE APPLICATION ENVIRONMENT                             */

          IF         COND(&DB2 *EQ 'y') THEN(DO)
          RTVDTAARA  DTAARA(TPCCINFO/DB2SETPGM) RTNVAR(&DB2SETPGM)
          RTVDTAARA  DTAARA(TPCCINFO/LIBRARY) RTNVAR(&LIBRARY)
          SBMJOB     CMD(CALL PGM(&LIBRARY/DB2SETPGM) PARM(&BLDLIB +
                       &BLDLIB &DATLIB &CRTSAVFILE &SAVLIB +
                       &JRNLIB &JRNASP &STRWHCHAR &ENDWHCHAR)) +
                       JOB(SETUP) JOBQ(&BLDLIB/TPCBLDJOBQ)
          ENDDO
          ELSE       CMD(DO)
          SBMJOB     CMD(CALL PGM(&BLDLIB/SETUP) PARM(&BLDLIB +
                       &BLDLIB &DATLIB &CRTSAVFILE &SAVLIB +
                       &JRNLIB &JRNASP &STRWHCHAR &ENDWHCHAR)) +
                       JOB(SETUP) JOBQ(&BLDLIB/TPCBLDJOBQ)
          ENDDO
          /* Check for compiling of Environment programs. */
 SKIPDB:  IF         COND(&ENV = 'Y') THEN(DO)

          CHGLIB     LIB(&APPLIB) TEXT('Applications Created on ' +
                       *CAT (&DATE))
          CRTCLPGM   PGM(&BLDLIB/CRTENVPGMS) +
                       SRCFILE(&BLDLIB/QCLSRC) +
                       SRCMBR(CRTENVPGMS) OPTION(*NOSOURCE)
          MONMSG     MSGID(CPF0000) EXEC(SNDPGMMSG MSG('The +
                       application programs could not be created'))
          SBMJOB     CMD(CALL PGM(&BLDLIB/CRTENVPGMS) +
                       PARM(&APPLIB &DATLIB)) JOB(CRTENVPGMS) +
                       JOBQ(QCTL)
          MONMSG     MSGID(CPF0000) EXEC(SNDPGMMSG MSG('The +
                       application programs could not be created'))
          ENDDO

 ENDPGM:  ENDPGM
```

CRTBLDPGMS: Create Database Build Programs

```
          PGM        PARM(&STRWH &ENDWH &BLDLIB &DATLIB)
          DCL        VAR(&STRWH) TYPE(*DEC) LEN(4 0)
          DCL        VAR(&ENDWH) TYPE(*DEC) LEN(4 0)
          DCL        VAR(&BLDLIB) TYPE(*CHAR) LEN(10)
          DCL        VAR(&DATLIB) TYPE(*CHAR) LEN(10)
          DCL        VAR(&VERSION) TYPE(*CHAR) LEN(1)
          DCL        VAR(&VERSIONNUM) TYPE(*DEC) LEN(1 0)
          DCL        VAR(&MSGID) TYPE(*CHAR) LEN(7)
/* This CL program compiles all of the CL, C and CBL programs   */
/* that are needed to do a complete rebuild of the database.    */
/* They are compiled in the order they are needed. None of the  */
/* programs are called from this program, only compiled.        */

          SNDPGMMSG  MSGID(CPF9898) MSGF(QCPFMSG) +
                       MSGDTA('Compiling Build Programs.') +
                       TOPGMQ(*EXT) MSGTYPE(*STATUS)
          CHKOBJ     OBJ(&BLDLIB/TPCCBLDMSG) OBJTYPE(*MSGQ)
          MONMSG     MSGID(CPF9801) EXEC(CRTMSGQ +
                       MSGQ(&BLDLIB/TPCCBLDMSG))
          CLRMSGQ    MSGQ(TPCCBLDMSG)

          CRTCLPGM   PGM(&BLDLIB/NATBLD) SRCFILE(&BLDLIB/QCLSRC) +
                       SRCMBR(NATBLD)

          CRTCLPGM   PGM(&BLDLIB/SETUP) SRCFILE(&BLDLIB/QCLSRC)

          CRTCLPGM   PGM(&BLDLIB/ITEMVIEW) SRCFILE(&BLDLIB/QCLSRC)

          CRTCLPGM   PGM(&BLDLIB/CSTMRVIEW) SRCFILE(&BLDLIB/QCLSRC)

          CRTCLPGM   PGM(&BLDLIB/NEWORDVIEW) SRCFILE(&BLDLIB/QCLSRC)

          CRTCLPGM   PGM(&BLDLIB/ORDERSVIEW) SRCFILE(&BLDLIB/QCLSRC)

          CRTCLPGM   PGM(&BLDLIB/ORDLINVIEW) SRCFILE(&BLDLIB/QCLSRC)

          CRTCLPGM   PGM(&BLDLIB/STOCKVIEW) SRCFILE(&BLDLIB/QCLSRC)

          CRTCLPGM   PGM(&BLDLIB/LOADTPCCF) SRCFILE(&BLDLIB/QCLSRC)

          /* Build Physical Files */
          /* Has to be called before CREATE of LOAD pgms */
          IF         COND(&STRWH *EQ 1) THEN(CALL +
                       PGM(&BLDLIB/NATBLD) PARM(&ENDWH &BLDLIB +
                       &DATLIB))

          ADDLIBLE &DATLIB
          MONMSG     MSGID(CPF0000)

          CRTBNDC    PGM(&BLDLIB/LOADW_D) SRCFILE(&BLDLIB/QCSRC)
          CRTBNDC    PGM(&BLDLIB/LOADITEM) SRCFILE(&BLDLIB/QCSRC)
          CRTBNDC    PGM(&BLDLIB/LOADITONLY) SRCFILE(&BLDLIB/QCSRC)
          CRTBNDC    PGM(&BLDLIB/LOADSTONLY) SRCFILE(&BLDLIB/QCSRC)
          CRTBNDC    PGM(&BLDLIB/LOADCST) SRCFILE(&BLDLIB/QCSRC)
          CRTBNDC    PGM(&BLDLIB/LOADORD) SRCFILE(&BLDLIB/QCSRC)

          ENDPGM
```

NATBLD: Create Physical and Logical Files

```
PGM
DCL        VAR(&NUMWHSE) TYPE(*DEC) LEN(4 0)
DCL        VAR(&ORDSIZ) TYPE(*DEC) LEN(9 0)
DCL        VAR(&ORDLINSIZ) TYPE(*DEC) LEN(10 0)
DCL        VAR(&NEWORDSIZ) TYPE(*DEC) LEN(9 0)
DCL        VAR(&DLVRYSIZ) TYPE(*DEC) LEN(7 0)
DCL        VAR(&DATE) TYPE(*CHAR) LEN(6)
DCL        VAR(&TPCCDBLIB) TYPE(*CHAR) LEN(10)
DCL        VAR(&TPCCSRCLIB) TYPE(*CHAR) LEN(10)
DCL        VAR(&DDSSRC)     TYPE(*CHAR) LEN(10) +
             VALUE(QDDSSRC)
DCL        VAR(&TPCCPGMLIB) TYPE(*CHAR) LEN(10)
DCL        VAR(&VERSION)    TYPE(*CHAR) LEN(2)
RTVDTAARA  DTAARA(&TPCCPGMLIB/TPCCDBLIB) +
             RTNVAR(&TPCCDBLIB)
RTVDTAARA  DTAARA(&TPCCDBLIB/TPCCWRHS) RTNVAR(&NUMWHSE)
RTVDTAARA  DTAARA(&TPCCDBLIB/TPCCDBVRS) RTNVAR(&VERSION)
RTVDTAARA  DTAARA(&TPCCSRCLIB/TPCCDBLIB) +
             RTNVAR(&TPCCDBLIB)
CHGVAR     VAR(&ORDSIZ) VALUE(&NUMWHSE * 36000)
CHGVAR     VAR(&ORDLINSIZ) VALUE(&NUMWHSE * 360000)
IF         COND(&ORDLINSIZ > 2147483646) THEN(CHGVAR +
             VAR(&ORDLINSIZ) VALUE(2147483646))
CHGVAR     VAR(&NEWORDSIZ) VALUE(&NUMWHSE * 11000)
CHGVAR     VAR(&DLVRYSIZ) VALUE(&NUMWHSE * 1000)
SNDPGMMSG  MSGID(CPF9898) MSGF(QCPFMSG) +
             MSGDTA('Creating Data File Library ' *CAT +
             &TPCCDBLIB) TOPGMQ(*EXT) MSGTYPE(*STATUS)
RTVSYSVAL  SYSVAL(QDATE) RTNVAR(&DATE)
CRTLIB     LIB(&TPCCDBLIB) TEXT('TPC-C Data Files Built ' +
             *CAT (&DATE))
MONMSG     MSGID(CPF0000)
ADDLIBLE   LIB(&TPCCDBLIB)
MONMSG     MSGID(CPF2103) EXEC(DO)
RMVLIBLE   LIB(&TPCCDBLIB)
ADDLIBLE   LIB(&TPCCDBLIB)
ENDDO
SNDPGMMSG  MSGID(CPF9898) MSGF(QCPFMSG) +
             MSGDTA('Creating the Physical Files.') +
             TOPGMQ(*EXT) MSGTYPE(*STATUS)
CHKOBJ     OBJ(&TPCCSRCLIB/MAINICFF) OBJTYPE(*FILE)
CHKOBJ     OBJ(&TPCCSRCLIB/DLVRICFF) OBJTYPE(*FILE)
CHKOBJ     OBJ(&TPCCSRCLIB/DLVRICFF) OBJTYPE(*FILE)

DLTF       FILE(&TPCCSRCLIB/MAINICFF)
DLTF       FILE(&TPCCSRCLIB/DLVRICFF)
DLTF       FILE(&TPCCSRCLIB/STOCICFF)

CRTICFF    FILE(&TPCCSRCLIB/MAINICFF) +
             SRCFILE(&TPCCSRCLIB/QDDSSRC) +
             ACQPGMDEV(PGMDEVR) LVLCHK(*NO)
ADDICFDEVE FILE(&TPCCSRCLIB/MAINICFF) PGMDEV(PGMDEVR) +
             RMTLOCNAME(*REQUESTER)

CRTICFF    FILE(&TPCCSRCLIB/NEWOICFF) +
             SRCFILE(&TPCCSRCLIB/QDDSSRC) +
             ACQPGMDEV(PGMDEVR) LVLCHK(*NO)
ADDICFDEVE FILE(&TPCCSRCLIB/NEWOICFF) PGMDEV(PGMDEVR) +
             RMTLOCNAME(*REQUESTER)

CRTICFF    FILE(&TPCCSRCLIB/PAYMICFF) +
             SRCFILE(&TPCCSRCLIB/QDDSSRC) +
             ACQPGMDEV(PGMDEVR) LVLCHK(*NO)
ADDICFDEVE FILE(&TPCCSRCLIB/PAYMICFF) PGMDEV(PGMDEVR) +
             RMTLOCNAME(*REQUESTER)

CRTICFF    FILE(&TPCCSRCLIB/ORDSICFF) +
             SRCFILE(&TPCCSRCLIB/QDDSSRC) +
             ACQPGMDEV(PGMDEVR) LVLCHK(*NO)
ADDICFDEVE FILE(&TPCCSRCLIB/ORDSICFF) PGMDEV(PGMDEVR) +
             RMTLOCNAME(*REQUESTER)

CRTICFF    FILE(&TPCCSRCLIB/DLVRICFF) +
             SRCFILE(&TPCCSRCLIB/QDDSSRC) +
             ACQPGMDEV(PGMDEVR) LVLCHK(*NO)
ADDICFDEVE FILE(&TPCCSRCLIB/DLVRICFF) PGMDEV(PGMDEVR) +
             RMTLOCNAME(*REQUESTER)

CRTICFF    FILE(&TPCCSRCLIB/STOCICFF) +
             SRCFILE(&TPCCSRCLIB/QDDSSRC) +
             ACQPGMDEV(PGMDEVR) LVLCHK(*NO)
ADDICFDEVE FILE(&TPCCSRCLIB/STOCICFF) PGMDEV(PGMDEVR) +
             RMTLOCNAME(*REQUESTER)
IF  COND(&VERSION *NE '30') THEN( +
DO)
  IF  COND(&VERSION *EQ '31') THEN( +
  CHGVAR VAR(&DDSSRC) VALUE(QDDSSRCV31))
  ELSE CMD( +
  DO)
    IF  COND(&VERSION *EQ '32') THEN( +
      CHGVAR VAR(&DDSSRC) VALUE(QDDSSRCV32))
    ELSE CMD( +
      CHGVAR VAR(&DDSSRC) VALUE(QDDSSRCV40))
  ENDDO
ENDDO

CRTPF      FILE(&TPCCDBLIB/AREFFIL) +
             SRCFILE(&TPCCSRCLIB/&DDSSRC) WAITRCD(*NOMAX)  +
             LVLCHK(*NO)

CRTPF      FILE(&TPCCDBLIB/NEWORDPF) +
             SRCFILE(&TPCCSRCLIB/QDDSSRC) SIZE(&NEWORDSIZ  +
             30000 30000) ALLOCATE(*YES) +
             WAITRCD(*NOMAX) SHARE(*YES) LVLCHK(*NO)

CRTPF      FILE(&TPCCDBLIB/WRHS) SRCFILE(&TPCCSRCLIB/QDDSSRC) +
             SIZE(*NOMAX) WAITRCD(*NOMAX) SHARE(*YES) +
             LVLCHK(*NO)

CRTPF      FILE(&TPCCDBLIB/STOCKPF) +
             SRCFILE(&TPCCSRCLIB/QDDSSRC) SIZE(*NOMAX) +
             WAITRCD(*NOMAX) SHARE(*YES) LVLCHK(*NO)

CRTPF      FILE(&TPCCDBLIB/ORDLINPF) +
             SRCFILE(&TPCCSRCLIB/&DDSSRC) +
             SIZE(&ORDLINSIZ 30000 25000) +
             ALLOCATE(*YES) WAITRCD(*NOMAX) +
             SHARE(*YES) LVLCHK(*NO)

CRTPF      FILE(&TPCCDBLIB/ITEM) SRCFILE(&TPCCSRCLIB/QDDSSRC) +
             SIZE(*NOMAX) WAITRCD(*NOMAX) SHARE(*YES) +
             LVLCHK(*NO)

CRTPF      FILE(&TPCCDBLIB/HSTRY) SRCFILE(&TPCCSRCLIB/&DDSSRC) +
             SIZE(&ORDSIZ 30000 30000) ALLOCATE(*YES) +
             WAITRCD(*NOMAX) SHARE(*YES) LVLCHK(*NO)

CRTPF      FILE(&TPCCDBLIB/DSTRCT) +
             SRCFILE(&TPCCSRCLIB/QDDSSRC) SIZE(*NOMAX) +
             WAITRCD(*NOMAX) SHARE(*YES) LVLCHK(*NO)

CRTPF      FILE(&TPCCDBLIB/CSTMRPF) +
             SRCFILE(&TPCCSRCLIB/&DDSSRC) SIZE(*NOMAX) +
             WAITRCD(*NOMAX) SHARE(*YES) LVLCHK(*NO)

CRTPF      FILE(&TPCCDBLIB/ORDERSPF) +
             SRCFILE(&TPCCSRCLIB/&DDSSRC) SIZE(&ORDSIZ +
             30000 30000) ALLOCATE(*YES) +
             WAITRCD(*NOMAX) SHARE(*YES) LVLCHK(*NO)

CRTPF      FILE(&TPCCDBLIB/DLVRYLOG) +
             SRCFILE(&TPCCSRCLIB/&DDSSRC) SIZE(&DLVRYSIZ +
             10000 10000) ALLOCATE(*YES) +
             WAITRCD(*NOMAX) SHARE(*YES) LVLCHK(*NO)

CRTPF      FILE(&TPCCDBLIB/DLVRYLOGA) +
             SRCFILE(&TPCCSRCLIB/QDDSSRC) SIZE(*NOMAX) +
             WAITRCD(*NOMAX) SHARE(*YES) LVLCHK(*NO)


CRTPRTF    FILE(&TPCCDBLIB/DBUGPRT) +
             SRCFILE(&TPCCSRCLIB/&DDSSRC) LVLCHK(*NO)

CRTPRTF    FILE(&TPCCDBLIB/DBUGPRT2) +
             SRCFILE(&TPCCSRCLIB/&DDSSRC) LVLCHK(*NO)

SNDPGMMSG  MSGID(CPF9898) MSGF(QCPFMSG) +
             MSGDTA('Physical Files created.') +
             TOPGMQ(*EXT) MSGTYPE(*STATUS)


ENDPGM
```

CRTHASH: Create Hashes

```
PGM        PARM(&DATLIB &BLDLIB)
DCL        VAR(&DATLIB) TYPE(*CHAR) LEN(10)
DCL        VAR(&BLDLIB) TYPE(*CHAR) LEN(10)

MONMSG CPF0000
ADDLIBLE   LIB(&BLDLIB)
MONMSG     MSGID(CPF2103)
ADDLIBLE   LIB(&DATLIB)
MONMSG     MSGID(CPF2103)
DLTF       FILE(QGPL/HASHES)
MONMSG     MSGID(CPF0000)
DLTPGM     PGM(QGPL/CSTMR)
DLTPGM     PGM(QGPL/ITEMLF)
DLTPGM     PGM(QGPL/STOCK)
DLTUSRSPC  USRSPC(QGPL/HASH)
CRTSRCPF   FILE(QGPL/HASHES) MBR(HASHES)
CRTCMOD    MODULE(&BLDLIB/C_CREATE) +
             SRCFILE(&BLDLIB/QCSRC) OUTPUT(*PRINT) +
             OPTIMIZE(*FULL) DBGVIEW(*SOURCE)
CRTPGM     PGM(&BLDLIB/C_CREATE) +
             MODULE(&BLDLIB/C_CREATE) +
             BNDSRVPGM(QSYS/QDBCRTHA) ACTGRP(*CALLER)
CRTCMOD    MODULE(&BLDLIB/C_CREATE2) +
             SRCFILE(&BLDLIB/QCSRC) OUTPUT(*PRINT) +
             OPTIMIZE(*FULL) DBGVIEW(*SOURCE)
CRTPGM     PGM(&BLDLIB/C_CREATE2) +
             MODULE(&BLDLIB/C_CREATE2) +
             BNDSRVPGM(QSYS/QDBCRTHA) ACTGRP(*CALLER)
CRTCMOD    MODULE(&BLDLIB/C_CREATE3) +
             SRCFILE(&BLDLIB/QCSRC) OUTPUT(*PRINT) +
             OPTIMIZE(*FULL) DBGVIEW(*SOURCE)
CRTPGM     PGM(&BLDLIB/C_CREATE3) +
             MODULE(&BLDLIB/C_CREATE3) +
             BNDSRVPGM(QSYS/QDBCRTHA) ACTGRP(*CALLER)
SBMJOB     CMD(CALL PGM(&BLDLIB/C_CREATE)) JOBQ(QCTL) +
             MSGQ(&BLDLIB/TPCCBLDMSG)
DLYJOB 5
RCVMSG     MSGQ(&BLDLIB/TPCCBLDMSG)
SBMJOB     CMD(CALL PGM(&BLDLIB/C_CREATE2)) JOBQ(QCTL) +
             MSGQ(&BLDLIB/TPCCBLDMSG)
DLYJOB 5
RCVMSG     MSGQ(&BLDLIB/TPCCBLDMSG)
SBMJOB     CMD(CALL PGM(&BLDLIB/C_CREATE3)) JOBQ(QCTL) +
             MSGQ(&BLDLIB/TPCCBLDMSG)
DLYJOB 5
RCVMSG     MSGQ(&BLDLIB/TPCCBLDMSG)

CLRLIB QRPLOBJ

ENDPGM
```

DLTLOGICLS: Delete Logical Files

```
PGM        PARM(&DATLIB)
DCL        VAR(&DATLIB) TYPE(*CHAR) LEN(10)
MONMSG     MSGID(CPF2105)

DLTF       FILE(&DATLIB/CSTMR)
DLTF       FILE(&DATLIB/CSTMRLFCRT)
DLTF       FILE(&DATLIB/CSTMRLFNAM)
DLTF       FILE(&DATLIB/NEWORD)
DLTF       FILE(&DATLIB/NEWORDLF)
DLTF       FILE(&DATLIB/ORDERS)
DLTF       FILE(&DATLIB/ORDERSLF)
DLTF       FILE(&DATLIB/ORDLIN)
DLTF       FILE(&DATLIB/ORDLINLF)
DLTF       FILE(&DATLIB/STOCK)
DLTF       FILE(&DATLIB/STOCKLF)
```

LOADTPCCF: Database Population Control Program

```
 INITTPCCF:  PGM        PARM(&DATALIB &OBJLIB &STRWHCHAR &WRHSCHAR)
             DCL        VAR(&DATALIB) TYPE(*CHAR) LEN(10)
             DCL        VAR(&OBJLIB) TYPE(*CHAR) LEN(10)
             DCL        VAR(&WRHSCHAR) TYPE(*CHAR) LEN(4)
             DCL        VAR(&STRWHCHAR) TYPE(*CHAR) LEN(4)
             DCL        VAR(&STRWRHS5) TYPE(*CHAR) LEN(5)
             DCL        VAR(&ENDWRHS5) TYPE(*CHAR) LEN(5)
             DCL        VAR(&NULLCH) TYPE(*CHAR) LEN(1) VALUE(X'00')

 /* This program submits jobs to call the C                        */
 /* programs that will fill the data base files.                   */
 /* The programs it submits are:                                   */
 /*   LOADW_D    - A C program that fills WRHS and DSTRCT files.   */
 /*   LOADITEM   - A C program that fills the ITEM & STOCK files.  */
 /*   LOADCST    - A C program that fills CSTMR and HSTRY files.   */
 /*   LOADORD    - A C program that fills ORDERS, ORDLIN & NEWORD. */
 /* This program does the overrides that determine how many        */
 /* records to write on each physical I/O.  Messages are sent to   */
 /* the display so the user can track the progress of the build.   */

             CRTMSGQ    MSGQ(&DATALIB/TPCCBLDMSG)
             MONMSG     MSGID(CPF2112)

             OVRDBF     FILE(WRHS) TOFILE(&DATALIB/WRHS) SHARE(*NO) +
                          SEQONLY(*YES 500) /* Blocking set to 500 +
                          records */
             OVRDBF     FILE(DSTRCT) TOFILE(&DATALIB/DSTRCT) +
                          SEQONLY(*YES 500) /* Blocking set to 500 +
                          records */
             OVRDBF     FILE(ITEM) TOFILE(&DATALIB/ITEM) +
                          SEQONLY(*YES 5000) /* Blocking set to +
                          5000 records */
             OVRDBF     FILE(STOCKPF) TOFILE(&DATALIB/STOCKPF) +
                          SEQONLY(*YES 5000) /* Blocking set to +
                          5000 records */
             OVRDBF     FILE(CSTMRPF) TOFILE(DUMMY/CSTMRPF) +
                          SEQONLY(*YES 1000) /* Blocking set to +
                          1000 records */
             OVRDBF     FILE(HSTRY) TOFILE(&DATALIB/HSTRY) +
                          SEQONLY(*YES 500) /* Blocking set to 500 +
                          records */
             OVRDBF     FILE(ORDERSPF) TOFILE(&DATALIB/ORDERSPF) +
                          SEQONLY(*YES 2000) /* Blocking set to +
                          2000 records */
             OVRDBF     FILE(NEWORDPF) TOFILE(&DATALIB/NEWORDPF) +
                          SEQONLY(*YES 2000) /* Blocking set to +
                          2000 records */
             OVRDBF     FILE(ORDLINPF) TOFILE(&DATALIB/ORDLINPF) +
                          SEQONLY(*YES 10000) /* Blocking set to +
                          10000 records */

             CHGVAR     VAR(&STRWRHS5) VALUE(&STRWHCHAR *CAT &NULLCH)
             CHGVAR     VAR(&ENDWRHS5) VALUE(&WRHSCHAR *CAT &NULLCH)

 /* Submit the job to call LOADW_D to fill WRHS and DSTRCT files    */

             SBMJOB     CMD(CALL PGM(&OBJLIB/LOADW_D) PARM(&STRWRHS5 +
                          &ENDWRHS5)) JOB(WDJOB) +
                        JOBQ(&OBJLIB/TPCBLDJOBQ) +
                        MSGQ(&DATALIB/TPCCBLDMSG)

 /* Submit the job to call LOADITEM to fill ITEM and STOCK files    */

             IF         COND(&STRWHCHAR *EQ '0001') THEN(DO)
             SBMJOB     CMD(CALL PGM(&OBJLIB/LOADITEM) PARM(&STRWRHS5 +
                          &ENDWRHS5)) JOB(ISJOB) +
                        JOBQ(&OBJLIB/TPCBLDJOBQ) +
                        MSGQ(&DATALIB/TPCCBLDMSG)
             ENDDO

 /* Submit the job to call LOADCST to fill CSTMR and HSTRY files    */

             SBMJOB     CMD(CALL PGM(&OBJLIB/LOADCST) PARM(&STRWRHS5 +
                          &ENDWRHS5)) JOB(CHJOB) +
                        JOBQ(&OBJLIB/TPCBLDJOBQ) +
                        MSGQ(&DATALIB/TPCCBLDMSG)

 /* Submit the job to call LOADORD to fill ORDERS, ORDLIN & NEWORD  */

             SBMJOB     CMD(CALL PGM(&OBJLIB/LOADORD) PARM(&STRWRHS5 +
                          &ENDWRHS5)) JOB(OONJOB) +
                        JOBQ(&OBJLIB/TPCBLDJOBQ) +
                        MSGQ(&DATALIB/TPCCBLDMSG)

 /* Wait for the files to be populated                              */


             RCVMSG     MSGQ(&DATALIB/TPCCBLDMSG) WAIT(*MAX)
             RCVMSG     MSGQ(&DATALIB/TPCCBLDMSG) WAIT(*MAX)
             RCVMSG     MSGQ(&DATALIB/TPCCBLDMSG) WAIT(*MAX)
             RCVMSG     MSGQ(&DATALIB/TPCCBLDMSG) WAIT(*MAX)

 /* Delete the overrides                                            */

             DLTOVR     FILE(WRHS)
             DLTOVR     FILE(DSTRCT)
             DLTOVR     FILE(ITEM)
             DLTOVR     FILE(STOCKPF)
             DLTOVR     FILE(ORDERSPF)
             DLTOVR     FILE(CSTMRPF)
             DLTOVR     FILE(HSTRY)
             DLTOVR     FILE(ORDLINPF)
             DLTOVR     FILE(NEWORDPF)



             ENDPGM
```

LOADW_D: Fill WRHS and DSTRCT files

```
/* This is the C program that fills the district file (DSTRCT).     */
  /*  */
/* Start include files */
#include <stdlib.h>
#include <errno.h>
#include <recio.h>
#include <string.h>
#include <xxfdbk.h>
#include <milib.h>
#include <micomput.h>
#include <decimal.h>
#include <time.h>
 /* */
#define RANDOM(a,b) ((rand() % (b-a+1))+a)
 /* */
/* MAPINC generates structures for the database files          */
#pragma mapinc("dstrct","*LIBL/dstrct(*all)","both","d _P",,"tpcc")
#include "dstrct"
#pragma mapinc("wrhs","*LIBL/wrhs(*all)","both", "d _P",,"tpcc")
#include "wrhs"
 /* */
  /* Prototypes */
 void LoadWare(int);
 void LoadDstrt(int, int);
 void CreateStr(int, int, char *);
 void CreateZip(char *);
 /* */
  _RFILE
         *Dstrct_File,
         *Wrhs_File;

 tpcc_DSRCD_both_t DRec;
 tpcc_WRRCD_both_t WRec;
  /* */
 static unsigned long int rand_next = 1; /* used in MaxRand routine */
 static char chAlpha[62] = "abcdefghijklmnopqrstuvwxyz0123456789";
 static char chUpAlpha[27] = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
 static char tempstr[81];

   /* there are 100 entries each of length 20 */
char  *CITY_ARR [] =
  {  "Saint_Paul          ", "Saint_Louis          ",   "Concord_Grapes      ",
     "Trenton_New_Jersey  ",  "Bismark_Doughnut     ",   "Cheyenne_Wyoming    ",
     "Juneau_Alaska       ",   "Honolulu_Hawaii     ",   "Phoenix_The_Big_Bird",
     "Topeka_Kansas       ",   "Montgomery          ",   "Elmwood_Illinois    ",
     "Madison_Dolly       ",   "Lansing_Michigan    ",   "Frankfurt_Germany   ",
     "Des_Moines_Iowa     ",
     "Lincoln_Abraham     ",
     "Pierre_South_Dakota ",
     "Dover_Cliffs        ",
     "Helena_Montana      ",
     "San_Antonio         ",
     "Saint_Petersburg    ",
     "Caribu_Maine        ",
     "Sault_Saint_Marie   ",
     "Los_Angeles         ",
     "Minneapolis_St_Paul ",
     "Dallas_Fort_Worth   ",
     "Kalamazoo_Michigan  ",
     "The_Land_Of_Oz      ",
     "Lexington_Kentucky  ",
     "Washington          ",
     "District_Of_Columbia",
     "New_Orleans         ",
     "Boise_Idaho         ",
     "Denver_Colorado     ",
     "Kensington_Maryland ",
     "Riverside_Ca        ",
     "Blue_Mountain       ",
     "New_York_City       ",
     "Nashville_Tennessee ",
     "Fayetteville        ",
     "Berkeley_California ",
     "Maryville_Missouri  ",
     "San_Francisco       ",
     "Indianapolis        ",
     "Saint_Joseph_Mo     ",
     "Rochester_Minnesota ",
     "Fort_Worth          ",
     "Holland_Michigan    ",
     "South_Orange_NJ     ",
     "Stillwater          ",
     "Eugene_Oregon       ",
     "Alexandria_Virginia ",
     "Salt_Lake_City      ",
     "Manitowoc_Wisconsin ",
     "Philadelphia        ",
     "Sioux_Falls         ",
     "West_Lafayette      ",
     "New_Brunswick_NJ    ",
     "Rochester_New_York  ",
     "Oakland_Ca          ",
     "Las_Vegas_Nevada    ",
     "Birmingham_Alabama  ",
     "Arkadelphia_Arkansas",
     "San_Juan_Puerto_Rico",
     "Omaha_Nebraska      ",
     "Walla_Walla         ",
     "Baraboo_Wisconsin   ",
     "Atlanta_Georgia     ",
     "Grand_Forks_ND      ",
     "Tempe_Arizona       ",
     "Knoxville_Tn        ",
     "Fort_Leavenworth    ",
     "Boston_Massachusetts",
     "Danville_Virginia   ",
     "Baltimore_Maryland  ",
     "New_Haven_Ct        ",
     "Claremont_California",
     "Otsego_Michigan     ",
     "Providence          ",
     "Jacksonville        ",
     "Columbia_SC         ",
     "London_England      ",
     "Paris_France        ",
     "Chicago_Illinois    ",
     "Albuquerque         ",
     "Raleigh_NC          ",
     "Kansas_City_Missouri",
     "Tacoma_Washington   ",
     "Oronoco_Mn          ",
     "Charleston_WV       ",
     "Newark_Delaware     ",
     "Burlington_Vermont  ",
     "Damariscotta_Maine  ",
     "Colorado_Springs_Co ",
     "Nacogdoches_Texas   ",
     "Barbourville_Ky     ",
     "North_Carolina      ",
     "Mary_Of_The_Woods   ",
     "Heston_And_Isleworth"    };

   /* There are 100 entries each of length 2.    */
char *state_arr[] =
   {  "AL", "AK", "AR", "AZ", "CA", "CO", "CT", "DE", "FL", "GA",
     "HI", "ID", "IL", "IN", "IA", "KS", "KY", "LA", "ME", "MD",
     "MA", "MI", "MN", "MS", "MO", "MT", "NE", "NV", "NH", "NM",
     "NJ", "NY", "NC", "ND", "OH", "OK", "OR", "PA", "RI", "SC",
     "SD", "TN", "TX", "UT", "VT", "VA", "WA", "WV", "WI", "WY",
```

```
    "AC", "AG", "AI", "AM", "AP", "AV", "AW", "CN", "CS", "DI",
    "DM", "DN", "DS", "ED", "EM", "EN", "HN", "HO", "IH", "IM",
    "IR", "IW", "JN", "KA", "KO", "LE", "LI", "NI", "NT", "NZ",
    "OC", "OM", "ON", "PR", "RA", "RO", "SK", "SM", "TC", "TM",
    "TY", "TV", "VI", "VN", "VW", "XT", "YK", "YN", "YW", "ZA"    };
/* */
int main(argc, argv)
register int argc; char **argv;
{
  short int i;
  short int iStrtWrhs = 1;
  short int iEndWrhs = 1;
  short int iCurWrhs = 1;
  short int iDist;
  int iRetCode = 0;
    /* Set up random number generator */
  time_t ltime;
  time(&ltime);
  srand(ltime);          /* seed the random num generator */
  rand_next = (unsigned long int)ltime;
      /* set up alpha seed for random chars */
  strcat(chAlpha, chUpAlpha);
    /* parse the input parms */
  for (argc--, argv++, i=0; argc>0; ++i, argc--, argv++)
  {
    if (i==0) {
      iStrtWrhs = atoi(*argv);
    }
    else {
      iEndWrhs = atoi(*argv);
    }
  }  /* end of parsing input parms */

       /*  */
  if( (Wrhs_File = _Ropen("wrhs","rr+")) == NULL)
  {
    printf("Open of Wrhs file failed\n");
    exit(99);
  }
  if( (Dstrct_File = _Ropen("dstrct","rr+")) == NULL)
  {
    printf("Open of District file failed\n");
    exit(99);
  }

  for (iCurWrhs=iStrtWrhs; iCurWrhs<=iEndWrhs; iCurWrhs++)
  {
    LoadWare(iCurWrhs);
    for (iDist=1; iDist<=10; iDist++)
    {
      LoadDstrt(iCurWrhs, iDist);
    }
  } /* of for (iCurWrhs=iStrtWrhs; iCurWrhs<=iEndWrhs; iCurWrhs++) */

 _Rclose(Dstrct_File);
 _Rclose(Wrhs_File);
 return(iRetCode);
} /* end of main */

/*****************************************************************/
/*   L o a d W a r e                                             */
/*      Create & write warehouse rec.                            */
/*                                                               */
/*  PARMS:                                                       */
/*     iWid => current warehouse id                              */
/*****************************************************************/
void LoadWare(iWid)
  short iWid;
{
  char curwhs[05] = "0001";
  char waddr1[21] = "1";
  char waddr2[21] = "Bldg 1";
  decimal(5,4) d;
  int iX;

  WRec.WID = iWid;
  sprintf(curwhs, "%04d", iWid);
      /* build wname */
  CreateStr(1, 5, tempstr);
  strcat(tempstr, curwhs);
  strcat(tempstr, "W");
  memset(WRec.WNAME, ' ', 10);
  strncpy(WRec.WNAME, tempstr, strlen(tempstr));
      /* build address1 */
  CreateStr(1, 10, waddr1);
  memset(tempstr, '\0', strlen(tempstr));
  strcat(tempstr,"1");
  strcat(tempstr, curwhs);
  strcat(tempstr, " Ave.");
  strcat(tempstr, waddr1);
  memset(WRec.WADDR1, ' ', 20);
  strncpy(WRec.WADDR1, tempstr, strlen(tempstr));
      /* build address2 */
  CreateStr(1, 10, waddr2);
  memset(tempstr, '\0', strlen(tempstr));
  strcat(tempstr,"Bldg 1");
  strcat(tempstr, curwhs);
  strcat(tempstr, waddr2);
  memset(WRec.WADDR2, ' ', 20);
  strncpy(WRec.WADDR2, tempstr, strlen(tempstr));
      /* build city */
  iX = RANDOM(0,99);
  memset(WRec.WCITY, ' ', 20);
  strncpy(WRec.WCITY, CITY_ARR[iX], strlen(CITY_ARR[0]));
      /* build state */
  strncpy(WRec.WSTATE, state_arr[iX], strlen(state_arr[0]));
      /* build zip code */
  CreateZip(tempstr);
  strncpy(WRec.WZIP, tempstr, strlen(tempstr));
      /* build tax */
  d = (RANDOM(0,2000))/10000.0;
  WRec.WTAX = d;
      /* build YTD balance */
  WRec.WYTD = 300000.00;

  _Rwrite(Wrhs_File,(void *)&WRec, sizeof(WRec));
  return;
} /* end of LoadWare */



/*****************************************************************/
/*   L o a d D s t r t                                           */
/*      Create & write district rec.                             */
/*                                                               */
/*  PARMS:                                                       */
/*     iWid => current warehouse id                              */
/*     uchDid => current district id                             */
/*****************************************************************/
void LoadDstrt(iWid, iCurDist)
  short int iWid;
  short int iCurDist;
{
  char curwhs[05] = "0001";
  char curdist[03] = "01";
  char daddr[21] = "1";
  char tstr[21];
  decimal(5,4) d;
  int iX;

  DRec.DWID = iWid;
  DRec.DID = iCurDist;
  sprintf(curwhs, "%04d", iWid);
  sprintf(curdist, "%02d", iCurDist);

      /* build dname */
  strcpy(tempstr, "D");
  strcat(tempstr, curwhs);
  strcat(tempstr, curdist);
  CreateStr(1, 4, tstr);
  strcat(tempstr, tstr);
  memset(DRec.DNAME, ' ', 10);
  strncpy(DRec.DNAME, tempstr, strlen(tempstr));
      /* build address1 */
  strcpy(daddr, curdist);
  iX = RANDOM(1,999);
  sprintf(tstr, "%03d", iX);
  strcat(daddr, tstr);
  strcat(daddr, " Ave.");
  CreateStr(1, 10, tempstr);
  strcat(daddr, tempstr);
  memset(DRec.DADDR1, ' ', 20);
  strncpy(DRec.DADDR1, daddr, strlen(daddr));
      /* build address2 */
  strcpy(daddr, "Bldg ");
  strcat(daddr, curdist);
  strcat(daddr, " ");
  CreateStr(1, 12, tempstr);
  strcat(daddr, tempstr);
  memset(DRec.DADDR2, ' ', 20);
  strncpy(DRec.DADDR2, daddr, strlen(daddr));
      /* build city */
  iX = RANDOM(0,99);
  memset(DRec.DCITY, ' ', 20);
  strncpy(DRec.DCITY, CITY_ARR[iX], strlen(CITY_ARR[0]));
      /* build state */
  strncpy(DRec.DSTATE, state_arr[iX], strlen(state_arr[0]));
      /* build zip code */
  CreateZip(tstr);
  strncpy(DRec.DZIP, tstr, strlen(tstr));

      /* build tax */
  d = (RANDOM(0,2000)) /10000.0;
  DRec.DTAX = d;
      /* build YTD balance */
  DRec.DYTD = 30000.00;
      /* build next order id */
  DRec.DNXTOR = 3001;

  _Rwrite(Dstrct_File,(void *)&DRec,  sizeof(DRec));
  return;
} /* end of LoadDstrt */


/*****************************************************************/
/*   C r e a t e S t r                                           */
/*      Create string of random alphanumeric characters.         */
/*                                                               */
/*  PARMS:                                                       */
/*     iMin => minimum length of string                          */
/*     iMax => maximum length of string                          */
/*     szTemp => address to store created string                 */
/*  If string is not variable length, then set iMin and iMax to  */
/*  the actual fixed length.                                     */
/*****************************************************************/
void CreateStr(iMin, iMax, szTemp)
  int iMin, iMax;
  char *szTemp;
{
  int iAlphalen,
      i, j;

  iAlphalen = RANDOM(iMin,iMax);   /* determine str len  */
  memset(szTemp, '\0', strlen(szTemp));
  for (j=0; j<iAlphalen; j++)
  {
    i = RANDOM(0,61);
    szTemp[j]= chAlpha[i];
  }
  return;
} /* end of CreateStr */

/*****************************************************************/
/*   C r e a t e Z i p                                           */
/*      Create zip code string.                                  */
/*                                                               */
/*  PARMS:                                                       */
/*    Returns zip code string.                                   */
/*****************************************************************/
void CreateZip(zipstr)
char *zipstr;
{
  int iZ;

  iZ = RANDOM(0,99999);
  sprintf(zipstr, "%05d", iZ);
  strcat(zipstr, "1111");
  return;
} /* end of CreateZip */


 /* Redefine max generated random num. Default is 32,767. */
#undef RAND_MAX
#define RAND_MAX 99999
 /*****************************************************************/
 /*   M a x R a n d                                               */
 /*      Create random number larger than 32767.                  */
 /*  NOTE: This is the rand() function used by AS/400 with one    */
 /*      slight modification.  In the rand() function, rand_next is */
 /*      shifted right by 16 rather than the 14 used here.  Shifting */
 /*      by 16 produces a max value of 65535.                     */
 /*                                                               */
 /*  PARMS:                                                       */
 /*    Returns zip code string.                                   */
 /*****************************************************************/
int MaxRand(void)
{
  int rand_temp;

  rand_next = rand_next * 1103515245 + 12345;
  rand_temp = (rand_next >> 14)%((unsigned long)(RAND_MAX)+1);
  return(rand_temp);

} /* end of CreateZip */
```

LOADITEM: Fill ITEM and STOCK Files

```
/* This is the C program that fills:                              */
/*   item file (ITEM)                                             */
/*   stock file (STOCK)                                           */
 /*  */
/* Start include files */
#include <stdlib.h>
#include <errno.h>
#include <recio.h>
#include <string.h>
#include <xxfdbk.h>
#include <milib.h>
#include <micomput.h>
#include <decimal.h>
#include <time.h>
 /* */
 #define MAXITEMS 100000
/* #define MAXITEMS 1000 */
#define RANDOM(a,b) ((rand() % (b-a+1))+a)
 /* */
/* MAPINC generates structures for the database files          */
#pragma mapinc("item","*LIBL/item(*all)","both","d _P",,"tpcc")
#include "item"
#pragma mapinc("stockpf","*LIBL/stockpf(*all)","both", "d _P",,"tpcc")
#include "stockpf"
 /* */
  /* Prototypes */
 void LoadItem(void);
 void LoadStock(int);
 void CreateStr(int, int, char *);
 /* */
  _RFILE
        *Item_File,
        *Stock_File;

 tpcc_ITRCD_both_t ItemRec;
 tpcc_STRCD_both_t StkRec;

   short int iStrtWrhs = 1;
   short int iEndWrhs = 1;
   short int iCurWrhs = 1;
 char chAlpha[62] = "abcdefghijklmnopqrstuvwxyz0123456789";
 char chUpAlpha[27] = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
 char tempstr[81];

   /* There are 3 sets of characters that are concatenated */
   /* to make the district information fields for Stock */
 char *DistInf1[] = {
   "abcdefghijk", "nowisthetim", "thisiswhere",
   "whatdoyouno", "notmuchthen", "whydoyouask",
   "Idonotknow_", "sowhereisit", "inthecities",
   "andthetowns"};

 char *DistInf2[] = {
   "01", "02", "03", "04", "05", "06", "07", "08", "09", "10"};

 char *DistInf3[] = {
   "mnopqrstuvw", "whydidyoudo", "interactive",
   "batchwork_", "systemview_", "performance",
   "Ineverwork_", "Ialwaysplay", "nexttolast_",
   "nowthefinal"};

   /* There are 100 entries, each of length 24 */
 char *ItemNameStr[]  = {
   "Ball_Point_Pen          ",   "Sailboat_Fuel_Tank      ",
   "Flowers_(Poppies)       ",   "VCR_Road_Warrior_Game   ",
   "Bird_House_Plans        ",   "Pushbutton_Telephone    ",
   "Surgical_Gloves         ",   "1_Gallon_Distilled_Water",
   "Black_Labrador_Puppies  ",   "Plain_Pockets_Pants     ",
   "Electric_Plane          ",   "Childs_Car_Seat         ",
   "Gray_Hooded_Sweatshirts ",   "Lone_Ranger_Costume     ",
   "Jiffy_Pain_Pills        ",   "Devil's_Food_Cake       ",
   "Full_Length_Fox_Fur     ",   "AM_FM_Stereo_Car_Radio  ",
   "Cheddar_Cheese_Ball     ",   "Elephant_Ear_Mushroom   ",
   "White_Traffic_Lane_Paint",   "Plastic_Garbage_Pail    ",
   "Four_Blade_Ceiling_Fans ",   "Doll_House_Furniture    ",
   "Apple_Core_Remover      ",   "Cola_Twelve_Pack        ",
   "Italian_Food_Cook_Book  ",   "Wool_Toe_Socks          ",
   "Feel_Good_Vitamins      ",   "100'_Rubber_Garden_Hose ",
   "Orange_Golf_Balls       ",   "Snorkle_And_Fins_Set    ",
   "IBM_Facsimile_Machine   ",   "5_lb._Bag_Flour         ",
   "Fleece_Lined_Wool_Gloves",   "Bermuda_Grass_Seed      ",
   "Front_And_Rear_Floor_Mat",   "Waiter_Costume          ",
   "Magical_Mystery_Maze    ",   "Red_Safety_Flares       ",
   "Silver_Push_Pins        ",   "QuartzDigital_Wristwatch",
   "10_lb_Test_Fishing_Line ",   "Blue_Pile_Carpeting     ",
   "Red_Brick_House         ",   "Bear_Claw_Doughnuts     ",
   "AS/400_Advanced_Series  ",   "12_Inch_Wooden_Ruler    ",
   "Square_Plastic_Calendar ",   "Twelve_Num_Two_Pencils  ",
   "Pad_Yellow_Legal_Paper  ",   "Change_Machine          ",
   "Street_Hockey_Balls     ",   "Half_Inch_3_Ring_Binders",
   "Sporting_Good_Catalog   ",   "Ten_Gallon_Hats         ",
   "Promotion_Notification  ",   "32_Gal._Keg_Beer        ",
   "Business_Cards          ",   "Pebble_Beach_Poster     ",
   "Texas_Residence_Forms   ",   "COBOL_Programmers_Guide ",
   "25_Inch_Color_TVs       ",   "Black_Magic_Marker      ",
   "Telephone_Books         ",   "Riding_Lawnmower        ",
   "$7500_Gift_Certificate  ",   "Compact_Disc_Player     ",
   "Four_Drawer_Cabinets    ",   "Dry-Erase_Markers       ",
   "Multi-System_Paper      ",   "Message_Waiting_Light   ",
   "Automotive_Care_Manuals ",   "Carton_Ceiling_Tiles    ",
   "12_Ounce_Styrofoam_Cup  ",   "Phone_Calling_Card      ",
   "Inflatable_Globe        ",   "Walnut_Magazine_Rack    ",
   "Over_Under_Shotgun      ",   "While_You_Were_Gone_Memo",
   "Three_Tea_Bags          ",   "Bowling_Pin_Spotter     ",
   "Golf_Club_Cleaner       ",   "Staple_Removers         ",
   "Downtown_Parking_Permit ",   "Sliding_Glass_Door_Locks",
   "Floral_Pattern_Lampshade",   "Junior_College_Books    ",
   "Two_Wheel_Bike          ",   "Zoo_Season_Pass         ",
   "Baseball_Tickets        ",   "Four_Person_Tent        ",
   "Cross_Country_Ski_Set   ",   "Motorcycle_Sidecars     ",
   "Fireplace_Tools         ",   "Radio_Controlled_Plane  ",
   "Oak_Wall_Clock          ",   "Rubber_Baby_Buggy_Wheel ",
   "Anti_Static_Strap       ",   "Old_Wooden_Toothpicks   " };


/* */
int main(argc, argv)
register int argc; char **argv;
{
  short int i;
  int iRetCode = 0;
  int item;

    /* Set up random number generator */
  time_t ltime;
  time(&ltime);
  srand(ltime);          /* seed the random num generator */
     /* set up alpha seed for random chars */
  strcat(chAlpha, chUpAlpha);
    /* parse the input parms */
  for (argc--, argv++, i=0; argc>0; ++i, argc--, argv++)
  {
    if (i==0) {
      iStrtWrhs = atoi(*argv);
    }
    else {
      iEndWrhs = atoi(*argv);
   /* strcpy(endwrhs, *argv);             */
   /* printf("endwrhs = %4s \n", endwrhs); */
    }
  }  /* end of parsing input parms */

       /*  */
  if( (Item_File = _Ropen("item","rr+")) == NULL)
  {
    printf("Open of Item file failed\n");
    exit(99);
  }
  if( (Stock_File = _Ropen("stockpf","rr+")) == NULL)
  {
    printf("Open of Stockpf file failed\n");
    exit(99);
  }
    /*************************/
    /*  Start of true build  */
    /*************************/
  LoadItem();
  for (item=1; item<=MAXITEMS; item++)
  {
    LoadStock(item);
  }

  _Rclose(Item_File);
  _Rclose(Stock_File);

} /* end of main */


/*****************************************************************/
/*   L o a d I t e m                                             */
/*      Create & write item records.                             */
/*                                                               */
/*  PARMS: none                                                  */
/*****************************************************************/
void LoadItem()
{
  decimal(5,2) d;
  int i, iX;

  for (i=1; i<=MAXITEMS; i++)
  {
    ItemRec.IID = i;
      /* build image id 1..10,000 */
    iX = RANDOM(1,10000);
    ItemRec.IMAGEID = iX;

      /* build iname */
    iX = RANDOM(0,99);
    memset(ItemRec.INAME, ' ', 24);
    strncpy(ItemRec.INAME, ItemNameStr[iX],
                           strlen(ItemNameStr[0]));

      /* build IPRICE */
    d = (RANDOM(100,10000)) / 100.00;
    ItemRec.IPRICE = d;
      /* Build IDATA random string 26 to 50 chars */
    CreateStr(26, 50, tempstr);
    if ((RANDOM(0,9)) == 0)
    {
      iX = (RANDOM(0,(strlen(tempstr)-8)));
      memcpy(&tempstr[iX], "ORIGINAL", 8);
    }
    memset(ItemRec.IDATA, ' ', 50);
    strncpy(ItemRec.IDATA, tempstr, strlen(tempstr));
    _Rwrite(Item_File,(void *)&ItemRec,  sizeof(ItemRec));
  } /* end of for */

  return;
} /* end of LoadItem */


/*****************************************************************/
/*   L o a d S t o c k                                           */
/*      Create & write stock records.                            */
/*                                                               */
/*  PARMS:                                                       */
/*     item => current item id                                   */
/*****************************************************************/
void LoadStock(item)
 int item;
{
  int i, iX;
  char StockDist[25] ;

  for (iCurWrhs=iStrtWrhs; iCurWrhs<=iEndWrhs; iCurWrhs++)
  {
      /* create STIID */
    StkRec.STIID = item;
      /* create STWID */
    StkRec.STWID = iCurWrhs;
      /* create STQTY between 10..100*/
    StkRec.STQTY = RANDOM(10,100);
     /* CREATES STDInn (district info) */
     iX = RANDOM(0, 9);
    memcpy(StockDist, DistInf1[iX], 11);
    memcpy((void *)&StockDist[11], DistInf2[0], 2);
     iX = RANDOM(0, 9);
    memcpy((void *)&StockDist[13], DistInf3[iX], 11);
    strncpy(StkRec.STDI01, StockDist, 24);
     iX = RANDOM(0, 9);
    memcpy(StockDist, DistInf1[iX], 11);
    memcpy((void *)&StockDist[11], DistInf2[1], 2);
     iX = RANDOM(0, 9);
    memcpy((void *)&StockDist[13], DistInf3[iX], 11);
    strncpy(StkRec.STDI02, StockDist, 24);
     iX = RANDOM(0, 9);
    memcpy(StockDist, DistInf1[iX], 11);
    memcpy((void *)&StockDist[11], DistInf2[2], 2);
     iX = RANDOM(0, 9);
    memcpy((void *)&StockDist[13], DistInf3[iX], 11);
    strncpy(StkRec.STDI03, StockDist, 24);
     iX = RANDOM(0, 9);
    memcpy(StockDist, DistInf1[iX], 11);
    memcpy((void *)&StockDist[11], DistInf2[3], 2);
     iX = RANDOM(0, 9);
    memcpy((void *)&StockDist[13], DistInf3[iX], 11);
    strncpy(StkRec.STDI04, StockDist, 24);
     iX = RANDOM(0, 9);
    memcpy(StockDist, DistInf1[iX], 11);
    memcpy((void *)&StockDist[11], DistInf2[4], 2);
```

```
     iX = RANDOM(0, 9);
    memcpy((void *)&StockDist[13], DistInf3[iX], 11);
    strncpy(StkRec.STDI05, StockDist, 24);
     iX = RANDOM(0, 9);
    memcpy(StockDist, DistInf1[iX], 11);
    memcpy((void *)&StockDist[11], DistInf2[5], 2);
     iX = RANDOM(0, 9);
    memcpy((void *)&StockDist[13], DistInf3[iX], 11);
    strncpy(StkRec.STDI06, StockDist, 24);
     iX = RANDOM(0, 9);
    memcpy(StockDist, DistInf1[iX], 11);
    memcpy((void *)&StockDist[11], DistInf2[6], 2);
     iX = RANDOM(0, 9);
    memcpy((void *)&StockDist[13], DistInf3[iX], 11);
    strncpy(StkRec.STDI07, StockDist, 24);
     iX = RANDOM(0, 9);
    memcpy(StockDist, DistInf1[iX], 11);
    memcpy((void *)&StockDist[11], DistInf2[7], 2);
     iX = RANDOM(0, 9);
    memcpy((void *)&StockDist[13], DistInf3[iX], 11);
    strncpy(StkRec.STDI08, StockDist, 24);
     iX = RANDOM(0, 9);
    memcpy(StockDist, DistInf1[iX], 11);
    memcpy((void *)&StockDist[11], DistInf2[8], 2);
     iX = RANDOM(0, 9);
    memcpy((void *)&StockDist[13], DistInf3[iX], 11);
    strncpy(StkRec.STDI09, StockDist, 24);
     iX = RANDOM(0, 9);
    memcpy(StockDist, DistInf1[iX], 11);
    memcpy((void *)&StockDist[11], DistInf2[9], 2);
     iX = RANDOM(0, 9);
    memcpy((void *)&StockDist[13], DistInf3[iX], 11);
    strncpy(StkRec.STDI10, StockDist, 24);
      /* create STDATA random string 26 to 50 chars */
    CreateStr(26, 50, tempstr);
    if ((RANDOM(0,9)) == 0)
    {
      iX = (RANDOM(0,(strlen(tempstr)-8)));
      memcpy(&tempstr[iX], "ORIGINAL", 8);
    }
    memset(StkRec.STDATA, ' ', 50);
    strncpy(StkRec.STDATA, tempstr, strlen(tempstr));

    StkRec.STYTD = 0;
    StkRec.STORDRS = 0;
    StkRec.STREMORD = 0;

    _Rwrite(Stock_File,(void *)&StkRec,  sizeof(StkRec));

  } /* end of for loop */

  return;
} /* end of LoadStock */


/*****************************************************************/
/*   C r e a t e S t r                                           */
/*      Create string of random alphanumeric characters.         */
/*                                                               */
/*  PARMS:                                                       */
/*     iMin => minimum length of string                          */
/*     iMax => maximum length of string                          */
/*     szTemp => address to store created string                 */
/*  If string is not variable length, then set iMin and iMax to  */
/*  the actual fixed length.                                     */
/*****************************************************************/
void CreateStr(iMin, iMax, szTemp)
  int iMin, iMax;
  char *szTemp;
{
  int iAlphalen,
      i, j;

  iAlphalen = RANDOM(iMin,iMax);    /* determine str len  */
  memset(szTemp, '\0', strlen(szTemp));
  for (j=0; j<iAlphalen; j++)
  {
    i = RANDOM(0,61);
    szTemp[j]= chAlpha[i];
  }
  return;
} /* end of CreateStr */
```

LOADCST: Fill CSTMR and HSTRY Files

```
/*****************************************************************/
/* This program fills the customer and history files.            */
/* It fills them all simultaneously. It fills a customer         */
/* record, then the history record for that customer.            */
/*                                                               */
/*                                                               */
/*   C  H  A  N  G  E  S     &      U  P  D  A  T  E  S          */
/*                                                               */
/*                                                               */
/*    PROGRAM-ID. LOADCSTORD.                                    */
/*    AUTHOR. PPCOC.                                             */
/*    INSTALLATION. ROCHESTER.                                   */
/*    DATE-WRITTEN. 03/04/96.                                    */
/*    DATE-COMPILED. 03/06/96.                                   */
/*****************************************************************/

/* Start include files */
#include <stdlib.h>
#include <errno.h>
#include <recio.h>
#include <string.h>
#include <xxfdbk.h>
#include <milib.h>
#include <micomput.h>
#include <decimal.h>
#include <time.h>
 /* */
#define CUSTPERDIST 3000
#define RANDOM(a,b) (rand() % (b-a+1))+a
 /* */
/* MAPINC generates structures for the database files         */
#pragma mapinc("cstmrpf","*libl/cstmrpf(*all)","both", "d _P",,"tpcc")
#include "cstmrpf"
#pragma mapinc("hstry","*libl/hstry(*all)","both", "d _P",,"tpcc")
#include "hstry"
  /*  */
  /* Prototypes */
 void LoadCust(int, int, int);
 void LoadHstry(int, int, int);
 void CreateCLast(int, char *);
int NURand(int, int, int);
void CreateStr(int, int, char *);
void CreateZip(char *);
void GetDateTime(char *, char *);
int MaxRand(void);
 /*  */
static int C=7;
static unsigned long int rand_next = 1; /* used in MaxRand routine */
    /*    */
      /* There are 100 names.       */
 char *szFNameâã =
   {"FrankGifford    ",  "FranTarkenton   ",   "GailStormyNight ",
     "JohnWayne      ", "JaneSeeDickRun  ",   "JeannieWithLight",
     "DavesNotHere   ",  "MaryMartin     ",   "JefferyTheWaiter",
     "CarriePromQueen ",  "StephanKing    ",   "KaraKing       ",
     "AnnieOakley    ",  "BillClements   ",   "FannyFarmer    ",
     "AllanLuden     ",  "BernieLattitude",   "MelindaFisher  ",
     "ElaineBoosler  ",  "FrankieAvalon  ",   "Francine       ",
     "GaleGordon     ",  "JohnBoyWalton  ",   "JeanneCRiley   ",
     "DavidBowie     ",  "KuklaFranOllie ",   "PhillipOfWales ",
     "KerryCWilliams ",  "Stephanie      ",   "AnnaSueBrighton ",
     "BillyBob       ",  "FayDunawayWith ",   "PamelaSueMartin ",
     "AlanAlda       ",  "BernardPFife   ",   "LindaDayGeorge  ",
     "RobertConrad   ",  "Franklin       ",   "FrancisTheMule  ",
     "GayleMcDonald  ",  "JanMurray      ",   "JoanneCassidy   ",
     "MarkTheSpot    ",  "Geoffrey       ",   "PhyllisDiller   ",
     "DoogieHowser   ",  "Anastasia      ",   "AlvinChipmunk   ",
     "AlexPKeaton    ",  "BelindaMcWilliam",  "EllynBurnstyne  ",
     "JayePMorgan    ",  "JoeFromKokomo  ",   "JoeyHeatherton  ",
     "Gilligan       ",  "TheSkipper2    ",   "CarryBishop     ",
     "BenjaminFranklin",  "AbrahamLincoln  ",  "GeorgeWBush     ",
     "VincentPrice   ",  "PotsieWeber    ",   "RichieCunningham",
     "Alexander      ",  "PeterTheGreater ",  "MickeyMouse     ",
     "DonaldDuck     ",  "BabyHuey       ",   "CrystalGayle    ",
     "SnidelyWhiplash ",  "HastaLaVista   ",   "GeorgeWashington",
     "KareemAbdulJabar",  "WhitneyHouston  ",  "BobSmith        ",
     "PerryMason     ",  "PaulDrake      ",   "DellaMainStreet ",
     "MissPiggy      ",  "TheCookieMonster",  "KermitTheFrog   ",
     "BluesBrothers  ",  "BeaverCleaver   ",   "Anriqueta       ",
     "Crenshaw       ",  "AndyTaylor     ",   "ClaytonWilliams ",
     "BillClinton    ",  "Elizabeth      ",   "MichaelJordan   ",
     "PaulRevere     ",  "JohnGlenn      ",   "WashingtonCarver",
     "DanJohnson     ",  "BubbaTexan     ",   "GGordonLiddy    ",
     "ScottSimpson   ",  "BenKnight      ",
     "NolanRyan      ",  "AlbertEinstein  "   };

  /*  */
  /* There are 100 cities. */
   char  *CITY_ARRâã =
  {  "Saint_Paul          ",  "Saint_Louis         ",
     "Concord_Grapes      ",  "Trenton_New_Jersey  ",
     "Bismark_Doughnut    ",  "Cheyenne_Wyoming    ",
     "Juneau_Alaska       ",  "Honolulu_Hawaii     ",
     "Phoenix_The_Big_Bird",  "Topeka_Kansas       ",
     "Montgomery          ",  "Elmwood_Illinois    ",
     "Madison_Dolly       ",  "Lansing_Michigan    ",
     "Frankfurt_Germany   ",  "Des_Moines_Iowa     ",
     "Byron_Minnesota     ",  "Pierre_South_Dakota ",
     "Dover_Cliffs        ",  "Helena_Montana      ",
     "San_Antonio         ",  "Saint_Petersburg    ",
     "Caribu_Maine        ",  "Sault_Saint_Marie   ",
     "Los_Angeles         ",  "Minneapolis_St_Paul ",
     "Dallas_Fort_Worth   ",  "Kalamazoo_Michigan  ",
     "The_Land_Of_Oz      ",  "Lexington_Kentucky  ",
     "Washington          ",  "District_Of_Columbia",
     "New_Orleans         ",  "Boise_Idaho         ",
     "Kensington_Maryland ",  "Denver_Colorado     ",
     "Blue_Mountain       ",  "Riverside_Ca        ",
     "New_York_City       ",  "Nashville_Tennessee ",
     "Mayberry_RFD        ",  "Berkeley_California ",
     "Maryville_Missouri  ",  "San_Francisco       ",
     "Indianapolis        ",  "Saint_Joseph_Mo     ",
     "Rochester_Minnesota ",  "Fort_Worth          ",
     "Holland_Michigan    ",  "South_Orange_NJ     ",
     "Stillwater          ",  "Eugene_Oregon       ",
     "Alexandria_Virginia ",  "Salt_Lake_City      ",
     "Manitowoc_Wisconsin ",  "Philadelphia        ",
     "Sioux_Falls         ",  "West_Lafayette      ",
     "New_Brunswick_NJ    ",  "Rochester_New_York  ",
     "Oakland_Ca          ",  "Las_Vegas_Nevada    ",
     "Birmingham_Alabama  ",  "Arkadelphia_Arkansas",
     "San_Juan_Puerto_Rico",  "Omaha_Nebraska      ",
     "Walla_Walla         ",  "Baraboo_Wisconsin   ",
     "Atlanta_Georgia     ",  "Grand_Forks_ND      ",
     "Tempe_Arizona       ",  "Knoxville_Tn        ",
     "Fort_Leavenworth    ",  "Boston_Massachusetts",
     "Danville_Virginia   ",  "Baltimore_Maryland  ",
     "New_Haven_Ct        ",  "Claremont_California",
     "Otsego_Michigan     ",  "Providence          ",
     "Jacksonville        ",  "Columbia_SC         ",
     "London_England      ",  "Paris_France        ",
     "Chicago_Illinois    ",  "Albuquerque         ",
     "Raleigh_NC          ",  "Kansas_City_Missouri",
     "Tacoma_Washington   ",  "Oronoco_Mn          ",
     "Charleston_WV       ",  "Newark_Delaware     ",
     "Burlington_Vermont  ",  "Damariscotta_Maine  ",
     "Colorado_Springs_Co ",  "Nacogdoches_Texas   ",
     "Barbourville_Ky     ",  "Boca_Raton_Fl       ",
     "Mary_Of_The_Woods   ",  "Heston_And_Isleworth"    };

   /*  */
   /*  There are 100 states */
char *state_arrâã =
   {  "AL", "AK", "AR", "AZ", "CA", "CO", "CT", "DE", "FL", "GA",
     "HI", "ID", "IL", "IN", "IA", "KS", "KY", "LA", "ME", "MD",
     "MA", "MI", "MN", "MS", "MO", "MT", "NE", "NV", "NH", "NM",
     "NJ", "NY", "NC", "ND", "OH", "OK", "OR", "PA", "RI", "SC",
     "SD", "TN", "TX", "UT", "VT", "VA", "WA", "WV", "WI", "WY",
     "AC", "AS", "AI", "AM", "AE", "AV", "AW", "CN", "CS", "DI",
     "DM", "DN", "DS", "ED", "EM", "EN", "HN", "HO", "IH", "IM",
     "IR", "IW", "JN", "KA", "KO", "LF", "LI", "NI", "NT", "NZ",
     "OC", "OM", "ON", "PR", "RA", "RO", "SK", "SM", "TC", "TM",
     "TY", "TV", "VI", "VN", "VW", "XT", "YK", "YN", "YW", "ZA"    };
 /*  */
tpcc_CSRCD_both_t CustRec;
tpcc_HSRCD_both_t HstRec;
 /*  */
static char chAlpha[62] =
       "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ";
static char chNumStr[11] = "0123456789";
static char tempstr[81];
static time_t ltime;
static char TimeStr[7];
static char DateStr[9];
 /*  */
  _RFILE
          *Cstmr_File,
          *Hstry_File;
/* */
int main(argc, argv)
register int argc; char **argv;
{
  short int i;
  short int iStrtWrhs = 1;
```

```
  short int iEndWrhs = 1;
  short int iCurWrhs = 1;
  short int iDist;
  int iCurCid;
  int iRetCode = 0;
    /* Set up random number generator */
  time(&ltime);
  srand(ltime);          /* seed the random num generator */
  rand_next = (unsigned long int)ltime;
      /* set up alpha seed for random chars */
  strcat(chAlpha, chNumStr);
    /* parse the input parms */
  for (argc--, argv++, i=0; argc>0; ++i, argc--, argv++)
  {
    if (i==0) {
      iStrtWrhs = atoi(*argv);
    }
    else {
      iEndWrhs = atoi(*argv);
    }
  }  /* end of parsing input parms */

  if( (Hstry_File = _Ropen("hstry","rr+","blkrcd=Y")) == NULL)
  {
    printf("Open of History file failed\n");
    exit(99);
  }
  if( (Cstmr_File = _Ropen("cstmrpf","rr+","blkrcd=Y")) == NULL)
  {
    printf("Open of Customer file failed\n");
    exit(99);
  }


  for (iCurCid=1; iCurCid<=CUSTPERDIST; iCurCid++)
  {
    for (iCurWrhs=iStrtWrhs; iCurWrhs<=iEndWrhs; iCurWrhs++)
    {
      for (iDist=1; iDist<=10; iDist++)
      {
        LoadHstry(iCurCid, iCurWrhs, iDist);
        LoadCust(iCurWrhs, iDist, iCurCid);
      }
    } /* of for (iCurWrhs=iStrtWrhs; iCurWrhs<=iEndWrhs; iCurWrhs++) */
  }

 _Rclose(Cstmr_File);
 _Rclose(Hstry_File);

 return(iRetCode);
} /* end of main */


/*****************************************************************/
/*   L o a d C u s t                                             */
/*      Create & write customer recs.                            */
/*                                                               */
/*  PARMS:                                                       */
/*     iWid => current warehouse id                              */
/*     iDid => current district id                               */
/*****************************************************************/
void LoadCust(iWid, iDid, iCid)
  short int iWid;
  short int iDid;
  int iCid;
{
  char tstr[21];
  int iX;
  static char cdataStr[500];

  CustRec.CWID = iWid;
  CustRec.CDID = iDid;
  CustRec.CID = iCid;

      /* build cfirst. array 0..99 */
  iX = RANDOM(0,99);
  memset(CustRec.CFIRST, ' ', 16);
  if ((iWid==1) && (iDid==1) && (iCid==1))
  {
     sprintf(CustRec.CFIRST,"C_load = %d",C);

  }
  else
  {
  strncpy(CustRec.CFIRST, szFNameâiXâ, strlen(szFName[0]));
  }
      /* build cinit. */
  strncpy(CustRec.CINIT, "OE", 2);
      /* build clast */
  if (iCid < 1001)
  {
    CreateCLast((iCid-1), tstr);
  }
  else
  {
    iX = NURand(255, 0, 999);
    CreateCLast((iX), tstr);
  }
  memset(CustRec.CLAST, ' ', 16);
  strncpy(CustRec.CLAST, tstr, strlen(tstr) );
      /* build address1 */
  CreateStr(10, 20, tstr);
  memset(CustRec.CADDR1, ' ', 20);
  strncpy(CustRec.CADDR1, tstr, strlen(tstr));
      /* build address2 */
  CreateStr(10, 20, tstr);
  memset(CustRec.CADDR2, ' ', 20);
  strncpy(CustRec.CADDR2, tstr, strlen(tstr));
      /* build city */
  iX = RANDOM(0,99);
  memset(CustRec.CCITY, ' ', 20);
  strncpy(CustRec.CCITY, CITY_ARR[iX], strlen(CITY_ARR[iX]));
      /* build state */
  strncpy(CustRec.CSTATE, state_arr[iX], 2);
      /* build zip code */
  CreateZip(tstr);
  strncpy(CustRec.CZIP, tstr, strlen(tstr));

      /* build phone # */
  memset(tstr, '\0', strlen(tstr));
  for (iX = 0; iX < 16; iX++)
  {
    CustRec.CPHONE[iX] = chNumStr[(RANDOM(0,9))];
  }
      /* build credit limit */
  CustRec.CCRDLM = 50000.00;
      /* build credit status */
  if ( (RANDOM(0,9)) > 0 ) {
    CustRec.CCREDT[0] = 'G';
  }
  else
    CustRec.CCREDT[0] = 'B';
  CustRec.CCREDT[1] = 'C';
      /* build Cust balance */
  CustRec.CBAL = -10.00;
      /* build Cust YTD payments */
  CustRec.CYTD = 10.00;
      /* build Cust pay count & delivery count */
  CustRec.CPAYCNT = 1;
  CustRec.CDELCNT = 0;
      /* build Cust Discount */
  CustRec.CDCT = ((RANDOM(0,5000)) / 10000.00);
      /* build Cust Data */
  CreateStr(300, 500, cdataStr);
  memset(CustRec.CDATA, ' ', 500);
  strncpy(CustRec.CDATA, cdataStr, strlen(cdataStr));
      /* build Cust Date & Time */
  GetDateTime(TimeStr, DateStr);
  strncpy(CustRec.CLDATE, DateStr, strlen(DateStr));
  strncpy(CustRec.CLTIME, TimeStr, strlen(TimeStr));

  _Rwrite(Cstmr_File,(void *)&CustRec,  sizeof(CustRec));

  return;
} /* end of LoadCust */


/*****************************************************************/
/*   L o a d H s t r y                                           */
/*      Create & write history rec.                              */
/*                                                               */
/*  PARMS:                                                       */
/*     iWid => current warehouse id                              */
/*****************************************************************/
void LoadHstry(iCid, iWid, iDid)
  int iCid;
  short iWid;
  short iDid;
{


  HstRec.HCWID = iWid;
  HstRec.HWID = iWid;
  HstRec.HDID = iDid;
  HstRec.HCDID = iDid;
  HstRec.HCID = iCid;

  HstRec.HAMT = 10.00;
      /* build Hist Data */
  CreateStr(12, 24, tempstr);
  memset(HstRec.HDATA, ' ', 24);
  strncpy(HstRec.HDATA, tempstr, strlen(tempstr));
      /* build Hist Date & Time */
  GetDateTime(TimeStr, DateStr);
  strncpy(HstRec.HTIME, TimeStr, strlen(TimeStr));
  strncpy(HstRec.HDATE, DateStr, strlen(DateStr));

  _Rwrite(Hstry_File,(void *)&HstRec,  sizeof(HstRec));
} /* end of LoadHstry */


/*****************************************************************/
/*   N U R a n d                                                 */
/*      Return integer value from nurand function.               */
/*      function is                                              */
/*   NURand(A,x,y) = (((rand(0,A) | rand(x,y))+C) % (y-x+1)) + x  */
/*     A is constant chosen according to size of range [x..y]    */
/*      expl | exp2 stands for bitwise logical OR of expl and exp2*/
/*      rand(x,y) stands for randomly selected within âx..yâ     */
/*        C is run-time constant randomly chosen within [0..A]   */
/*                                                               */
/*  PARMS:                                                       */
/*     A => integer value                                        */
/*     x => minimum value in range                               */
/*     y => maximum value in range                               */
/*****************************************************************/
int NURand(A, x, y)
  int A;
  int x;
  int y;
{
  int retval;
  int exp1, exp2;

  exp1 = RANDOM(0,A);
  exp2 = RANDOM(x,y);
  exp1 = (exp1 | exp2) + C;
  exp2 = (y-x+1) + x;
  retval = exp1 % exp2;
  return(retval);
} /* end of NURand */


/*****************************************************************/
/*   C r e a t e C L a s t                                       */
/*      Build cust last name from predefined syllables.          */
/*                                                               */
/*  PARMS:                                                       */
/*     iNum => num used to build last name                       */
/*     NameStr => address to store last name                     */
/*****************************************************************/
void CreateCLast(iNum, NameStr)
  int iNum;
  char *NameStr;
{
  static char *sylble[] =
                {"BAR", "OUGHT", "ABLE", "PRI", "PRES",
                 "ESE", "ANTI", "CALLY", "ATION", "EING" };
  strcpy(NameStr, sylble[iNum/100]);
  strcat(NameStr, sylble[(iNum/10)%10]);
  strcat(NameStr, sylble[(iNum%10)]);

  return;
} /* end of CreateCLast */


/*****************************************************************/
/*   C r e a t e S t r                                           */
/*      Create string of random alphanumeric characters.         */
/*                                                               */
/*  PARMS:                                                       */
/*     iMin => minimum length of string                          */
/*     iMax => maximum length of string                          */
/*     szTemp => address to store created string                 */
/*  If string is not variable length, then set iMin and iMax to  */
/*  the actual fixed length.                                     */
/*****************************************************************/
void CreateStr(iMin, iMax, szTemp)
  int iMin, iMax;
  char *szTemp;
{
  int iAlphalen,
      i, j;

  iAlphalen = RANDOM(iMin,iMax); /* determine str len  */
  memset(szTemp, '\0', strlen(szTemp));
  for (j=0; j<iAlphalen; j++)
  {
```

```
    i = RANDOM(0,61);
    szTemp[j]= chAlpha[i];
  }
  return;
} /* end of CreateStr */


/*****************************************************************/
/*   G e t D a t e T i m e                                       */
/*      Get current date/time stamp.                             */
/*                                                               */
/*  PARMS:                                                       */
/*     szTime => pointer to time string                          */
/*     szDate => pointer to date string                          */
/*****************************************************************/
void GetDateTime(szTime, szDate)
  char *szTime;
  char *szDate;
{
  char tstr[5];
  static struct tm *DateTime;
  short iYear;

  time(&ltime);   /* get current date & time */
  DateTime = gmtime(&ltime);
    /* return time */
  sprintf(tstr, "%02d", DateTime->tm_hour);
  strcpy(szTime, tstr);
  sprintf(tstr, "%02d", DateTime->tm_min);
  strcat(szTime, tstr);
  sprintf(tstr, "%02d", DateTime->tm_sec);
  strcat(szTime, tstr);
    /* return date */
  sprintf(tstr, "%02d", (DateTime->tm_mon + 1));
  strcpy(szDate, tstr);
  sprintf(tstr, "%02d", DateTime->tm_mday);
  strcat(szDate, tstr);
  iYear = DateTime->tm_year + 1900;
  sprintf(tstr, "%04d", iYear);
  strcat(szDate, tstr);
} /* end of GetDateTime */


/*****************************************************************/
/*   C r e a t e Z i p                                           */
/*      Create zip code string.                                  */
/*                                                               */
/*  PARMS:                                                       */
/*    Returns zip code string.                                   */
/*****************************************************************/
void CreateZip(zipstr)
char *zipstr;
{
/* Redefine max generated random number. Default is 32,767. */
  int iZ;

  iZ = MaxRand();
  sprintf(zipstr, "%05d", iZ);
  strcat(zipstr, "1111");
  return;
} /* end of CreateZip */


/* Redefine max generated random number. Default is 32,767. */
#undef RAND_MAX
#define RAND_MAX +99999
/*****************************************************************/
/*   M a x R a n d                                               */
/*      Create random number larger than 32767.                  */
/*  NOTE: This is the rand() function used by AS/400 with one    */
/*      slight modification.  In the rand() function, rand_next is  */
/*      shifted right by 16 rather than the 14 used here.  Shifting */
/*      by 16 produces a max value of 65535.                     */
/*                                                               */
/*  PARMS:                                                       */
/*    Returns zip code string.                                   */
/*****************************************************************/
int MaxRand(void)
{
  int rand_temp;

  rand_next = rand_next * 1103515245 + 12345;
  rand_temp = (rand_next >> 14)%((unsigned long)(RAND_MAX)+1);
  return(rand_temp);
} /* end of CreateZip */
```

LOADORD: Fill ORDERS, ORDLIN, and NEWORD Files

```
/*****************************************************************/
/* This program fills the neword, orders, and ordlin */
/* files.  It fills them all simultaneously. It      */
/* will create an order record, selecting the customer for that  */
/* order at random (each customer will have one and only one order  */
/* but which order that will be is determined with random numbers). */
/* The next thing done is only done for orders between 2,101 and     */
/* 3000 for each district, this is the creation of a neworder       */
/* record. The orderline records for that order are created last,   */
/* the number of lines for the  order is between 5 and 15, selected */
/* at random.                                                       */
/*   C  H  A  N  G  E  S     &      U  P  D  A  T  E  S             */
/*                                                                  */
/*                                                                  */
/*                                                                  */
/*    PROGRAM-ID. LOADORD.                                          */
/*    AUTHOR. TPCC.                                                 */
/*    INSTALLATION. ROCHESTER.                                      */
/*    DATE-WRITTEN. 03/07/96.                                       */
/*    DATE-COMPILED. 03/07/96.                                      */
/*****************************************************************/

/* Start include files */
#include <stdlib.h>
#include <errno.h>
#include <recio.h>
#include <string.h>
#include <xxfdbk.h>
#include <milib.h>
#include <micomput.h>
#include <decimal.h>
#include <time.h>
#include <xxcvt.h>
 /* */
/* MAPINC generates structures for the database files          */
#pragma mapinc("newordpf","*libl/newordpf(*all)","both key","d _P",,"tpcc")
#include "newordpf"
#pragma mapinc("orderspf","*libl/orderspf(*all)","both key", "d _P",,"tpcc ")
#include "orderspf"
#pragma mapinc("ordlinpf","*libl/ordlinpf(*all)","both key","d _P",,"tpcc" )
#include "ordlinpf"
  /*
        */
        /*     Orders per district     */
#define ORDERSPERDIST 3000
#define MAX_RAND 100000
#define RANDOM(a,b) (rand() % (b-a+1))+a
#define MAXNUM1 600

 void LoadOrders(int, int, int);
 void LoadNeword(int, int, int);
 int GetCstmrID(int);
 int GetItemID(void);
 void GetDateTime(char *, char *);
 void GetCarrier(char *);
 void GetAmount(char *);
 int MaxRand(void);
 void BldArray(void);

 int v;

    _RFILE
          *NewOrder_File,
          *Orders_File,
          *OrderLine_File;

tpcc_ORRCD_both_t OrderRec;
tpcc_OLRCD_both_t OrdlnRec;
tpcc_NORCD_both_t NewordRec;

 static time_t ltime;
 static char TimeStr[7];
 static char DateStr[9];
 static char CarrStr[3];
 static char AmntStr[8];
 static int ocid;
 static unsigned long int rand_next = 1;
 int  numarray[ORDERSPERDIST+1];
  /*
        */

        /*    There are 10 entries of length 24 each  */
  char  *DIST_INFOâã =
      {"Info for District Num 00",
       "Info for District Num 01",
       "Info for District Num 02",
       "Info for District Num 03",
       "Info for District Num 04",
       "Info for District Num 05",
       "Info for District Num 06",
       "Info for District Num 07",
       "Info for District Num 08",
       "Info for District Num 09",
       "Info for District Num 10" };
    /*
            */
  /* There are 3 sets of characters that are concatenated */
  /* to make the district information fields for Stock */
char *DistInfl[] = {
  "abcdefghijk", "nowisthetim", "thisiswhere",
  "whatdoyouno", "notmuchthen", "whydoyouask",
  "Idonotknow_", "sowhereisit", "inthecities",
  "andthetowns"};

char *DistInf2[] = {
  "01", "02", "03", "04", "05", "06", "07", "08", "09", "10"};

char *DistInf3[] = {
  "mnopqrstuvw", "whydidyoudo", "interactive",
  "batchwork__", "systemview_", "performance",
  "Ineverwork_", "Ialwaysplay", "nexttolast_",
  "nowthefinal"};

    /*
            */
            char zeroesd[8] = "00000000";
            char zeroest[6] = "000000";
    /*
            */
            char *CarId[] = { "00","01", "02", "03", "04", "05",
                              "06", "07", "08", "09", "10" };
    /*
            */

int main(argc, argv)
register int argc; char **argv;
{
   short int i;
   short int iStrWrhs=1;
   short int iEndWrhs=1;
   short int iCurWrhs=1;
   short int iCurDist;
   int iCurOid;
   int rcode = 0;


  if( (Orders_File = _Ropen("orderspf","rr+","blkrcd=Y")) == NULL)
  {
    printf("Open of orders file failed\n");
    exit(99);
  }
  if( (OrderLine_File = _Ropen("ordlinpf","rr+","blkrcd=Y")) == NULL)
  {
    printf("Open of Orderline file failed\n");
    exit(99);
  }
  if( (NewOrder_File = _Ropen("newordpf","rr+","blkrcd=Y")) == NULL)
  {
    printf("Open of NewOrder file failed\n");
    exit(99);
  }

    /* Set up random number generator */
  time(&ltime);
  srand(ltime);          /* seed the random num generator */
  rand_next = (unsigned long int)ltime;

  BldArray();
    /* parse the input parms */
  for (argc--, argv++, i=0; argc>0; ++i, argc--, argv++)
  {
    if (i==0) {
      iStrWrhs = atoi(*argv);
    }
    else  {
      iEndWrhs = atoi(*argv);
    }
  }  /* end of parsing input parms */

  for (iCurOid=1; iCurOid<=ORDERSPERDIST; iCurOid++)
  {
```

```
    for (iCurWrhs=iStrWrhs; iCurWrhs<=iEndWrhs; iCurWrhs++)
    {

      for (iCurDist=1; iCurDist<=10; iCurDist++)
      {

        LoadOrders(iCurWrhs, iCurDist, iCurOid);
        if(iCurOid >= 2101)
        {
          LoadNeword(iCurWrhs, iCurDist, iCurOid);
        }
      }
    }
   }
 _Rclose(Orders_File);
 _Rclose(OrderLine_File);
 _Rclose(NewOrder_File);

 return(rcode);
} /* end of main */


/*****************************************************************/
/*   L o a d O r d e r s                                         */
/*      Create & write orders and Orderline rec.                 */
/*                                                               */
/*  PARMS:                                                       */
/*     iCurWrhs  => current warehouse id                         */
/*     iCurDist  => current district id                          */
/*     iCurOid   => current Order id                             */
/*****************************************************************/
void LoadOrders(iCurWrhs, iCurDist, iCurOid)
  short int iCurWrhs;
  short int iCurDist;
  int iCurOid;
{
  decimal(7,2) d=0.00;
  int i,tmpvar;
  short int oln,curoln;
  char StockDist[25];

  OrderRec.OID  = OrdlnRec.OLOID = iCurOid;
  OrderRec.OWID = OrdlnRec.OLWID = iCurWrhs;
  OrderRec.ODID = OrdlnRec.OLDID = iCurDist;

     OrderRec.OCID = GetCstmrID(iCurOid);
     GetDateTime(TimeStr, DateStr);
     strncpy(OrderRec.OENTDT,DateStr,strlen(DateStr));
     strncpy(OrderRec.OENTTM,TimeStr,strlen(TimeStr));
     if(iCurOid<2101)
     {
       GetCarrier(CarrStr);
       strncpy(OrderRec.OCARID,CarrStr,strlen(CarrStr));
     }
     else
     {
       memset(OrderRec.OCARID,'\0', strlen(OrderRec.OCARID));
     }
     OrderRec.OLINES = RANDOM(5, 15);
     curoln = OrderRec.OLINES;
     OrderRec.OLOCAL = 1;
     _Rwrite(Orders_File,(void *)&OrderRec, sizeof(OrderRec));
     for(oln=1;oln<=curoln;oln++)
     {
         OrdlnRec.OLOID = iCurOid;
         OrdlnRec.OLNBR = oln;
         while((OrdlnRec.OLIID = MaxRand())==0);
         OrdlnRec.OLSPWH = iCurWrhs;
         OrdlnRec.OLQTY = 5;
         if(iCurOid<2101)
         {
strncpy(OrdlnRec.OLDLVD, OrderRec.OENTDT, sizeof(OrderRec.OENTDT));
strncpy(OrdlnRec.OLDLVT, OrderRec.OENTTM, sizeof(OrderRec.OENTTM));
           OrdlnRec.OLAMNT = 0.00;
         }
         else
         {
           strncpy(OrdlnRec.OLDLVD, zeroesd, 8);
           strncpy(OrdlnRec.OLDLVT, zeroest, 6);
           tmpvar = MaxRand();
           d = tmpvar / 100;
           OrdlnRec.OLAMNT = d;
         }
     tmpvar = RANDOM(0, 9);
    memcpy(StockDist, DistInf1[tmpvar], 11);
    memcpy((void *)&StockDist[11], DistInf2[iCurDist-1],2);
     tmpvar = RANDOM(0, 9);
    memcpy((void *)&StockDist[13], DistInf3[tmpvar],11);
    strncpy(OrdlnRec.OLDSTI, StockDist, 24);
         _Rwrite(OrderLine_File,(void *)&OrdlnRec, sizeof(OrdlnRec));
     } /* End of order line loop  */
 return;
}      /*End of LoadOrders  */



/*****************************************************************/
/*   L o a d N e w o r d                                         */
/*      Create & write neword rec.                               */
/*                                                               */
/*  PARMS:                                                       */
/*     iCurWrhs  => current warehouse id                         */
/*     iCurDist  => current district id                          */
/*     iCurOid   => current order id                             */
/*****************************************************************/
void LoadNeword(iCurWrhs, iCurDist, iCurOid)
  short int iCurWrhs;
  short int iCurDist;
  short int iCurOid;
 {

   NewordRec.NOWID = iCurWrhs;
   NewordRec.NODID = iCurDist;
   NewordRec.NOOID = iCurOid;
   _Rwrite(NewOrder_File,(void *)&NewordRec, sizeof(NewordRec));
   return;
}      /*End of LoadNeword  */

/*****************************************************************/
/*   G e t C s t m r I D                                         */
/*      Generates a random customer id.                          */
/*                                                               */
/*  PARMS:                                                       */
/*     oid => Order ID                                           */
/*****************************************************************/
int GetCstmrID(oid)
int oid;
{


   if( (OrderRec.OWID == 1) && (OrderRec.ODID == 1) )
   {
        if (oid==1)
        {         v=RANDOM(1, ORDERSPERDIST);
                  ocid = numarray[v];
        }
       else
         {
                  v++;
                  if(v > ORDERSPERDIST)
                  {
                  v = 1;
                  }
                  ocid= numarray[v];
         }
   }
   else
   {
     ocid++;
     if(ocid > ORDERSPERDIST)
     {
       ocid = 1;
     }
   }
   return(ocid);
} /* End of GetCstmrID */

/*****************************************************************/
/*   G e t D a t e T i m e                                       */
/*      Get current date/time stamp.                             */
/*                                                               */
/*  PARMS:                                                       */
/*     szTime => pointer to time string                          */
/*     szDate => pointer to date string                          */
/*****************************************************************/
void GetDateTime(szTime, szDate)
  char *szTime;
  char *szDate;
{
  char tstrâ5ã;
  static struct tm *DateTime;
  short iYear;

  time(&ltime);   /* get current date & time */
  DateTime = gmtime(&ltime);
    /* return time */
  sprintf(tstr, "%02d", DateTime->tm_hour);
  strcpy(szTime, tstr);
  sprintf(tstr, "%02d", DateTime->tm_min);
  strcat(szTime, tstr);
  sprintf(tstr, "%02d", DateTime->tm_sec);
  strcat(szTime, tstr);
    /* return date */
  sprintf(tstr, "%02d", (DateTime->tm_mon + 1));
  strcpy(szDate, tstr);
  sprintf(tstr, "%02d", DateTime->tm_mday);
  strcat(szDate, tstr);
  iYear = DateTime->tm_year + 1900;
  sprintf(tstr, "%04d", iYear);
  strcat(szDate, tstr);
  return;
} /* end of GetDateTime */


/*****************************************************************/
/*   G e t C a r r i e r                                         */
/*      Get carrier id if oid < 2101.                            */
/*                                                               */
/*  PARMS:                                                       */
/*     szCarr => pointer to carrier string                       */
/*****************************************************************/
void GetCarrier(szCarr)
  char *szCarr;
{
  char tstr[2];
  int ri;
  ri = RANDOM(1, 10);
  strncpy(szCarr, CarId[ri],strlen(CarId[0]));
  return;
} /* end of GetCarrier */


/*****************************************************************/
/*   M a x R a n d                                               */
/*      Get item id between 1 & 100,000                          */
/*                                                               */
/*  PARMS:                                                       */
/*     None.                                                     */
/*****************************************************************/
int MaxRand()
{
  int rand_temp;

  rand_next = rand_next * 1103515245 + 12345;
  rand_temp = (rand_next >> 14)%((unsigned long)(MAX_RAND)+1);
  return(rand_temp);
}
/*****************************************************************/
/*   Build the array for OCIDs.                                  */
/*      Generates a unique random customer id and builds an array.  */
/*                                                               */
/*  PARMS:                                                       */
/*     None                                                      */
/*****************************************************************/
void BldArray()
{
int i,j,k=0,n=0,x,num,same,y1=0,y2=0,y3=0,y4=0,y5=0,count=0;
int numarray1[MAXNUM1],
    numarray2[MAXNUM1],
    numarray3[MAXNUM1],
    numarray4[MAXNUM1],
    numarray5[MAXNUM1];



    for (j=0; j <= ORDERSPERDIST; j++)
      {        numarray[j] = 0;
               numarray1[j] = 0;
               numarray2[j] = 0;
               numarray3[j] = 0;
               numarray4[j] = 0;
               numarray5[j] = 0;
     }
         x=MAXNUM1;

     for (i=0; i <= ORDERSPERDIST; i++)
       {
           num = RANDOM(1,ORDERSPERDIST);
           same = 0;
        for (j=0; j < x; j++)
          {
           if (num == numarray1[j])   same=1;
           if (num == numarray2[j])   same=1;
           if (num == numarray3[j])   same=1;
           if (num == numarray4[j])   same=1;
           if (num == numarray5[j])   same=1;
          }
          if (same==0) {
```

```
          count++;
          numarray[n++]=num;
          if ((count>0) && (count<600))
           numarray1[y1++]=num;
          if ((count>=600) && (count<1200))
            numarray2[y2++]=num;
           if ((count>=1200) && (count<1800))
            numarray3[y3++]=num;
           if ((count>=1800) && (count<2400))
            numarray4[y4++]=num;
           if ((count>=2400) && (count<3000))
            numarray5[y5++]=num;
            }
            else i--;
        }
} /* End of BldArray */
```

SETUP: System Setup Program

```
            PGM        PARM(&OBJLIB &BLDLIB &DATLIB &CRTSAVFILE +
                         &SAVLIB &JRNLIB &JRNASP &STRWHC &ENDWHC)

/* This program was changed to submit several stages of it to    */
/* batch. 12/19/93                                               */
/* FURTHER CHANGES MADE 1/07/94                                  */
/* This program is one of the last called during the creation of */
/* the TPCC data base.  It creates the data queue from which     */
/* each user will be assigned a warehouse/district when they     */
/* sign on.  It also creates the job description and user        */
/* profile for the interactive users, next it creates the job    */
/* and subsystem descriptions for the batch jobs which will      */
/* execute the deffered portion of the delivery transaction.     */
/* It then creates the journal and journal receiver which are    */
/* used when a transaction is to be rolled-back or when lost     */
/* transactions need to be re-applied.  After creating the       */
/* journal it reorganizes the customer file to be sorted by      */
/* customer last name and first name, then the customer logical  */
/* file is created.  After all of the files are created the      */
/* journaling of the physical files is begun.  The last thing    */
/* this program does is to save the DB so we can use the restore */
/* function to get the DB back to its initial state.             */

             DCL        VAR(&OBJLIB) TYPE(*CHAR) LEN(10)
             DCL        VAR(&BLDLIB) TYPE(*CHAR) LEN(10)
             DCL        VAR(&DATLIB) TYPE(*CHAR) LEN(10)
             DCL        VAR(&SAVLIB) TYPE(*CHAR) LEN(10)
             DCL        VAR(&JRNLIB) TYPE(*CHAR) LEN(10)
             DCL        VAR(&JRNASP) TYPE(*CHAR) LEN(1)
             DCL        VAR(&CRTSAVFILE) TYPE(*CHAR) LEN(1)
             DCL        VAR(&STRWH) TYPE(*DEC) LEN(4 0)
             DCL        VAR(&ENDWH) TYPE(*DEC) LEN(4 0)
             DCL        VAR(&STRWHC) TYPE(*CHAR) LEN(4)
             DCL        VAR(&ENDWHC) TYPE(*CHAR) LEN(4)
/* Monitor for messages and setup library list                   */

             MONMSG     MSGID(CPF1064) /* Class already exists */
             MONMSG     MSGID(CPD1411) /* Subsystem already exists */
             MONMSG     MSGID(CPD1416) /* Subsystem already active */
             MONMSG     MSGID(CPD1475) /* Routing Entry already +
                          exists */
             MONMSG     MSGID(CPD1535) /* Job Queue Entry already +
                          exists */
             MONMSG     MSGID(CPD1611) /* JOBD already exists */
             MONMSG     MSGID(CPD1621) /* JOBQ not FOUND      */
             MONMSG     MSGID(CPF1621) /* JOBD not created    */
             MONMSG     MSGID(CPF1696) /* Subsystem not created */
             MONMSG     MSGID(CPF1697) /* Subsystem not changed */
             MONMSG     MSGID(CPF2103) /* Library already in list */
             MONMSG     MSGID(CPF2105) /* Object not found       */
             MONMSG     MSGID(CPF2110) /* Object not found       */
             MONMSG     MSGID(CPF2111) /* Library already exists */
             MONMSG     MSGID(CPF2204) /* User Profile Does Not +
                          Exist */
             MONMSG     MSGID(CPF2555) /* Reply list entry exists */
             MONMSG     MSGID(CPF3323) /* Job Queue already exists */
             MONMSG     MSGID(CPF3770) /* No objects saved or    +
                              restored  */
             MONMSG     MSGID(CPF5813) /* File already exists    */
             MONMSG     MSGID(CPF7010) /* Journal already exists */
             MONMSG     MSGID(CPF7032) /* File not journaled     */
             MONMSG     MSGID(CPF7302) /* File not created       */
             MONMSG     MSGID(CPF7311) /* Errors not allowed in  +
                               DDS                              */
             MONMSG     MSGID(CPF9812) /* File not found         */

             ADDRPYLE   SEQNBR(3103) MSGID(CPA4067) RPY(G) /* +
                          Save of Data over existing Save Files */

             ADDLIBLE   LIB(&BLDLIB)
             ADDLIBLE   LIB(&DATLIB)
             CHGVAR     VAR(&STRWH) VALUE(&STRWHC)
             CHGVAR     VAR(&ENDWH) VALUE(&ENDWHC)



             SNDPGMMSG  MSGID(CPF9898) MSGF(QCPFMSG) +
                          MSGDTA('Creating Logical Files') +
                          TOPGMQ(*EXT) MSGTYPE(*STATUS)
/*  BUILD INDEXES                                                */

             SBMJOB     CMD(CALL PGM(&OBJLIB/ORDERSVIEW) +
                          PARM(&DATLIB &BLDLIB &STRWHC &ENDWHC)) +
                          JOB(ORDERSVIEW) JOBQ(&OBJLIB/TPCBLDJOBQ) +
                          MSGQ(&BLDLIB/TPCCBLDMSG)

             SBMJOB     CMD(CALL PGM(&OBJLIB/ORDLINVIEW) +
                          PARM(&DATLIB &BLDLIB)) JOB(ORDLINVIEW) +
                          JOBQ(&OBJLIB/TPCBLDJOBQ) +
                          MSGQ(&BLDLIB/TPCCBLDMSG)

             SBMJOB     CMD(CALL PGM(&OBJLIB/NEWORDVIEW) +
                          PARM(&DATLIB &BLDLIB)) JOB(NEWORDVIEW) +
                          JOBQ(&OBJLIB/TPCBLDJOBQ) +
                          MSGQ(&BLDLIB/TPCCBLDMSG)

             SBMJOB     CMD(CALL PGM(&OBJLIB/CSTMRVIEW) PARM(&DATLIB +
                          &BLDLIB)) JOB(CSTMRVIEW) +
                          JOBQ(&OBJLIB/TPCBLDJOBQ) +
                          MSGQ(&BLDLIB/TPCCBLDMSG)



/* Creation of TPCC User Profile                                 */

 ENDIT:      CRTJOBD    JOBD(&OBJLIB/TPCCJOBD) JOBQ(QGPL/QBATCH) +
                          TEXT('Job description for TPCCUSER') +
                          LOG(0) INQMSGRPY(*DFT)
             DLTUSRPRF  USRPRF(TPCCUSER)
             MONMSG     MSGID(CPF2215) EXEC(GOTO CMDLBL(CHGPRF))
             CRTUSRPRF  USRPRF(TPCCUSER) CURLIB(&OBJLIB) +
                          INLPGM(QCMD) TEXT('TPCC User created by +
                          BLDTPCC') SPCAUT(*SAVSYS *ALLOBJ *JOBCTL) +
                          JOBD(&OBJLIB/TPCCJOBD) DLVRY(*HOLD)
             GOTO       CMDLBL(CRTJOBQ)

/* Change the profile if it existed previously                   */

 CHGPRF:     CHGUSRPRF  USRPRF(TPCCUSER) CURLIB(&OBJLIB) +
                          INLPGM(*LIBL/QCMD) TEXT('TPCC User - +
                          Changed by BLDTPCC') SPCAUT(*SAVSYS +
                          *ALLOBJ *JOBCTL) JOBD(&OBJLIB/TPCCJOBD) +
                          DLVRY(*HOLD)


/* Create job queue, class and subsystem description for the   */
/* batch portion of the delivery transaction                   */

 CRTJOBQ:    CRTJOBQ    JOBQ(&OBJLIB/DLVRYJOBQ) TEXT('Delivery Job +
                          Queue')
             CRTCLS     CLS(&OBJLIB/DLVRYCLS) RUNPTY(40) +
                          PURGE(*NO) TEXT('Delivery Job Class')
             CRTSBSD    SBSD(&OBJLIB/BATCHSBS) POOLS((1 *BASE)) +
                          SGNDSPF(QSYS/QDSIGNON)
             ADDJOBQE   SBSD(&OBJLIB/BATCHSBS) +
                          JOBQ(&OBJLIB/DLVRYJOBQ) MAXACT(50) +
                          SEQNBR(1000)
             ADDRTGE    SBSD(&OBJLIB/BATCHSBS) SEQNBR(1000) +
                          CMPVAL(DLVRYJOB) PGM(QSYS/QCMD) +
                          CLS(&OBJLIB/DLVRYCLS)

/* Create the journal receiver in the mirrored user ASP        */

             IF         COND(&STRWHC *EQ '0001') THEN(DO)

             CRTLIB     LIB(&JRNLIB) ASP(&JRNASP)

             CRTMSGQ    MSGQ(&DATLIB/TPCCJRNMSG) TEXT('TPC-C Journal +
                          Hit Threshold Message')
             MONMSG     MSGID(CPF2112)

             CRTJRNRCV  JRNRCV(&JRNLIB/TPCCJRNRCV)              +
                          THRESHOLD(1500000) TEXT('TPCC Journal +
                          Receiver')
             CRTJRNRCV  JRNRCV(&JRNLIB/TEMPRCV)              +
                          TEXT('TPCC Temporary Receiver (only used +
                          in cleanup pgm)')
             CRTJRN     JRN(&DATLIB/TPCCJRN) +
                          JRNRCV(&JRNLIB/TPCCJRNRCV) +
                          MSGQ(&DATLIB/TPCCJRNMSG) +
                          MNGRCV(*SYSTEM) RCVSIZOPT(*MINFIXLEN) +
                          TEXT('TPCC Journal')
             ENDDO

             CHGJRN     JRN(&DATLIB/TPCCJRN) JRNRCV(*GEN) +
                          MSGQ(&DATLIB/TPCCJRNMSG) MNGRCV(*SYSTEM) +
                          RCVSIZOPT(*MINFIXLEN)

/* WAIT FOR COMPLETION OF REORGANIZES                            */

             RCVMSG     MSGQ(&BLDLIB/TPCCBLDMSG) MSGTYPE(*ANY) +
                          WAIT(*MAX)
             RCVMSG     MSGQ(&BLDLIB/TPCCBLDMSG) MSGTYPE(*ANY) +
                          WAIT(*MAX)
             RCVMSG     MSGQ(&BLDLIB/TPCCBLDMSG) MSGTYPE(*ANY) +
                          WAIT(*MAX)
             RCVMSG     MSGQ(&BLDLIB/TPCCBLDMSG) MSGTYPE(*ANY) +
                          WAIT(*MAX)


/* Start journaling                                              */

             CALL       PGM(&BLDLIB/STRJRNTPCC)

/* Save of TPCC Data Files to Save Files.  Used in Cleanup Program */

             IF         COND(&CRTSAVFILE = 'Y') THEN(DO)
               CRTLIB     LIB(&SAVLIB) TEXT(&DATLIB *CAT ' SAVF')
               CLRLIB     LIB(&SAVLIB)
               CRTSAVF    FILE(&SAVLIB/&DATLIB) TEXT(&DATLIB *CAT ' +
                            SAVF')
               SAVLIB     LIB(&DATLIB) DEV(*SAVF) +
                            SAVF(&SAVLIB/&DATLIB) ACCPTH(*YES)
               MONMSG CPF3701 /*IGNORE WHEN CANNOT SAVE MSGQ*/
             ENDDO

/* Submit this job AFTER save is done or save will be incomplete  */

 ENDPGM:     ENDPGM
```

ORDERSVIEW: Creates Orders View Logical Files

```
             PGM        PARM(&DATALIB &SRCLIB &STRWHC &ENDWHC)
             DCL        VAR(&STRWH) TYPE(*DEC) LEN(4 0)
             DCL        VAR(&ENDWH) TYPE(*DEC) LEN(4 0)
             DCL        VAR(&STRWHC) TYPE(*CHAR) LEN(4)
             DCL        VAR(&ENDWHC) TYPE(*CHAR) LEN(4)
             DCL        VAR(&DATALIB) TYPE(*CHAR) LEN(10)
             DCL        VAR(&PARMSTR) TYPE(*CHAR) LEN(18)
             DCL        VAR(&SRCLIB) TYPE(*CHAR) LEN(10)
             /* CREATE THE UNIQUE VIEW OVER ORDERSPF */
             CHKOBJ     OBJ(&DATALIB/ORDERSLF) OBJTYPE(*FILE)
             MONMSG     MSGID(CPF9801) EXEC(CRTLF +
                          FILE(&DATALIB/ORDERSLF) +
                          SRCFILE(&SRCLIB/QDDSSRC) LVLCHK(*NO))
             /* FOR AUDIT-READY DATA BASES, GENERATE CORRECT +
                CUSTOMER ID'S FOR PRE-DEFINED ORDERS */
             CHKOBJ     OBJ(&DATALIB/ORDERS) OBJTYPE(*FILE)
             MONMSG     MSGID(CPF9801) EXEC(CRTLF +
                          FILE(&DATALIB/ORDERS) +
                          SRCFILE(&SRCLIB/QDDSSRC) LVLCHK(*NO))

             ENDPGM
```

ORDLINVIEW: Creates Orders View Logical Files

```
PGM        PARM(&DATALIB &SRCLIB)
DCL        VAR(&DATALIB) TYPE(*CHAR) LEN(10)
DCL        VAR(&SRCLIB) TYPE(*CHAR) LEN(10)
/* CREATE THE UNIQUE VIEW OVER ORDLINPF */
CHKOBJ     OBJ(&DATALIB/ORDLIN) OBJTYPE(*FILE)
MONMSG     MSGID(CPF9801) EXEC(CRTLF +
             FILE(&DATALIB/ORDLIN) +
             SRCFILE(&SRCLIB/QDDSSRC) LVLCHK(*NO))
/* CREATE PARTIAL KEY VIEW */
CHKOBJ     OBJ(&DATALIB/ORDLINLF) OBJTYPE(*FILE)
MONMSG     MSGID(CPF9801) EXEC(CRTLF +
             FILE(&DATALIB/ORDLINLF) +
             SRCFILE(&SRCLIB/QDDSSRC) LVLCHK(*NO))
ENDPGM
```

NEWORDVIEW: Creates New Orders View Logical Files

```
PGM        PARM(&DATALIB &SRCLIB)
DCL        VAR(&DATALIB) TYPE(*CHAR) LEN(10)
DCL        VAR(&SRCLIB) TYPE(*CHAR) LEN(10)
/* CREATE THE UNIQUE VIEW OVER NEWORDPF */
CHKOBJ     OBJ(&DATALIB/NEWORD) OBJTYPE(*FILE)
MONMSG     MSGID(CPF9801) EXEC(CRTLF +
             FILE(&DATALIB/NEWORD) +
             SRCFILE(&SRCLIB/QDDSSRC) LVLCHK(*NO))
/* CREATE PARTIAL KEY VIEW */
CHKOBJ     OBJ(&DATALIB/NEWORDLF) OBJTYPE(*FILE)
MONMSG     MSGID(CPF9801) EXEC(CRTLF +
             FILE(&DATALIB/NEWORDLF) +
             SRCFILE(&SRCLIB/QDDSSRC) LVLCHK(*NO))
ENDPGM
```

CSTMRVIEW: Creates Customer View Logical Files

```
PGM        PARM(&DATALIB &SRCLIB)
DCL        VAR(&DATALIB) TYPE(*CHAR) LEN(10)
DCL        VAR(&SRCLIB) TYPE(*CHAR) LEN(10)
/* CREATE THE LAST/FIRST NAME VIEW OVER CSTMRPF*/
CHKOBJ     OBJ(&DATALIB/CSTMRLFCRT) OBJTYPE(*FILE)
MONMSG     MSGID(CPF9801) EXEC(CRTLF +
             FILE(&DATALIB/CSTMRLFCRT) +
             SRCFILE(&SRCLIB/QDDSSRC) LVLCHK(*NO))
/* CREATE THE VIEW FOR ACCESS BY LAST NAME */
CHKOBJ     OBJ(&DATALIB/CSTMRLFNAM) OBJTYPE(*FILE)
MONMSG CPF9801 EXEC( +
CRTLF      FILE(&DATALIB/CSTMRLFNAM)  +
             SRCFILE(&SRCLIB/QDDSSRC) LVLCHK(*NO))
ENDPGM
```

STRJRNTPCC: Start Journaling for all TPCC Physical Files

```
      PGM

/*********************************************************/
/*                                                       */
/* FUNCTION: THIS PROGRAM STRARTS JOURNALING FOR ALL     */
/*           PHYSICAL FILES FOR THE TPCC WORKLOAD.       */
/*                                                       */
/* INVOCATION:  THIS PROGRAM IS CALLED VIA:              */
/*                                                       */
/*           CALL  PGM(STRJRNTPCC)                       */
/*           CALLED FROM SETUP AND CLEANUP               */
/*                                                       */
/* INPUT:   NONE                                         */
/*                                                       */
/* OUTPUT:  JOURNALING START FOR ALL TPCC FILES          */
/*                                                       */
/* DEPENDENCIES AND ASSUMPTIONS:   NONE                  */
/*                                                       */
/*********************************************************/

/* Monitor for messages                                  */

           MONMSG     MSGID(CPF7032)
           MONMSG     MSGID(CPF7030)

/* Start journaling on the physical files */

           STRJRNPF   FILE(CSTMRPF) +
                        JRN(TPCCJRN) IMAGES(*BOTH) +
                        OMTJRNE(*OPNCLO)
           STRJRNPF   FILE(DSTRCT) JRN(TPCCJRN) +
                        IMAGES(*BOTH) +
                        OMTJRNE(*OPNCLO)
           STRJRNPF   FILE(HSTRY) +
                        JRN(TPCCJRN) IMAGES(*BOTH) +
                        OMTJRNE(*OPNCLO)
           STRJRNPF   FILE(ITEM) JRN(TPCCJRN) +
                        IMAGES(*BOTH) +
                        OMTJRNE(*OPNCLO)
           STRJRNPF   FILE(NEWORDPF) JRN(TPCCJRN) +
                        IMAGES(*BOTH) +
                        OMTJRNE(*OPNCLO)
           STRJRNPF   FILE(ORDERSPF) JRN(TPCCJRN) +
                        IMAGES(*BOTH) +
                        OMTJRNE(*OPNCLO)
           STRJRNPF   FILE(STOCKPF) JRN(TPCCJRN) +
                        IMAGES(*BOTH) +
                        OMTJRNE(*OPNCLO)
           STRJRNPF   FILE(WRHS) JRN(TPCCJRN) +
                        IMAGES(*BOTH) +
                        OMTJRNE(*OPNCLO)
           STRJRNPF   FILE(ORDLINPF) +
                        JRN(TPCCJRN) IMAGES(*BOTH) +
                        OMTJRNE(*OPNCLO)

ENDPGM:    ENDPGM
```

CRTENVPGMS: Create Environment Programs

```
           PGM        PARM(&APPLIB &DATLIB)
           DCL        VAR(&APPLIB) TYPE(*CHAR) LEN(10)
           DCL        VAR(&DATLIB) TYPE(*CHAR) LEN(10)

/* This CL program compiles all of the CL, C and CBL programs   */
/* that are needed to do a complete rebuild of the database.    */
/* They are compiled in the order they are needed. None of the  */
/* programs are called from this program, only compiled.        */

           /* ADD CODE TO BUILD APPP PROGRAMS */
           MONMSG CPF0000
           ADDLIBLE   LIB(&DATLIB)
           MONMSG     MSGID(CPF2103)
           ADDLIBLE   LIB(&APPLIB)
           MONMSG     MSGID(CPF2103)
           CRTCLPGM   PGM(&APPLIB/ATPCCMTRA) SRCFILE(&APPLIB/QCLSRC)
           CRTCLPGM   PGM(&APPLIB/ATPCCMTRD) SRCFILE(&APPLIB/QCLSRC)
           CRTCLPGM   PGM(&APPLIB/ATPCCMTRS) SRCFILE(&APPLIB/QCLSRC)
           CRTSQLCBLI OBJ(&APPLIB/NOPAYOSMTR) +
                        SRCFILE(&APPLIB/QSQLCBLSRC) COMMIT(*ALL) +
                        OPTION(*NOGEN)
           CRTCBLMOD  MODULE(&APPLIB/NOPAYOSMTR) +
                        SRCFILE(QTEMP/QSQLTEMP) +
                        OPTION(*NOMONOPRC) DBGVIEW(*SOURCE) +
                        OPTIMIZE(*FULL) LINKLIT(*PRC)
           CRTPGM     PGM(&APPLIB/NOPAYOSMTR) +
                        MODULE(&APPLIB/NOPAYOSMTR) +
                        BNDSRVPGM(QSYS/QDBRUNHA) ACTGRP(*CALLER)
           CRTCBLMOD  MODULE(&APPLIB/STKLVLMTR) +
                        SRCFILE(&APPLIB/QLBLSRC) +
                        OPTION(*NOMONOPRC) DBGVIEW(*SOURCE) +
                        OPTIMIZE(*FULL) LINKLIT(*PRC)
           CRTPGM     PGM(&APPLIB/STKLVLMTR) +
                        MODULE(&APPLIB/STKLVLMTR) +
                        BNDSRVPGM(QSYS/QDBRUNHA) ACTGRP(*CALLER)
           CRTCBLMOD  MODULE(&APPLIB/DLVRYMTR) +
                        SRCFILE(&APPLIB/QLBLSRC) +
                        OPTION(*NOMONOPRC) DBGVIEW(*SOURCE) +
                        OPTIMIZE(*FULL) LINKLIT(*PRC)
           CRTPGM     PGM(&APPLIB/DLVRYMTR) +
                        MODULE(&APPLIB/DLVRYMTR) +
                        BNDSRVPGM(QSYS/QDBRUNHA) ACTGRP(*CALLER)

           CLRLIB QRPLOBJ

           ENDPGM
```

C_CREATE: Create ITEM Hash

```
#include <stdlib.h>
#include <recio.h>
#include <stddef.h>
#include <string.h>
#include <stdio.h>
#include <errno.h>

/*============================================================*/
/* internal defines                                           */
/*============================================================*/

void main(argc, argv)
register int argc; char *argv[];
{
  long xx;
  char flag;
  char hash_name[10];
  char pf[10];
  char pf_lib[10];
  char lf[10];
  char lf_lib[10];
  char expression[255];

  typedef _Packed struct key_struct {
                        char name[10];
                        long value;
     } key_ranges_structure[5];

  key_ranges_structure key_ranges;

  flag = '1';
  memcpy(hash_name ,"ITEMLF    ",10);
  memcpy(pf        ,"ITEM      ",10);
  memcpy(lf        ,"ITEMLF    ",10);
  memcpy(pf_lib    ,"TPCCD698  ",10);
  memcpy(lf_lib    ,"TPCCD698  ",10);
  pf_lib[10] = '\0';
  lf_lib[10] = '\0';
  memcpy(expression,"*DFT                          ",33);
  expression[33] = '\0';

  key_ranges[0].value = 100000;

  xx=qdbcrtha(flag,hash_name,pf,pf_lib,
              lf,lf_lib,key_ranges,expression);
```

```
  xx =0;

}
```

C_CREATE2: Create STOCK Hash

```
#include <stdlib.h>
#include <recio.h>
#include <stddef.h>
#include <string.h>
#include <stdio.h>
#include <errno.h>

/*============================================================*/
/* internal defines                                           */
/*============================================================*/

void main(argc, argv)
register int argc; char *argv[];
{
  long xx;
  char flag;
  char hash_name[10];
  char pf[10];
  char pf_lib[10];
  char lf[10];
  char lf_lib[10];
  char expression[255];

  typedef _Packed struct key_struct {
                       char name[10];
                       long value;
     } key_ranges_structure[5];

  key_ranges_structure key_ranges;

  flag = '0';
  memcpy(hash_name ,"STOCK     ",10);
  memcpy(pf        ,"STOCKPF   ",10);
  memcpy(lf        ,"STOCK     ",10);
  memcpy(pf_lib    ,"TPCCD698  ",10);
  memcpy(lf_lib    ,"TPCCD698  ",10);
  pf_lib[10] = '\0';
  lf_lib[10] = '\0';
  memcpy(expression,"*DFT                          ",33);
  expression[33] = '\0';

  key_ranges[0].value = 3700;
  key_ranges[1].value = 100000;

  xx=qdbcrtha(flag,hash_name,pf,pf_lib,
              lf,lf_lib,key_ranges,expression);

  xx =0;

}
```

C_CREATE3: Create CSTMR Hash

```
#include <stdlib.h>
#include <recio.h>
#include <stddef.h>
#include <string.h>
#include <stdio.h>
#include <errno.h>

/*============================================================*/
/* internal defines                                           */
/*============================================================*/

void main(argc, argv)
register int argc; char *argv[];
{
  long xx;
  char flag;
  char hash_name[10];
  char pf[10];
  char pf_lib[10];
  char lf[10];
  char lf_lib[10];
  char expression[255];

  typedef _Packed struct key_struct {
                       char name[10];
                       long value;
     } key_ranges_structure[5];

  key_ranges_structure key_ranges;

  flag = '0';
  memcpy(hash_name ,"CSTMR     ",10);
  memcpy(pf        ,"CSTMRPF   ",10);
  memcpy(lf        ,"CSTMR     ",10);
  memcpy(pf_lib    ,"TPCCD698  ",10);
  memcpy(lf_lib    ,"TPCCD698  ",10);
  pf_lib[10] = '\0';
  lf_lib[10] = '\0';
  memcpy(expression,"*DFT                          ",33);
  expression[33] = '\0';

  key_ranges[0].value = 3000;
  key_ranges[1].value = 10;
  key_ranges[2].value = 3700;

  xx=qdbcrtha(flag,hash_name,pf,pf_lib,
              lf,lf_lib,key_ranges,expression);

  xx =0;

}
```

Appendix D. Application Source Code

Program Flow.

CLIENT:

COMMON.C:

```
#define _COMMON_C
#include "Common.h"

/* Establish a listening port */
int Listen(int port) {
  struct sockaddr_in    sin;
  int                   sd, on=1;

  /* Get a socket descriptor */
  if ((sd=socket(AF_INET,SOCK_STREAM,0))<0) {
    perror("socket() failed");
    exit(-1);
  }

  /* Allow socket descriptor to be reuseable */
  if (setsockopt(
                 sd,
                 SOL_SOCKET,
                 SO_REUSEADDR,
                 (char *)&on,
                 sizeof(on)
                 )<0) {
    perror("setsockopt() failed");
    close(sd);
    exit(-1);
  }

  /* bind to an address */
  memset(&sin, 0x00, sizeof(struct sockaddr_in));
  sin.sin_family        =AF_INET;
  sin.sin_port          =htons(port);
  sin.sin_addr.s_addr   =htonl(INADDR_ANY);

  if(bind(sd,(struct sockaddr *)&sin,sizeof(sin))<0) {
    perror("bind() failed");
    close(sd);
    exit(-1);
  }

  if(listen(sd,500)<0) {
    perror("listen() failed");
    close(sd);
    exit(-1);
  }

  return sd;
}


/* Establish connection to specified address */
int Connect(int ipAddr, int port) {
  struct sockaddr_in    sin;
  int                   sd;

  /* Get a socket descriptor */
  if ((sd=socket(AF_INET,SOCK_STREAM,0))<0) {
    perror("socket() failed");
    exit(-1);
  }

  memset(&sin, 0x00, sizeof(struct sockaddr_in));
  sin.sin_family        =AF_INET;
  sin.sin_port          =htons(port);
  sin.sin_addr.s_addr   =htonl(ipAddr);

  if(connect(sd,(struct sockaddr *)&sin,sizeof(sin))<0) {
    perror("connect() failed");
    close(sd);
    exit(-1);
  }

  return sd;
}
```

COMMON.H:

```
#define _COMMON_C
#include "Common.h"

/* Establish a listening port */
int Listen(int port) {
  struct sockaddr_in    sin;
  int                   sd, on=1;

  /* Get a socket descriptor */
  if ((sd=socket(AF_INET,SOCK_STREAM,0))<0) {
    perror("socket() failed");
    exit(-1);
  }

  /* Allow socket descriptor to be reuseable */
  if (setsockopt(
                 sd,
                 SOL_SOCKET,
                 SO_REUSEADDR,
                 (char *)&on,
                 sizeof(on)
                 )<0) {
    perror("setsockopt() failed");
    close(sd);
    exit(-1);
  }

  /* bind to an address */
  memset(&sin, 0x00, sizeof(struct sockaddr_in));
  sin.sin_family        =AF_INET;
  sin.sin_port          =htons(port);
  sin.sin_addr.s_addr   =htonl(INADDR_ANY);

  if(bind(sd,(struct sockaddr *)&sin,sizeof(sin))<0) {
    perror("bind() failed");
    close(sd);
    exit(-1);
  }

  if(listen(sd,500)<0) {
    perror("listen() failed");
    close(sd);
    exit(-1);
  }

  return sd;
}


/* Establish connection to specified address */
int Connect(int ipAddr, int port) {
  struct sockaddr_in    sin;
  int                   sd;

  /* Get a socket descriptor */
  if ((sd=socket(AF_INET,SOCK_STREAM,0))<0) {
    perror("socket() failed");
    exit(-1);
  }

  memset(&sin, 0x00, sizeof(struct sockaddr_in));
  sin.sin_family        =AF_INET;
  sin.sin_port          =htons(port);
  sin.sin_addr.s_addr   =htonl(ipAddr);

  if(connect(sd,(struct sockaddr *)&sin,sizeof(sin))<0) {
    perror("connect() failed");
    close(sd);
    exit(-1);
  }

  return sd;
}
```

DELVRYSRV.C:

```
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <atmi.h>       /* TUXEDO Header File */
#include <string.h>
#include <recio.h>
#include <xxfdbk.h>
#include <userlog.h>    /* TUXEDO Header File */
#include <decimal.h>
#include <quscrtus.h>
#include <qusptrus.h>
#include <qusgen.h>
#include <qp0wpid.h>
#include "common.h"

/* tpsvrinit is executed when a server is booted, before it begins
   processing requests.  It is not necessary to have this function.
   Also available is tpsvrdone (not used in this example), which is
   called at server shutdown time.
*/



   static int sd, addr=0, port=7000;
   dlv_input di;
   int rc=670;

tpsvrinit(int argc, char *argv[])
{
        /* Some compilers warn if argc and argv aren't used. */
        argc = argc;
        argv = argv;

        /* userlog writes to the central TUXEDO message log */
        userlog("Welcome to the tpcc server");
    addr=inet_addr("9.5.149.167");
    sd=Connect(addr,port);
        return(0);
}

/* This function performs the actual service requested by the client.
   Its argument is a structure containing among other things a pointer
   to the data buffer, and the length of the data buffer.
*/



     /*       DELIVERY       */

DELIVRY(TPSVCINFO *rqst)
{
    memcpy(&di, rqst->data, 230);

    if(send(sd, (char *) &di,230,0)<0) {
      perror("\nsend() failed");
      tpreturn(TPFAIL, 0, rqst->data, 0L, 0);
    }


    /* Return the transformed buffer to the requestor. */
    tpreturn(TPSUCCESS, 0, rqst->data, rc, 0);
}
```

FORMCHILDS.C:

```
#include "Common.h"

 char FrmSrvCgiPath[]="/usr/bin/FormSerCgi";


void main(int argc, char *argv[]) {
  struct inheritance    spw_inherit;
  char                  *childArgv[5], shmid_s[20];
  char                  *dumy=NULL;
  int                   sd, sd2, shmid;
  pid_t                 child;
  struct sockaddr_in    sin;
  int numjobs, i , length;
  char numjobs_s[5];
  char cmd[50];
shmid = atoi(argv[1]);
numjobs = atoi(argv[2]);
memcpy(cmd,"DLYJOB DLY(300)", 17);
cmd[49] = '\0';
```

```
/* system(cmd); */

  sprintf(shmid_s,"%i",shmid);
  sprintf(numjobs_s,"%i",numjobs);
  childArgv[0]=shmid_s;
  childArgv[1]=shmid_s;
  childArgv[2]=numjobs_s;
  childArgv[3]=NULL;

   memset(&spw_inherit,0x00,sizeof(spw_inherit));

   length=sizeof(sin);

   for(i=1; i<=numjobs; i++) {

   if(child=(spawn(
                   FrmSrvCgiPath,
                   1,
                   &sd,
                   &spw_inherit,
                   childArgv,
                   &dumy
                   ))<0) {
      perror("spawn() failed");
      close(sd);
      exit(-1);
    }

  }

  close(sd);

  free((void *)&child);
  exit(0);
}
```

FORMPARENT.C:

```
#include "Common.h"

char FrmChildPath[]="/usr/bin/FormChilds";
static int numchildjobs=4;

#pragma convert(819)

const char *FORM[NUM_FORMS]={

    "<HTML>"
      "<HEAD><TITLE>Welcome To TPC-C</TITLE></HEAD><BODY><center><h2>"
      "Please Identify your Warehouse and District for this session.<BR>"
      "</h2></center><FORM ACTION=\"tpcc.dll\" METHOD=\"GET\">"
      "<INPUT TYPE=\"hidden\" NAME=\"FORMID\" VALUE=\"1\">"
      "<INPUT TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"-2\">"
      "<INPUT TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"0\">"
      "Warehouse ID <INPUT NAME=\"w_id\" SIZE=4 MAXLENGTH=4><BR>"
      "District ID <INPUT NAME=\"d_id\" SIZE=2 MAXLENGTH=2><BR>"
      "<HR>"
      "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"Submit\">"
      "</FORM><BODY>"
      "</HTML>",

    "<HTML><HEAD><TITLE>TPC-C Main Menu</TITLE></HEAD><BODY>"
    "<center><h2>Select Desired Transaction.</h2></center><BR><HR>"
    "<FORM ACTION=\"tpcc.dll\" METHOD=\"GET\">"
    "<INPUT TYPE=\"hidden\" NAME=\"FORMID\" VALUE=\"2\">"
    "<INPUT TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"?   \">"
    "<INPUT TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"? \">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..NewOrder..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Payment..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Delivery..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Order-Status..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Stock-Level..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Exit..\">"
    "</FORM></BODY>"
    "</HTML>",


    "<HTML><HEAD><TITLE>TPC-C New Order</TITLE></HEAD>"
    "<body>"
    "<center><h1>TPC-C New Order</h1></center>"
    "<FORM ACTION=\"tpcc.dll\" METHOD=\"GET\">"
    "<INPUT TYPE=\"hidden\" NAME=\"PI*\" VALUE=\"\">"
    "<INPUT TYPE=\"hidden\" NAME=\"STATUSID\" VALUE=\"0\">"
    "<INPUT TYPE=\"hidden\" NAME=\"ERROR\" VALUE=\"0\">"
    "<INPUT TYPE=\"hidden\" NAME=\"FORMID\" VALUE=\"3\">"
    "<INPUT TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"?   \">"
    "<INPUT TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=? >"
    "<pre>"
    "Warehouse: ?     "
    "District: <input name=\"D_ID\" size=2 MAXLENGTH=2>"
    "<br>Customer: <input name=\"C_ID\" size=4 MAXLENGTH=4>"
    "</pre>"
    "<p>"
    "<table>"
    "<tr>"
    "<th>Supp_w</th>"
    "<th>Item_Id</th>"
    "<th>Qty</th>"
    "<th>Item Name</th>"
    "<th>Stock</th>"
    "<th>B/G</th>"
    "<th>Price</th>"
    "<th>Amount</th>"
    "</tr>"
    "<tr>"
    "<td>"
    "<input name=\"S00\" size=4 MAXLENGTH=4>"
    "</td>"
    "<td>"
    "<input name=\"I00\" size=6 MAXLENGTH=6>"
    "</td>"
    "<td>"
    "<input name=\"Q00\" size=2 MAXLENGTH=2>"
    "</td>"
    "<td colspan=5>"
    "</td>"
    "</tr>"
    "<tr>"
    "<td>"
    "<input name=\"S01\" size=4 MAXLENGTH=4>"
    "</td>"
    "<td>"
    "<input name=\"I01\" size=6 MAXLENGTH=6>"
    "</td>"
    "<td>"
    "<input name=\"Q01\" size=2 MAXLENGTH=2>"
    "</td>"
    "<td colspan=5>"
    "</td>"
    "</tr>"
    "<tr>"
    "<td>"
    "<input name=\"S02\" size=4 MAXLENGTH=4>"
    "</td>"
    "<td>"
    "<input name=\"I02\" size=6 MAXLENGTH=6>"
    "</td>"
    "<td>"
    "<input name=\"Q02\" size=2 MAXLENGTH=2>"
    "</td>"
    "<td colspan=5>"
    "</td>"
    "</tr>"
    "<tr>"
    "<td>"
    "<input name=\"S03\" size=4 MAXLENGTH=4>"
    "</td>"
    "<td>"
    "<input name=\"I03\" size=6 MAXLENGTH=6>"
    "</td>"
    "<td>"
    "<input name=\"Q03\" size=2 MAXLENGTH=2>"
    "</td>"
    "<td colspan=5>"
    "</td>"
    "</tr>"
    "<tr>"
    "<td>"
    "<input name=\"S04\" size=4 MAXLENGTH=4>"
    "</td>"
    "<td>"
    "<input name=\"I04\" size=6 MAXLENGTH=6>"
    "</td>"
    "<td>"
    "<input name=\"Q04\" size=2 MAXLENGTH=2>"
    "</td>"
    "<td colspan=5>"
    "</td>"
    "</tr>"
    "<tr>"
    "<td>"
    "<input name=\"S05\" size=4 MAXLENGTH=4>"
    "</td>"
    "<td>"
    "<input name=\"I05\" size=6 MAXLENGTH=6>"
    "</td>"
    "<td>"
    "<input name=\"Q05\" size=2 MAXLENGTH=2>"
    "</td>"
    "<td colspan=5>"
    "</td>"
    "</tr>"
    "<tr>"
    "<td>"
    "<input name=\"S06\" size=4 MAXLENGTH=4>"
    "</td>"
    "<td>"
    "<input name=\"I06\" size=6 MAXLENGTH=6>"
    "</td>"
    "<td>"
    "<input name=\"Q06\" size=2 MAXLENGTH=2>"
    "</td>"
    "<td colspan=5>"
    "</td>"
    "</tr>"
    "<tr>"
    "<td>"
    "<input name=\"S07\" size=4 MAXLENGTH=4>"
    "</td>"
    "<td>"
    "<input name=\"I07\" size=6 MAXLENGTH=6>"
    "</td>"
    "<td>"
    "<input name=\"Q07\" size=2 MAXLENGTH=2>"
    "</td>"
    "<td colspan=5>"
    "</td>"
    "</tr>"
    "<tr>"
    "<td>"
    "<input name=\"S08\" size=4 MAXLENGTH=4>"
    "</td>"
    "<td>"
    "<input name=\"I08\" size=6 MAXLENGTH=6>"
    "</td>"
    "<td>"
    "<input name=\"Q08\" size=2 MAXLENGTH=2>"
    "</td>"
    "<td colspan=5>"
    "</td>"
    "</tr>"
    "<tr>"
    "<td>"
    "<input name=\"S09\" size=4 MAXLENGTH=4>"
    "</td>"
    "<td>"
    "<input name=\"I09\" size=6 MAXLENGTH=6>"
    "</td>"
    "<td>"
    "<input name=\"Q09\" size=2 MAXLENGTH=2>"
    "</td>"
    "<td colspan=5>"
    "</td>"
    "</tr>"
    "<tr>"
    "<td>"
    "<input name=\"S10\" size=4 MAXLENGTH=4>"
    "</td>"
    "<td>"
    "<input name=\"I10\" size=6 MAXLENGTH=6>"
    "</td>"
    "<td>"
    "<input name=\"Q10\" size=2 MAXLENGTH=2>"
    "</td>"
    "<td colspan=5>"
    "</td>"
    "</tr>"
    "<tr>"
    "<td>"
    "<input name=\"S11\" size=4 MAXLENGTH=4>"
    "</td>"
    "<td>"
    "<input name=\"I11\" size=6 MAXLENGTH=6>"
    "</td>"
    "<td>"
    "<input name=\"Q11\" size=2 MAXLENGTH=2>"
    "</td>"
    "<td colspan=5>"
    "</td>"
    "</tr>"
    "<tr>"
    "<td>"
    "<input name=\"S12\" size=4 MAXLENGTH=4>"
    "</td>"
```

```
    "<td>"
    "<input name=\"I12\" size=6 MAXLENGTH=6>"
    "</td>"
    "<td>"
    "<input name=\"Q12\" size=2 MAXLENGTH=2>"
    "</td>"
    "<td colspan=5>"
    "</td>"
    "</tr>"
    "<tr>"
    "<td>"
    "<input name=\"S13\" size=4 MAXLENGTH=4>"
    "</td>"
    "<td>"
    "<input name=\"I13\" size=6 MAXLENGTH=6>"
    "</td>"
    "<td>"
    "<input name=\"Q13\" size=2 MAXLENGTH=2>"
    "</td>"
    "<td colspan=5>"
    "</td>"
    "</tr>"
    "<tr>"
    "<td>"
    "<input name=\"S14\" size=4 MAXLENGTH=4>"
    "</td>"
    "<td>"
    "<input name=\"I14\" size=6 MAXLENGTH=6>"
    "</td>"
    "<td>"
    "<input name=\"Q14\" size=2 MAXLENGTH=2>"
    "</td>"
    "<td colspan=5>"
    "</td>"
    "</tr>"
    "</table>"
    "<p>"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"Process\">"
    "</form>"
    "</body>"
    "</HTML>",

    "<HTML><HEAD><TITLE>TPC-C Payment</TITLE></HEAD>"
    "<body>"
    "<center><h1>TPC-C Payment</h1></center>"
    "<FORM ACTION=\"tpcc.dll\" METHOD=\"GET\">"
    "<INPUT TYPE=\"hidden\" NAME=\"PI*\" VALUE=\"\">"
    "<INPUT TYPE=\"hidden\" NAME=\"STATUSID\" VALUE=\"0\">"
    "<INPUT TYPE=\"hidden\" NAME=\"ERROR\" VALUE=\"0\">"
    "<INPUT TYPE=\"hidden\" NAME=\"FORMID\" VALUE=\"4\">"
    "<INPUT TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"?   \">"
    "<INPUT TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"? \">"
    "<pre>"
    "Date: @                    <br><br>"
    "Warehouse: ?           "
    "District: <input name=\"D_ID\" size=2 MAXLENGTH=2><br>"
    "Customer: <input name=\"C_ID\" size=4 MAXLENGTH=4>  "
    "Cust-Warehouse: <input name=\"C_W_ID\" size=4 MAXLENGTH=4>  "    "Cust-District: <input
name=\"C_D_ID\" size=1 MAXLENGTH=2><br>"
    "Name:    <input name=\"LNAME\" size=16 MAXLENGTH=16 >     Since:<br>"
    "                             Credit:<br>"
    "                             Disc:<br>"
    "                             Phone:<br><br>"
    "Amount Paid:      $<input name=\"PAID\" size=7 MAXLENGTH=7>"
    "     New Cust Balance:<br>"
    "Credit Limit:<br><br>"
    "Cust-Data:<br><br><br><br>"
    "</pre>"
    "<p>"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"Process\">"
    "</form>"
    "</body>"
    "</HTML>",

    "<HTML><HEAD><TITLE>TPC-C Delivery</TITLE></HEAD>"
    "<body>"
    "<center><h1>TPC-C Delivery</h1></center>"
    "<FORM ACTION=\"tpcc.dll\" METHOD=\"GET\">"
    "<INPUT TYPE=\"hidden\" NAME=\"PI*\" VALUE=\"\">"
    "<INPUT TYPE=\"hidden\" NAME=\"STATUSID\" VALUE=\"0\">"
    "<INPUT TYPE=\"hidden\" NAME=\"ERROR\" VALUE=\"0\">"
    "<INPUT TYPE=\"hidden\" NAME=\"FORMID\" VALUE=\"5\">"
    "<INPUT TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"?   \">"
    "<INPUT TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"? \">"
    "<pre>"
    "Warehouse: ?           "
    "<br>                   "
    "<br>Carrier Number: <INPUT TYPE=TEXT SIZE=2 MAXLENGTH=2 NAME=\"CARRIER\">"
    "<br>                   "
    "<br>Execution Status:"
    "</pre>"
    "<p>"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"Process\">"
    "</form>"
    "</body>"
    "</HTML>",

    "<HTML><HEAD><TITLE>TPC-C Order-Status</TITLE></HEAD>"
    "<body>"
    "<center><h1>TPC-C Order-Status</h1></center>"
    "<FORM ACTION=\"tpcc.dll\" METHOD=\"GET\">"
    "<INPUT TYPE=\"hidden\" NAME=\"PI*\" VALUE=\"\">"
    "<INPUT TYPE=\"hidden\" NAME=\"STATUSID\" VALUE=\"0\">"
    "<INPUT TYPE=\"hidden\" NAME=\"ERROR\" VALUE=\"0\">"
    "<INPUT TYPE=\"hidden\" NAME=\"FORMID\" VALUE=\"6\">"
    "<INPUT TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"?   \">"
    "<INPUT TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"? \">"
    "<pre>"
    "Warehouse: ?           "
    "District: <input name=\"D_ID\" size=2 MAXLENGTH=2><br>"
    "Customer: <input name=\"C_ID\" size=4 MAXLENGTH=4>"
    "Name: <input name=\"LNAME\" size =16 MAXLENGTH=16><br>"
    "Cust-Balance: <br><br>"
    "Order-Number:    "
    "Entry-Date:   "
    "Carrier-Number: <br>"
    "</pre>"
    "<p>"
    "<table>"
    "<tr>"
    "<th>Supply_W</th>"
    "<th>Item_Id</th>"
    "<th>Qty</th>"
    "<th>Amount</th>"
    "<th>Delivery-Date</th>"
    "</tr>"
    "</table>"
    "<p>"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"Process\">"
    "</form>"
    "</body>"
    "</HTML>",

    "<HTML><HEAD><TITLE>TPC-C Stock Level</TITLE></HEAD>"
    "<body>"
    "<center><h1>TPC-C Stock Level</h1></center>"
    "<FORM ACTION=\"tpcc.dll\" METHOD=\"GET\">"
    "<INPUT TYPE=\"hidden\" NAME=\"PI*\" VALUE=\"\">"
    "<INPUT TYPE=\"hidden\" NAME=\"STATUSID\" VALUE=\"0\">"
    "<INPUT TYPE=\"hidden\" NAME=\"ERROR\" VALUE=\"0\">"
    "<INPUT TYPE=\"hidden\" NAME=\"FORMID\" VALUE=\"7\">"
    "<INPUT TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"?   \">"
    "<INPUT TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"? \">"
    "<pre>"
    "Warehouse: ?           "
    "<br>                   "
    "<br>Stock Level Threshold: \
    <INPUT TYPE=TEXT SIZE=2 MAXLENGTH=2 NAME=\"THRESHOLD\">"
    "<br>                   "
    "<br>Low Stock:"
    "</pre>"
    "<p>"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"Process\">"
    "</form>"
    "</body>"
    "</HTML>",

    "<HTML>"
    "<HEAD><TITLE>Welcome To TPC-C</TITLE></HEAD><BODY><center><h2>"    "The Warehouse and
District you entered are not valid.  Please try again.<BR>"
    "</h2></center><FORM ACTION=\"tpcc.dll\" METHOD=\"GET\">"
    "<INPUT TYPE=\"hidden\" NAME=\"FORMID\" VALUE=\"1\">"
    "<INPUT TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"-2\">"
    "<INPUT TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"0\">"
    "Warehouse ID <INPUT NAME=\"w_id\" SIZE=4 MAXLENGTH=4><BR>"
    "District ID <INPUT NAME=\"d_id\" SIZE=2 MAXLENGTH=2><BR>"
    "<HR>"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"Submit\">"
    "</FORM></BODY>"
    "</HTML>",

    "<HTML>"
    "<HEAD><TITLE>TPC-C data not found error</TITLE></HEAD><BODY>"
    "<center><H2>TPC-C Benchmark application data not found.</h2><BR>"
    "<H3>ERROR 404</h3></center><BR>"
    "</BODY>"
    "</HTML>",

    "<HTML>"
    "<HEAD><TITLE>TPC-C application exit </TITLE></HEAD><BODY>"
    "<center><H2>TPC-C Benchmark application has ended.</h2><BR>"
    "<H3>GOOD-BYE!</h2></center><BR>"
    "</BODY>"
    "</HTML>"
};
#pragma convert(0)

#define NUM_SVRS 5



void main(int argc, char *argv[]) {
    struct inheritance    spw_inherit;
    char                  *childArgv[4], shmid_s[20];
    char                  *dumy=NULL, *base;
    char                  buffer[BufferSize];
    char              warehouse[WarehouseFieldSize], district[DistrictFieldSize];
    char                  *hiddenWid, *hiddenDid, *contLen;
    int                   status, sd, sd2, shmid;
    int                   formPort=80;
    int                   header_size; /* header size          */
    int                   content_size;/* size of the contents */
    int                   i, j, length;
    Config_t              *config;
    pid_t                 child;
    struct                sockaddr_in    sin;
    int                   numusers, numchildjobs, numformchilds;
    char                  numchildjobs_s[5];


    numusers = atoi(argv[1]);
    Qp0zInitEnv();

  /* Determine amount of shared memory required */
    length=sizeof(Config_t)
      +((NUM_TRANS+2)*strlen(HEADER))
      +16; /* Slop */

    for(i=0;i<NUM_FORMS;++i) length+=strlen(FORM[i]);

    if((shmid=shmget(
                                IPC_PRIVATE,
                                length,
                                S_IRUSR|S_IWUSR
                                ))<0) {
              perror("\nshmget() failed");
              exit(1);
    }

    if((config=shmat(
                                shmid,
                                NULL,
                                0
                                ))==NULL) {
              perror("\nshmat() failed");
              exit(1);
    }


    sprintf(shmid_s,"%i",shmid);
    childArgv[0]=shmid_s;
    childArgv[1]=shmid_s;
    childArgv[3]=NULL;

    base=(char *)config+sizeof(Config_t);


  /* Create operation forms and spawn the jobs     */

    for(i=0;i<NUM_FORMS;++i) {
              if ( i!= NOT_FOUND_ERROR )
              {
                  if ( i != EXIT )
                  {
                            sprintf(base,"%s%s",HEADER,FORM[i]);
                            header_size = strlen( HEADER );
                  }
                  else
                  {
                            sprintf(base,"%s%s",EXITHEADER,FORM[i]);
                            header_size = strlen( EXITHEADER );
                  }
              }
              else
              {
                  sprintf(base,"%s%s",ERRORHEADER,FORM[i]);
                  header_size = strlen( ERRORHEADER );
              }
```

```
    /* Locate areas containing content length, and hidden w_id/d_id */

              contLen=strchr(base,delimiter);

              if (( i!= WELCOME )&&(i!= NOT_FOUND_ERROR)&&(i!=INPUT_ERROR)&&
                  ( i != MENU )&&( i != EXIT ))
              {
                  config->form[i].hiddenWid=strchr(contLen+1,delimiter);
                  config->form[i].hiddenDid=strchr(config->form[i].hiddenWid+1,delimiter);
                  config->form[i].wid=strchr(config->form[i].hiddenDid+1,delimiter);
              }
              else
              {
                  if ( i != MENU )
                  {
                            contLen=strchr(base,delimiter);
                            config->form[i].hiddenWid=NULL;
                            config->form[i].hiddenDid=NULL;
                            config->form[i].wid=NULL;
                  }
                  else /* the menu form */
                  {
                            contLen=strchr(base,delimiter);/* first ? */
                            config->form[i].hiddenWid=strchr(contLen+1,delimiter);/* next
?*/                         config->form[i].hiddenDid=
                              strchr(config->form[i].hiddenWid+1,delimiter);
                           config->form[i].wid=NULL;
                         /* next ? in the form */
                  }
              }
              config->form[i].addr=base;
              config->form[i].size=strlen(config->form[i].addr);
              content_size = config->form[i].size- header_size;

    /* Poke in the content length */
              binToIA5(content_size,contLen);
              config->form[i].date_location=strchr(base,date_marker);
              base=base+config->form[i].size;
    } /* end of the form generating section */


    sd=Listen(formPort);

    memset(&spw_inherit,0x00,sizeof(spw_inherit));

    length=sizeof(sin);

    numformchilds = numusers / numchildjobs;

    if ( numformchilds == 0) {
              numformchilds=1;
              numchildjobs=1;
    }

    sprintf(numchildjobs_s, "%i", numchildjobs);

    childArgv[2]=numchildjobs_s;


    for(i=1; i<=numformchilds; i++) {

              if(child=(spawn(
                                        FrmChildPath,
                                        1,
                                        &sd,
                                        &spw_inherit,
                                        childArgv,
                                        &dumy
                                        ))<0) {
                  perror("spawn() failed");
                  close(sd);
                  exit(-1);
              }

    }

    close(sd);

    free((void *)&child);
    shmdt(config);
    exit(0);
}
```

FORMSERVER.C:

```
#include "Common.h"
#include "atmi.h"
#include "uunix.h"

#define DATA_ERROR 30
#define SIGNON_ERROR 31
/*****************************************************************/
/*                                                               */
/* External functions                                            */
/* External functions must be linked in                          */
/*                                                               */
/*****************************************************************/
int scan_new_order(char * input, new_order_input * output);
int scan_payment(char * input, payment_input * output);
int scan_order_status(char * input, order_status_input * output);
int scan_delivery(char * input, dlv_input * output);
int scan_stock_level(char * input, stock_input * output);
int getdttmstr(char * timein, char* timestamp, char type);

#pragma convert(819)

 const char DASH='-';         /* dash for the time                        */
 const char COLON=':';        /* get the colon in asci for the time       */
 const char NINE='9';         /* Ascii character nine for scaning input   */
 const char ONE='1';          /* For the form numbers                     */
 const char ASCI_3 = '3';     /* For the form numbers                     */
 const char INPUT_BLANK = '+';/* blank inserted character from brawser    */
 const char ASCI_C = 'C';       /* ASCII C used in CMD                    */
 const char ASCI_N = 'N';       /* ASCII N for the new order transaction  */
 const char ASCI_P = 'P';       /* ASCII P for the payment transcation    */
 const char ASCI_O = 'O';       /* ASCII O for the order status transaction  */
 const char ASCI_D = 'D';       /* ASCII D for the delivery transaction   */
 const char ASCI_S = 'S';       /* ASCII S for the stock level transaction   */
 const char ASCI_E = 'E';       /* ASCII E for the exit action selection  */
 const char ASCI_F = 'F';       /* ASCII E for the exit action selection  */
 const char ASCI_W = 'W';       /* ASCII W for the welcome screen         */
 const char ASCI_w = 'w';       /* ASCII w for the welcome screen         */

 const char GET_STRING[]="GET /";/* GET for form action                   */
 const char TEST_DATA[]= "tpcc.dll?";/* First part of the input           */
 const char WELCOME_DATA[]="welcome.html "; /* welcome string             */
 const char STATUSID[]= "STATUSID=";     /* Status ID text                */
 const char COMMNAD[] = "CMD=";          /* Command string                */
 const char TERMID[]  = "TERMID=";       /* Terminal ID text              */
 const char SYNCID[]  = "SYNCID=";       /* Sync ID text                  */
 const char ERROR[]   = "ERROR=";        /* Error text part of the data   */
 char FORMID[]        = "FORMID=";       /* Form ID text                  */
 char WAREHOUSEID[]   = "w_id=";         /* warehouse number text         */
 char DISTRICTID[]    = "d_id=";         /* district number               */
 char CMDSTRING[]     = "CMD=..";        /* command for start             */


#pragma convert(0)

  char          *date_loc;/* pointer to insert the date and time */
  char warehousebWarehouseFieldSizec, districtbDistrictFieldSizec;
  char          *writePtr;
  Config_t *config;
  int           writeLen;
  int           form_number;/* results form ID */
  char          *hiddenWid, *hiddenDid;
  char          *wid;/* warehouse insert location */
  char          cl_timeb27c; /* converted output return              */

  iconv_t         toEBCDIC, toIA5;
  char            scratchBuffer[BufferSize];
  char            buffer[BufferSize];

  new_order_display       nd;
  payment_display         pd;
  order_status_display    od;
  dlv_input               *di;
  stock_display          s_d;
  struct sockaddr_in    sin;

/*****************************************************************//*  Internal procedures
                                        *//*
          *//*****************************************************************/int
set_error_report( int status);
char scan_strings  ( char ** input_string, char * output_loc, int size,
                     char   terminator );
char parse_input ( char * input, char ** new_location );

void process_normal  ( char * temp );
void process_exit    ( char * temp );
void process_start   ( void );
void display_error   ( void );
void copy_date       ( void );
void initTux         ( void );
void process_input   ( char * data_input );
int postResult(void *res_buf, char* warehouse, char * district, int form,
               char *scratchBuffer );

void main(int argc, char *argv[]) {
  int  sd, sd2, i, j, length, termId=0;
  char *temp;
  char          *contLen, *base, *actionLoc;
  char          *new_ptr;/* location of the data after parsing   */
  char          data_error;/* data error flag                     */
  int           contentLen;

  for(i=0; i < BufferSize; i++) buffer[i] = '\0';
  for(i=0; i < WarehouseFieldSize; i++) warehouse[i] = '\0';
  for(i=0; i < DistrictFieldSize; i++) district[i] = '\0';
  /* Get access to shared memory */
  if((config=shmat(
                   atoi(argv[1]),
                   NULL,
                   0
                  ))==NULL) {
    perror("shmat() failed");
    close(0);
    exit(-1);
  }


    length=sizeof(sin);
    if((sd2=accept(sd,(struct sockaddr *)&sin,&length))<0) {
      perror("accept() failed");
      close(sd);
      exit(-1);
    }


      initTux(); /* setup the tuxedo functions and buffers */
      for(;;) {
        *((int *)warehouse)    =0x20202020;/* Prime warehouse identifier */
        *((short *)district)   =0x2020;    /* Prime district identifier */


      if((length=recv(sd2,buffer,BufferRecvSize,0))<1) {
        perror("recv() failed");
        break;
      }
      else {
               bufferblengthc='\0';

      temp = buffer;/* set up input pointer                          */
      data_error = parse_input( temp, &new_ptr );
      if ( data_error == 'N' )/* if the input data is valid  */
      {
         temp = new_ptr;/* go to the start of command */
         if ( *temp == ASCI_F )/* if the data is FORMID=        */
         {
             process_normal( temp );
             if (( form_number != NOT_FOUND_ERROR )&&
                ( form_number != WELCOME )&&
                ( form_number != INPUT_ERROR )&&
                ( form_number != EXIT ))
             {
             /* Poke the warehouse id and district id in the form */
                for(i=0;i<WarehouseFieldSize;++i)
                {
                    hiddenWidbic=warehousebic;
                }
                if ( form_number != MENU )
                {
                   for(i=0;i<WarehouseFieldSize;++i)
                   {
                      widbic=warehousebic;
                   }
                }
                for(i=0;i<DistrictFieldSize;++i)
                {
                    hiddenDidbic=districtbic;
                }
             } /* end of form_number test */
         }    /* end of if ASCI_F */
         else
         {
             if (*temp == ASCI_C )/* text CMD=Begin */
             {
                 process_start ( );
             }
             else
             {
                 if (*temp == ASCI_P )/* if transaction_input_form */
```

```
                {
                   process_input ( temp );/* process input data */                }
                else
                {
                   display_error ();
                }
             }
          }
       } /* end of if data_error == 'N' ) */
       else /* data error not equal N */
       {
          if ( data_error == 'W' )/* welcome screen */
               /* send the signon screen           */
          {
              process_start (  );
          }
          else
          {
              display_error ();
          }
       }

       if(write(sd2,writePtr,writeLen)<0)
       {
         perror("write() failed");
         break;/* error management for write error */
       }
       if (( form_number == EXIT )|(form_number == NOT_FOUND_ERROR))
          break; /* exit the receive loop */
     } /* end of the recv else statment */
   } /* end of for loop */
   close(sd2);

   /* Free Buffers & Detach from System/T */
   tpterm();
   shmdt(config);
   iconv_close(toIA5);
   iconv_close(toEBCDIC);
   close(sd);
   exit(0);
}
void process_start ( void )
{
       writePtr=config->formbWELCOMEc.addr;
       writeLen=config->formbWELCOMEc.size;
}
void display_error ( void )
{
       form_number = NOT_FOUND_ERROR;
       writePtr=config->formbNOT_FOUND_ERRORc.addr;
       writeLen=config->formbNOT_FOUND_ERRORc.size;
}
void process_normal ( char * temp )
{
   char form_selector;
   char * marker;                /* Location of the formid string    */
   char          input_error='Y'; /* input error flag */
   char          input_error_1='Y';/* scan warehouse input error flag */
   char          input_error_2='Y';/* scan district input error flag  */
   int           i;

    /* Find the text string FORMID= and then get the form number       */
    if((marker=strstr(temp,FORMID))!=NULL)
    {
        /* advance pointer to the end of the formid string */
        temp = marker + strlen(FORMID);
        form_number = *temp - ONE;/* get the form number               then */
    }
    else/* FORMID not found */
    {
       form_number = NOT_FOUND_ERROR;/* set form number to error form */
       i = form_number;/* set form index */
       /* setup error form and return */
       writePtr=config->formbic.addr;
       writeLen=config->formbic.size;
       hiddenWid=config->formbic.hiddenWid;/* set the warehouse field */
       hiddenDid=config->formbic.hiddenDid;/* set the district field  */
       wid=config->formbic.wid;           /* set the warehouse field */
       return;
    }
   /* check the form number */
   if ( form_number == WELCOME )/* welcome form */
   {
       /* check for the warehouse id  text                        */
if((marker=strstr(temp,WAREHOUSEID))!=NULL)
      {
          temp=marker+strlen(WAREHOUSEID)-1;/* go just to the = sign */
          input_error = 'Y';
          /* scan for warehouse ID */
          input_error_1 = scan_strings(&temp,warehouse,5,ampersand);
          {
            if((temp=strchr(temp,parameter))!=NULL)
            /* advance pointer to start of district data */
            {
                       /* scan for district ID */
                input_error_2=scan_strings(&temp,district,3,ampersand);
            }
          }
          if (( input_error_1 == 'N' )&&(input_error_2 == 'N'))
            input_error = 'N';/* set the input error flag */
          if ( input_error == 'N') /* district and warehouse are correct */
          {
              form_number = MENU;/* set up the menu display */
              i = form_number;/* set form index */
              /* setup menu form and return */
              writePtr=config->formbic.addr;
              writeLen=config->formbic.size;
              hiddenWid=config->formbic.hiddenWid;/* set the warehouse field */
              hiddenDid=config->formbic.hiddenDid;/* set the district field  */
          }
          else
          {
              form_number = INPUT_ERROR;
              /* setup input error screen form and return */
              i = form_number;
          } /* end of inner else */
       }
       else /* warehouse id not found */
       {
          form_number = NOT_FOUND_ERROR;/* set form number to error screen */
          i = form_number;
       }
       writePtr=config->formbic.addr;
       writeLen=config->formbic.size;
       hiddenWid=config->formbic.hiddenWid;
       hiddenDid=config->formbic.hiddenDid;
       wid=config->formbic.wid;           /* set the warehouse field */
   } /* end of check for Welcome FORM  */

   else  /* The form is a memu selection       */
   {
     /* check for the TERMID text                       */
     if((marker=strstr(temp,TERMID))!=NULL)
     {
          temp = marker + strlen(TERMID);/* advance the pointer */
          for(
               i=0;
               tempbic&&tempbic!=INPUT_BLANK&&tempbic!=ampersand&&
                 (i<WarehouseFieldSize);
               ++i
                )
          warehousebic=tempbic;
          /* check SYNCID text */
          if((marker=strstr(temp,SYNCID))!=NULL)
          {
              temp = marker + strlen(SYNCID);/* go to the syncid string */
              /* advance pointer to syncid                     */
              for(
                  i=0;
                  tempbic&&tempbic!=INPUT_BLANK&&tempbic!=ampersand&&
                    (i<DistrictFieldSize);
                  ++i
                    )
               districtbic=tempbic;
              /* check for the &CMD=.. text */
              if((marker=strstr(temp,CMDSTRING))!=NULL)
              {
                 /* advance the pointer to the end of the command string   */
                 temp = marker + strlen(CMDSTRING);
                 form_selector = *temp;
                 /* form_selector is the first character of the command    */
              }
              else
              {
               form_selector = ASCI_N;/* default to new order */
              }
              /* check the form number */
              if ( form_selector == ASCI_N )
                   i= NEW_ORDER;/* set up the new order form              */
              else
              {
                 if ( form_selector == ASCI_P )
                 {
                    i= PAYMENT;
                    date_loc=config->formbic.date_location;
                    copy_date();
                 }
                 else
                 {
                    if ( form_selector == ASCI_O )
                       i= ORDER_STATUS;
                    else
                    {
                       if ( form_selector == ASCI_D )
                          i= DELIVERY;
                       else
                       {
                           if ( form_selector == ASCI_S )
                              i= STOCK_LEVEL;
                           else
                           {
                              if ( form_selector == ASCI_E )
                                 i = EXIT;
                              else
                                 i = NEW_ORDER; /* send new order  */
}/* end else if ( form_selector == ASCI_S ) */
                           }     /* end else if ( form_selector == ASCI_D ) */
                       }         /* end else if ( form_selector == ASCI_O ) */
                    }            /* end else if ( form_selector == ASCI_P ) */
                 }               /* end else if ( form_selector == ASCI_N ) */
              form_number = i; /* set the form number */
              writePtr=config->formbic.addr;
              writeLen=config->formbic.size;
              /* Set local ptrs to hidden vars */
              hiddenWid=config->formbic.hiddenWid;
              hiddenDid=config->formbic.hiddenDid;
              wid=config->formbic.wid;   /* set the warehouse field */
            } /* end of if SYNCID found */
            else
            {
              form_number = NOT_FOUND_ERROR; /* set the form number */
              writePtr=config->formbNOT_FOUND_ERRORc.addr;
              writeLen=config->formbNOT_FOUND_ERRORc.size;
            }
          }   /* end of if TERMID found */
          else
          {
              form_number = NOT_FOUND_ERROR; /* set the form number */
              writePtr=config->formbNOT_FOUND_ERRORc.addr;
              writeLen=config->formbNOT_FOUND_ERRORc.size;
          }
      }       /* end of else form id not equal WELCOME form       */}



void copy_date(void)
          /* Poke date into form when required */
{
  int i;/* for loop counter */
  _MI_Time     time_of_day; /* time of day  */
  mattod( time_of_day );/* get the time of day */
  time_of_dayb7c='B';
  getdttmstr ( time_of_day, cl_time, 'm' );/* get time string */
  cl_timebl9c = '\0'; /* terminate the string */
  for(i=0;i<19;++i) {
    switch ( cl_timebic )
    {
             case '-':
                  date_locbic = DASH;
                break;
             case ':':
                  date_locbic = COLON;
                break;
             case ' ':
                  date_locbic = BLANK;
                break;
             default:
                  date_locbic=(cl_timebic-'0')+ZERO;
                break;
          } /* end of switch */
      } /* end of for loop */
}/* end of copy date  */

/*****************************************/
/* checks if the input data is valid     */
/*****************************************/
char parse_input ( char * input, char ** new_ptr )
{
   int index;
   int test_string_size; /* set longer then max string */   char * test_string;/* string to
test                */   char * data_string;/* pointer to new test string    */   char mode =
'N';
   int normal_len;
   int welcome_len;
   normal_len = strlen( TEST_DATA )-1;/* index of length */
   welcome_len = strlen ( WELCOME_DATA )-1;/* inedx of length */
   for(index=0;index<5;index++)
   {
```

```
      if ( inputbindexc != GET_STRINGbindexc )
      {
           *new_ptr = NULL;/* set a null pointer for return */
           return ('Y');
      }
   }
   if ( inputb5c == TEST_DATAb0c )
   {
      test_string = (unsigned char *)&TEST_DATAb0c;
      test_string_size = strlen (TEST_DATA);
      mode = 'N';/* set the processing mode */
   }
   else
   {
       if ( inputb5c == WELCOME_DATAb0c )
       {
          test_string =(unsigned char *)&WELCOME_DATAb0c;
          test_string_size = strlen (WELCOME_DATA);
          mode = 'W';/* set the processing mode */
       }
       else
       {
           *new_ptr = NULL;/* set a null pointer for return */
           return ('Y');
       }
   }
   data_string =(unsigned char *)&inputb5c;   /* get starting point      */
   for(index=1;index<test_string_size;index++)/* index 0 has ben tested  */
   {
      if ( data_stringbindexc != test_stringbindexc )
      {
            *new_ptr = NULL;/* set a null pointer for return */
            return ('Y'); /* error detected */
      }
   }  /* end of the for loop                      */
   *new_ptr = data_string + test_string_size;
   return (mode);
}



/***************************************/
/* checks if the input data is numeric   */
/***************************************/
char scan_strings ( char ** current_ptr, char * output_loc, int size,
                    char terminator )
{
   int blank_count=0;/* blank character counter */
   char * start_ptr;/* pointer to the active string start           */
   char * output_ptr; /* pointer to current string location         */
   int  scan_count;          /* for loop counter                    */
   char scan_error='N';      /* scan error flag                     */
   start_ptr = *current_ptr+1;/* set the starting postion pointer    */
   output_ptr = output_loc;/* saved location                        */
   for( scan_count = 0;scan_count<size; scan_count++)
   {
     *current_ptr=*current_ptr+1;/* next character point       */
     if (((**current_ptr) >= ZERO)&&((**current_ptr)<= NINE))
     {
         if( blank_count == 0 )      /* not after a trailing blank */
         {
                  *output_ptr = **current_ptr; /* save the data              */
                  output_ptr  =  output_ptr+1;/* go to the next location     */
          }
          else
          {
                  scan_error = 'Y';/* this is an input error */
                  scan_count = size;/* you are done scanning */
           }
     }
     else /* end of if numeric */
     {
       if ( (**current_ptr) == terminator )
       {
          scan_count = size;/* you are done scanning */
          if ((*current_ptr)==start_ptr)/* no character were processed       */
          {
              scan_error = 'Y';
          }
       }
       else/* character is not a terminator */
       {
          if ( (**current_ptr) == INPUT_BLANK )/* check for blanks */
          {
             if ((*current_ptr)==start_ptr)    /* leading blank             */
             {
                start_ptr = start_ptr+1;       /* skip this character       */
             }
             else
             {
                blank_count=blank_count+1;   /* count the traling input blank */
             }
          }
          else
          {
              scan_error = 'Y';/* this is an input error */            scan_count = size;/*
you are done scanning */
          }
       }
      }/* end of else not numeric */
    }/* end of the for loop */
    if ((*current_ptr)==start_ptr)   /* blank input   */
    {
          scan_error = 'Y';
    }
    return scan_error;
}


/***********************************************/
/* Initialize the environment including tuxedo */
/***********************************************/
void initTux        ( void )
{
  toIA5=iconv_open(ToIA5,FromEBCDIC);
  if(toIA5.return_value<0){
    perror("\niconv_open() failed");
    exit(1);
  }

  toEBCDIC=iconv_open(ToEBCDIC,FromIA5);
  if(toEBCDIC.return_value<0){
    perror("\niconv_open() failed");
    exit(1);
  }

  /* set up the tuxedo environment   */
  putenv("TUXDIR=/qopensys/tuxsdk");
  putenv("APPDIR=/home/tpcckvclnt");
  putenv("TUXCONFIG=/home/tpcckvclnt/tuxconfig");


       /* Attach to System/T as a Client Process */
       if (tpinit((TPINIT *) NULL) == -1) {
                (void) fprintf(stderr, "Tpinit failed\n");
                exit(1);
       }

  /* allocate space for buffers       */

  if((nd.new_inp=(new_order_input *)tpalloc("CARRAY", NULL, 230))==NULL) {
                    (void) fprintf(stderr, "Can't allocate buffer for newordi\n" );
                    exit(1);
                }
  if((pd.pay_inp= (payment_input *) tpalloc("CARRAY", NULL, 230))==NULL) {
                    (void) fprintf(stderr, "Can't allocate buffer for paymnti\n" );
                    exit(1);
                }
  if((di= (dlv_input *) tpalloc("CARRAY", NULL, 230))==NULL) {
                    (void) fprintf(stderr, "Can't allocate buffer for delvryi\n" );
                    exit(1);
                }
 if((od.ordsts_inp=(order_status_input *)tpalloc("CARRAY", NULL, 230))==NULL) {
                    (void) fprintf(stderr, "Can't allocate buffer for ordstsi\n" );
                    exit(1);
                }
  if((s_d.stk_inp= (stock_input *) tpalloc("CARRAY", NULL, 230))==NULL) {
                    (void) fprintf(stderr, "Can't allocate buffer for stklvli\n" );
                    exit(1);
                }


  if((nd.new_out= (new_order_output *) tpalloc("CARRAY", NULL, 670))==NULL) {
                    (void) fprintf(stderr, "Can't allocate buffer for newordo\n" );
                    exit(1);
                }
  if((pd.pay_out= (payment_output *) tpalloc("CARRAY", NULL, 670))==NULL) {
                    (void) fprintf(stderr, "Can't allocate buffer for paymnto\n" );
                    exit(1);
                }
  if((od.ordsts_out=(order_status_output *)tpalloc("CARRAY",NULL,670))==NULL) {
                    (void) fprintf(stderr, "Can't allocate buffer for ordstso\n" );
                    exit(1);
                }
  if((s_d.stk_out= (stock_output *) tpalloc("CARRAY", NULL, 670))==NULL) {
                    (void) fprintf(stderr, "Can't allocate buffer for stklvlo\n" );
                    exit(1);
                }
}



/*********************************************/
/* Processes the transaction input data      */
/*********************************************/

void process_input  ( char * input_buffer )
{
  int             formId;/* form ID from the input */
  int             formIdPost;/* modified form input */
  int           i, j;
  int           contentSize;
  unsigned char *fromPtr, *toPtr;
  Config_t      *config;
  int           count;
  int           counter;/* for loop counter */
  char          *work;/* buffer to get the warehouse number */
  unsigned int  length, inSize, outSize;
  char          argument[BufferSize];/* location after converting */
  char          *temp, *inStr;
  long          send_len = 670, *recv_len;
  int           ret;
  _MI_Time      time_of_day;    /* time of day */


  recv_len = &send_len;
  /* Find the form id number and save as the FormId */
  if ((work=strstr(input_buffer, FORMID))==NULL)
  {
        printf("\nMissing FORMID");
        formId = NUM_FORMS;
  }
  else
  {
     work = work + sizeof(FORMID) -1;
     formId = *work - ONE;/* get the form id number */
  }
  /* Find start of argument area */
  if((inStr=strstr(input_buffer,TERMID))==NULL)
     printf("\nMissing TERMID");
  else if((temp=strchr(inStr,BLANK))==NULL)/* locate next blank */
     printf("\nMissing Blank");
  else {
      inSize=outSize=(temp-inStr);
      temp=argument;
      temp[inSize]=0;        /* Insert string terminator */

      /* Convert arguments to EBCDIC */
          if(iconv(toEBCDIC,&inStr,&inSize,&temp,&outSize)<0){
            perror("\niconv() failed");
            exit(1);
          }
      formIdPost = formId;
      work = argument + sizeof(termid) -1;
  memset ( warehouse, '\0', WarehouseFieldSize + 1 );
  for ( counter=0;(counter<WarehouseFieldSize)&&
        (workbcounterc!='&')&&
        (workbcounterc!='+');
        counter++)
     warehousebcounterc = workb counter c;
  if ((work = strstr( work, "SYNCID=" ))==NULL )
  {
     printf ( "District not found\n" );
  }
  else
  {
     memset ( district, '\0', DistrictFieldSize + 1 );
     work = work + 7;
   for ( counter=0;(counter<DistrictFieldSize)&&
        (workbcounterc!='&')&&
        (workbcounterc!='+');
        counter++)
     districtbcounterc = workb counter c;
  }

  /* process transaction based on type    */
  switch(formId) {
    case NEW_ORDER:

   count = scan_new_order( argument, nd.new_inp);
  if ( count == 0 )/* new order data valid */
  {
                 ret=tpcall("NEWORDER",(char *)nd.new_inp,send_len,
                (char **) &nd.new_out, (long *) recv_len, TPNOTIME);

                 if(ret == -1) {
                     (void) fprintf(stderr, "Can't send request to NEWORDER \n" );
                     (void) fprintf(stderr, "Tperrno = %d, ret=%d\n", tperrno,ret);
```

```
                    tpterm();
                    exit(1);
                }
            }
else    /*  invalid new order data    */
            formIdPost = set_error_report( count );

               /* Respond to the client */
                    postResult(&nd, warehouse, district, formIdPost, scratchBuffer);

   break;

   case PAYMENT:
         count = scan_payment ( argument, pd.pay_inp );
         if(count == 0) /* payment data valid */
         {

               ret=tpcall("PAYMNT",(char *)pd.pay_inp,send_len,
           (char **) &pd.pay_out, (long *) recv_len, TPNOTIME);

  if(ret == -1) {
    (void) fprintf(stderr, "Can't send request to service PAYMENT\n");
    (void) fprintf(stderr, "Tperrno = %d\n", tperrno);
    tpterm();
    exit(1);
  }
  }        /* payment data invalid */
   else       formIdPost = set_error_report( count );
         /* Respond to the client */
                    postResult(&pd, warehouse, district, formIdPost, scratchBuffer);

     break;

   case ORDER_STATUS:
         count=scan_order_status ( argument, od.ordsts_inp );
         if (count == 0) {  /* order status data valid */

               ret=tpcall("ORDSTS",(char *)od.ordsts_inp,send_len,
           (char **) &od.ordsts_out, (long *) recv_len, TPNOTIME);


  if(ret == -1) {
    (void) fprintf(stderr, "Can't send request to service ORDSTS\n");
    (void) fprintf(stderr, "Tperrno = %d\n", tperrno);
    tpterm();
    exit(1);
  }
    }          /* order status data invalid */
      else       formIdPost = set_error_report( count );
                    postResult(&od, warehouse, district, formIdPost, scratchBuffer);

      break;

   case DELIVERY:
         count= scan_delivery( argument, di );
         if (count == 0) {   /* delivery data valid */
            /* do a get time here */
            mattod ( time_of_day );
            time_of_day[7]= 'B'; /* set ending flag */
            getdttmstr (time_of_day, di->time, 'h');

               ret=tpcall("DELIVRY",(char *)di,send_len,
           (char **) &di, (long *) recv_len, TPNOTIME);

  if(ret == -1) {
    (void) fprintf(stderr, "Can't send request to service DELIVERY\n");
    (void) fprintf(stderr, "Tperrno = %d\n", tperrno);
    tpterm();
    exit(1);
  }
         }            /* delivery data invalid */
         else       formIdPost = set_error_report( count );
                    postResult(di, warehouse, district, formIdPost, scratchBuffer);
     break;

   case STOCK_LEVEL:
         count= scan_stock_level ( argument, s_d.stk_inp );
         if (count == 0) {  /* stock level data valid */

               ret=tpcall("STKLVL",(char *)s_d.stk_inp, send_len,
           (char **) &s_d.stk_out, (long *) recv_len, TPNOTIME);


  if(ret == -1) {
    (void) fprintf(stderr, "Can't send request to service STKLVL\n");
    (void) fprintf(stderr, "Tperrno = %d\n", tperrno);
    tpterm();
    exit(1);
  }
}
        /* stock level  data invalid */
       else       formIdPost = set_error_report( count );
                    postResult(&s_d, warehouse, district, formIdPost, scratchBuffer);
     break;

   case NUM_FORMS:    /* invalid form number  */
      formIdPost = SIGNON_ERROR;
                    postResult(&nd, warehouse, district, formIdPost, scratchBuffer);
     break;

    default:      /* return an error if no match above */      printf("Invalid form ID = %d",
formId);
      return;

  }/* end of switch */
  inSize = outSize = contentSize = strlen(scratchBuffer);

  /* Copy header into output buffer */
  sprintf(buffer,"%s",HEADER);
  length = strlen(buffer);

  /* Convert EBCDIC result to ASCII */
  toPtr = buffer + length;
  length += inSize;
  fromPtr = scratchBuffer;

  if(iconv(toIA5,&fromPtr,&inSize,&toPtr,&outSize)<0){
    perror("\niconv() failed");
    exit(1);
  }

  /* Poke in response length */
  binToIA5(contentSize,strchr(buffer,delimiter));
  writePtr = buffer;/* set up the output function */
  writeLen = length;/* set the output length  */
  }
}

int set_error_report( int status )
{
  if ( status == 100 )
  {
      return SIGNON_ERROR;
  }
  else
  {
      return DATA_ERROR;
  }
}
```

GETDTTMSTR.C:

```
/*****************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <stddef.h>
#include <math.h>
#include <ctype.h>
#include <decimal.h>
#include <recio.h>
#include <errno.h>
#include <time.h>
#include <xxfdbk.h>
#include <xxcvt.h>
#include <micomput.h>
#include <QSYSINC/MIH/MIDTTM>
#include <qsysinc/mih/cvtch>
/*****************************************************************/
/* Function Prototypes                                           */
/*****************************************************************/
void MI_Template (_INST_Template_T1 **, _DDAT_T **, _DDAT_T **, char );

/*****************************************************************/
/* Date and time coversion pointers                              */
/*****************************************************************/
 _INST_Template_T1 *inst_t1;          /* MI template         */
 _DDAT_T   *ddat_t1 , *ddat_t2 ;      /* DDAT pointers             */
 /* Internal format for end of Gregorian timeline: Jan. 1,  0001 */ #define
GREGORIAN_TIMELINE_START 1721424
 /* Internal format for end of Gregorian timeline: Jan. 1, 10000 */ #define
GREGORIAN_TIMELINE_END   5373485
/*****************************************************************/
/*        Start                                                  */
/*****************************************************************/
getdttmstr(char * timein, char* timestamp, char type)
{

 char tmstampb27c;
 static char first = 'Y';

 if (( timeinb7c == ' ' )||(timeinb7c=='0'))
 {
 switch ( type )
 {
    case 'd':
       memset ( timestamp, '0', 8 );/* set the null date */
    break;
    case 't':
       memset ( timestamp, '0', 6 );/* set the null date */
    break;
    case 'h':
       memset ( timestamp, '0', 8 );/* set the null date */
    break;
    case 'b':
       memset ( timestamp, '0', 14 );/* set the null date */
    break;
    case 'l':
       memcpy( &timestampb0c,/* copy hour min sec */
       "0000-00-00.00.00.00.000000\0", 27 );
    break;
    case 'm':
       memcpy( &timestampb0c,/* copy hour min sec */
       "0000-00-00 00:00:00.000000\0", 27 );/* set null string */
    break;
    case 'n':
       memcpy( &timestampb0c,/* copy hour min sec */
       "0000/00/00 00:00:00.000000\0", 27 );/* set null string */
    break;
    default:
    break;
  } /* end of switch */
 }
 else
 {
 MI_Template(&inst_t1,&ddat_t1,&ddat_t2, 'P');
 timeinb7c = 0;/* clear the era flag */
 cvtts(tmstamp, timein, inst_t1);

 switch ( type )
 {
    case 'd':
       memcpy( &timestampb4c, &tmstampb0c, 4 ); /* copy the year */
       memcpy( &timestampb0c, &tmstampb5c, 2 ); /* copy month    */
       memcpy( &timestampb2c, &tmstampb8c, 2 ); /* copy day     */
    break;
    case 't':
    case 'h':
       memcpy( &timestampb0c, &tmstampb11c, 2 ); /* copy hour     */
       memcpy( &timestampb2c, &tmstampb14c, 2 );/* copy minuites */
       memcpy( &timestampb4c, &tmstampb17c, 2 ); /* copy secound  */
       if ( type == 'h')
       memcpy( &timestampb6c, &tmstampb20c, 2 ); /* copy secound  */
    break;
    case 'b':
       memcpy( &timestampb4c, &tmstampb0c, 4 ); /* copy the year */
       memcpy( &timestampb0c, &tmstampb5c, 2 ); /* copy month    */
       memcpy( &timestampb2c, &tmstampb8c, 2 ); /* copy days     */
       memcpy( &timestampb8c, &tmstampb11c, 2 ); /* copy hour     */
       memcpy( &timestampb10c, &tmstampb14c, 2 );/* copy minuites */
       memcpy( &timestampb12c, &tmstampb17c, 2 ); /* copy secound  */
    break;
    case 'l':
       memcpy( &timestampb0c, &tmstampb0c, 27 );/* copy hour min sec */
    break;
    case 'm':
    case 'n':
       memcpy(&timestampb0c,&tmstampb5c,2);       /* get month    */
       memcpy(&timestampb3c,&tmstampb8c,2);       /* get day      */
       memcpy(&timestampb6c,&tmstampb0c,4);       /* get year     */
       memcpy(&timestampb11c,&tmstampb11c,2);     /* get hours    */
       memcpy(&timestampb14c,&tmstampb14c,2);     /* get minutes  */
       memcpy(&timestampb17c,&tmstampb17c,2);     /* get seconds  */
       timestampb10c = ' ';/* set the time seperator */
       timestampb13c = ':';/* set the time seperator */
       timestampb16c = ':';/* set the time seperator */
       if ( type == 'n' )
       {
         timestampb2c = '/';/* set the date seperator */
         timestampb5c = '/';/* set the date seperator */
       }
       else
```

```
      {
        timestampb2c = '-';/* set the date seperator */
        timestampb5c = '-';/* set the date seperator */
      }
    break;
    default:
    break;
 } /* end of switch */
 } /* end of else   */
}
void makemitime ( char * date, char *time, char * flag,
                  char * output )
{
  char workingb27c;/* the working 26 character data input */
  char cvt_type = 'I';
  char impi_timeb9c = {0x7D,0x66,0x27,0x9F,0xE4,0xF5,0x00,0x00};
 char tmstamplb27c = "1998-07-24-13.16.55.000000";

  /* format   YYYY-MM-DDHH.MI.SS.999999   */
 /* location  01234567890123456789012345  */
 /* tens      0         1         2       */
  memcpy ( &workingb0c, &dateb4c, 4); /* get the year */
  memcpy ( &workingb5c, &dateb0c, 2); /* get the month */
  memcpy ( &workingb8c, &dateb2c, 2); /* get the day   */
  memcpy ( &workingb11c, &timeb0c, 2); /* get the hour  */
  memcpy ( &workingb14c, &timeb2c, 2); /* get the minite */
  memcpy ( &workingb17c, &timeb4c, 2); /* get the secound */
  workingb4c = '-'; /* set the dashes */
  workingb7c = '-'; /* set the dashes */
  workingb10c = '-'; /* set the dashes */
  workingb13c = '.'; /* set the time seperator */
  workingb16c = '.';/* set the porid */
  workingb19c = '.';/* set the porid */
  workingb26c = '\0';/* terminate the string */
  if ( flagb0c == 'H' )
  {
        memset ( &workingb20c, '9', 6 );/* set the value to the maximume */
  }
  else
  {
        memset ( &workingb20c, '0', 6 );/* set the value to the minimume */
  }
    MI_Template(&inst_t1,&ddat_t1,&ddat_t2, cvt_type);
    /* printf ( "The time  stamp was %s\n", tmstampl ); */
       printf ( "The time is         %s\n", working );
    cvtts( output, working, inst_t1);
}
 /******************** MI_Template ***********************************/
 /* Function: MI_Template                                            */
 /*                                                                  */
 /* Description: Allocate and fill in MI instruction template used   */
 /*              in date and time conversions.                       */
 /*                                                                  */
 /********************************************************************/
void  MI_Template(_INST_Template_T1 **inst_t, _DDAT_T **ddat1,
             _DDAT_T **ddat2, char cvt_type)
{
 static char first = 'Y';
 /********************************************************************/
 /* Date and Time conversion (mi template)                           */
 /********************************************************************/
  _Era_Table_T     *era_t1,  *era_t2; /* Era table pointers          */
  _Calendar_Table_T  *cal_t1,  *cal_t2; /* Calendar table pointers     */

 int               DDAT_Length, Calendar_Offset;
 int               DDAT_Size, Template_Size;
 int               DDAT_Offset1, DDAT_Offset2;
 /********************************************************************/
 if (first == 'Y')             /* First call ?                 */
 {                             /* Yes, allocate template etc.  */    first ='N';

 DDAT_Length = sizeof(_DDAT_T)+2*(sizeof(_Calendar_Table_T))
              -sizeof(short) + sizeof(_Era_Table_T)-1;

 Calendar_Offset = offsetof(_DDAT_T, Tables)+sizeof(_Era_Table_T);

 DDAT_Size = 2*DDAT_Length +
             2*(sizeof(int)) + /* DDAT_Offset  */
             10 +              /* reserved4    */
             sizeof(short) +   /* Num_DDATS    */
             sizeof(int);      /* DDAT_Size    */
                               /* DDAT offset  */

 Template_Size = 2*DDAT_Length+offsetof(_INST_Template_T1,DDAT)
                 + sizeof(int);
                               /* DDAT1 offset */
 DDAT_Offset1  = (offsetof(_INST_Template_T1,DDAT) -
                 offsetof(_INST_Template_T1,DDAT_Size)) + sizeof(int);
                               /* DDAT2 offset */

 DDAT_Offset2  = (offsetof(_INST_Template_T1,DDAT) -
                 offsetof(_INST_Template_T1,DDAT_Size))+sizeof(int)+DDAT_Length;

    *inst_t        = (_INST_Template_T1 *)malloc(Template_Size);

 /* Fill in Instruction Template */


 memset(*inst_t,'\0', sizeof(_INST_Template_T1));
 (* inst_t)->Template_Size= Template_Size; /* Instruction template size */
 (* inst_t)->DDAT_1       = 1;             /* Operand 1 DDAT        */
 (* inst_t)->DDAT_2       = 2;             /* Operand 2 DDAT        */
 (* inst_t)->Length_1     = 26;            /* Result timestamp      */
 (* inst_t)->Length_2     = 8;             /* input mitime          */
 (* inst_t)->DDAT_Size       = DDAT_Size;  /* Size of DDAT list     */
 (* inst_t)->Num_DDATs       = 2;          /* Number of DDATs       */
 (* inst_t)->DDAT_Offsetb0c  = DDAT_Offset1; /* DDAT offset1       */
 *((* inst_t)->DDAT_Offset+1)= DDAT_Offset2; /* DDAT offset2       */

 *ddat1   = (_DDAT_T *)&((* inst_t)->DDAT_Offset+2);
 era_t1 = (_Era_Table_T *)&((* ddat1)->Tables); /* set era table ptr */
 cal_t1    = (_Calendar_Table_T *)((char *)*ddat1 + Calendar_Offset);
 *ddat2   = (_DDAT_T *)((char *)*ddat1 + DDAT_Length);
 era_t2    = (_Era_Table_T *)&((* ddat2)->Tables);
 cal_t2    = (_Calendar_Table_T *)((char *)*ddat2 + Calendar_Offset);

 /* Fill in DDAT1 */

 (* ddat1)->DDAT_Length = DDAT_Length;      /* Set DDAT length       */
 (* ddat1)->Format_Code =_SAA_TIMESTAMP; /* Set result format code     */
 (* ddat1)->Hour_Zone = 24;            /* set target hours           */
 (* ddat1)->Min_Zone  = 60;            /* set target minutes         */
 (* ddat1)->Calendar_Offset=Calendar_Offset;/* Calendar offset       */
 memset ((* ddat1)->reserved, 0, 6 );  /* set the reserved values    */
 /* Fill in Era Table for DDAT1 */

    era_t1->Num_Elems = 1;               /* Set number of era elements  */
    era_t1->Elementb0c.Origin_Date = GREGORIAN_TIMELINE_START;
    era_t1->Elementb0c.Era_Nameb0c= 'A'; /* Set era name, to ....       */
    era_t1->Elementb0c.Era_Nameblc= 'D'; /* AD                          */
    memset (era_t1->Elementb0c.reserved, 0, 12 );/* set the reserved v alues */


 /* Fill in Calendar Table for DDAT1 */

    cal_t1->Num_Elems = 2;               /* Set Calendar elements       */
    cal_t1->Element->Effect_Date     = GREGORIAN_TIMELINE_START;
    cal_t1->Element->Type            = 0x0001; /* Set element type      */
    memset(cal_t1->Element->reserved,'\0',10); /* Initialize reserved   */
    (cal_t1->Element+1)->Effect_Date = GREGORIAN_TIMELINE_END;
    (cal_t1->Element+1)->Type = 0;
    memset((cal_t1->Element+1)->reserved,'\0',10);

 /* Fill in DDAT2 */

 (* ddat2)->DDAT_Length = DDAT_Length;     /* Set DDAT length        */
 (* ddat2)->Format_Code =_IMPI_CLOCK;  /* Set source format code     */
 (* ddat2)->Hour_Zone = 24;           /* set target hours            */
 (* ddat2)->Min_Zone  = 60;           /* set target minutes          */
 (* ddat2)->Calendar_Offset=Calendar_Offset;/* Calendar offset       */
 (* ddat2)->Calendar_Offset = Calendar_Offset;
 memset ((* ddat2)->reserved, 0, 6 );/* set the reserved values   */

 /* Fill in Era Table for DDAT2 */

    era_t2->Num_Elems = 1;
    era_t2->Elementb0c.Origin_Date   = GREGORIAN_TIMELINE_START;
    era_t2->Elementb0c.Era_Nameb0c   = 'A';
    era_t2->Elementb0c.Era_Nameblc   = 'D';
    memset (era_t2->Elementb0c.reserved, 0, 12 );/* set the reserved v alues */

 /* Fill in Calendar Table for DDAT2 */

    cal_t2->Num_Elems                = 2;
    cal_t2->Element->Effect_Date     = GREGORIAN_TIMELINE_START;
    cal_t2->Element->Type            = 0x0001;
    memset(cal_t2->Element->reserved,'\0',10);
    (cal_t2->Element+1)->Effect_Date = GREGORIAN_TIMELINE_END;
    (cal_t2->Element+1)->Type        = 0;
    memset((cal_t2->Element+1)->reserved,'\0',10);
 }/* end of if (first) */
 if (cvt_type == 'I')                        /* converting to mitime ?  */
 {                                           /* Yes, then ....          */
    (* ddat1)->Format_Code =_IMPI_CLOCK;     /* Set source format code  */
    (* ddat2)->Format_Code =_SAA_TIMESTAMP;  /* Set source format code  */
    (* inst_t)->Length_1     = 8;            /* Result timestamp        */
    (* inst_t)->Length_2     = 26;           /* input mitime            */
 }
 else
 {
    (* ddat1)->Format_Code =_SAA_TIMESTAMP;  /* Set source format code  */
    (* ddat2)->Format_Code =_IMPI_CLOCK;     /* Set source format code  */
    (* inst_t)->Length_1     = 26;           /* Result timestamp        */
    (* inst_t)->Length_2     = 8;            /* input mitime            */
 }
}/* end of proceedure */
```

IMPORT.H:

```
/*********************************************************************/
/* Defines for parameters passed to DBLOAD                           */
/*********************************************************************/
#define FROMFILE      1
#define TOFILE        2
#define TOFILENAME    3
#define TOLIB         4
#define FROMRCD       5
#define TORCD         6
#define MTASK         7
#define TASKNBR       8
#define NBRTASKS      9
#define FTYPE         10
#define STR_DELIMIT   11
#define DELIMIT       12
#define ROW           13
#define DATFMT        14
#define DATSEP        15
#define TIMFMT        16
#define TIMSEP        17
#define DECMPT        18
/*********************************************************************/
/* Miscellaneous Defines                                             */
/*********************************************************************/
#define FALSE      0
#define TRUE       !FALSE
#define AS400   /* AS400    */
/************************/
/* MoveField type codes  */
/************************/
#define CHAR_GRAPH     0x00
#define VAR_CHAR_GRAPH 0x01
#define INTEGER        0x02
#define SMALLINT       0x03
#define FLOAT          0x04
#define PACKED         0x05
#define ZONED          0x06
#define DATE           0x07
#define TIME           0x08
#define TIMESTAMP      0x09
#define NUMARIC_CHAR   0x0A
/************************/
/* Date and Time defines */
/************************/
#define USA            0x00
#define ISO            0x01
#define EUR            0x02
#define JIS            0x03
#define MDY            0x04
#define DMY            0x05
#define YMD            0x06
#define JUL            0x07
#define HMS            0x08
#define SAA            0x09
/*********************************************************************/
/* ERRORS                                                            */
/*********************************************************************/
#define  OPEN_IMPORT         1
#define  OPEN_DBFILE         2
#define  COL_DEL_NOT_FOUND   3
#define  COLUMNS_ERROR       4
#define  FIELD_LENGTH        5
#define  STRING_DELIMIT      6
#define  NULL_ERROR          7
#define  MEM_ERROR           8
#define  MISSING_STR_DELIMIT 9
#define  STR_RECORD         10
#define  END_RECORD         11
#define  OPEN_HDRFILE       12
#define  TSKS_ERROR         13
#define  NUMERIC_ERROR      14
#define  DECIMAL_ERROR      15
#define  WRITE_ERROR        16
/*********************************************************************/
```

```
/* Common structure                                                  */
/*********************************************************************/
 typedef struct
 {
 char * colptr;                  /* Data ptr - import file records   */
 char  * inptr;                  /* Field ptr - import file records  */
 char  * outptr;                 /* Output buffer pointer            */
 char  * outstart;               /* Pointer to start of output buffer*/
 char  * decptr;                 /* Decimal point pointer            */
 char  * transptr;               /* Pointer to transaction input     */
 int   ix;                       /* Loop index                       */
 decimal(3,0) new_count;         /* New order line numbers           */
 short buflen;                   /* Import input buffer length       */
 short columnCnt;                /* Number of output columns allowed */
 short fldcnt;                   /* Current number of fields found   */
 short fldlen;                   /* Length of current input field    */
 short Blankcnt;                 /* Trailing blank count (numeric)   */
 char col_delimit;               /* Column delimiter                 */
 char row_delimit;               /* Row delimiter                    */
 char decmpt;                    /* decimal point                    */

 struct
 {
 unsigned Data  : 1;             /* Numeric field data flag          */
 } Numeric;

 char header;                    /* Looking at header data           */
 char end_trans;                 /* processing end of new_order      */
 char tran_type;                 /* transaction type flag            */
 char last_name;                 /* set when last anme was detected  */
 char cust_number;               /* set when a customer number used  */
 char tran_error;                /* transaction error flag           */
 char sign_on_error;             /* sign on sreen error              */
 } ImportParms;

/*********************************************************************/
/* Output record column descriptions                                 */
/*********************************************************************/
 typedef struct
 {
  unsigned char  fldtype;        /* Field data type                  */
  unsigned char  MoveField_code; /* Branch code for MoveField rtn    */
  int            fldlen;         /* Field length                     */
  short          precision;      /* Precision for decimal fields     */
  short          scale;          /* Scale for decimal fields         */
  short          left;           /* Data length left of decimalpoint */
  short          offset;         /* Offset to this field from start  */
                                 /* offset from ip->outstart         */
  unsigned char  number_chk;     /* Validity check flag              */
  unsigned char  skip;           /* skip saving this item            */
 } column_info;
column_info *ci;                 /* Current column_info ptr          */
column_info *ci_str;             /* Pointer to start of column_info  */
/*********************************************************************/
/* Function Prototypes          ***                                  */
/*********************************************************************/
void ImportFields(ImportParms * );
void ColRowDelimiter(ImportParms *);
void MoveField(ImportParms *);
void do_new_order(ImportParms * ip);
void do_payment(ImportParms * ip);
void do_order_status(ImportParms * ip);
void check_input_string(ImportParms * ip);/* check the input string   */
```

NEWORDSRV.C:

```
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <atmi.h>       /* TUXEDO Header File */
#include <string.h>
#include <recio.h>
#include <xxfdbk.h>
#include <userlog.h>    /* TUXEDO Header File */
#include <decimal.h>
#include <quscrtus.h>
#include <qusptrus.h>
#include <qusgen.h>
#include <qp0wpid.h>
#include "common.h"

/* tpsvrinit is executed when a server is booted, before it begins
   processing requests.  It is not necessary to have this function.
   Also available is tpsvrdone (not used in this example), which is
   called at server shutdown time.
*/


   static int sd, addr=0, port=4000;
   new_order_input ni;


tpsvrinit(int argc, char *argv[])
{
       /* Some compilers warn if argc and argv aren't used. */
       argc = argc;
       argv = argv;

       /* userlog writes to the central TUXEDO message log */
       userlog("Welcome to the tpcc server");
    addr=inet_addr("9.5.149.167");
    sd=Connect(addr,port);
       return(0);
}

/* This function performs the actual service requested by the client.
   Its argument is a structure containing among other things a pointer
   to the data buffer, and the length of the data buffer.
*/


     /*       NEW ORDER      */

NEWORDER(TPSVCINFO *rqst)
{

    int                      rc;

    memcpy(&ni, rqst->data, 230);
    if(send(sd, (char *) &ni,230,0)<0) {
      perror("\nsend() failed");
      tpreturn(TPFAIL, 0, rqst->data, 0L, 0);
    }


    rc=recv(sd, rqst->data, 670,0);
    if(rc<0) {
      perror("\nrecv[0]() failed");
      tpreturn(TPFAIL, 0, rqst->data, 0L, 0);
    }

    /* Return the transformed buffer to the requestor. */
    tpreturn(TPSUCCESS, 0, rqst->data, rc, 0);
}
```

ORDSTSSRV.C:

```
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <atmi.h>       /* TUXEDO Header File */
#include <string.h>
#include <recio.h>
#include <xxfdbk.h>
#include <userlog.h>    /* TUXEDO Header File */
#include <decimal.h>
#include <quscrtus.h>
#include <qusptrus.h>
#include <qusgen.h>
#include <qp0wpid.h>
#include "common.h"

/* tpsvrinit is executed when a server is booted, before it begins
   processing requests.  It is not necessary to have this function.
   Also available is tpsvrdone (not used in this example), which is
   called at server shutdown time.
*/


   static int sd, addr=0, port=6000;
   int rc;


tpsvrinit(int argc, char *argv[])
{
       /* Some compilers warn if argc and argv aren't used. */
       argc = argc;
       argv = argv;

       /* userlog writes to the central TUXEDO message log */
       userlog("Welcome to the tpcc server");
    addr=inet_addr("9.5.149.167");
    sd=Connect(addr,port);
       return(0);
}

/* This function performs the actual service requested by the client.
   Its argument is a structure containing among other things a pointer
   to the data buffer, and the length of the data buffer.
*/


     /*       ORDER STATUS   */

ORDSTS(TPSVCINFO *rqst)
{


    if(send(sd, rqst->data,230,0)<0) {
      perror("\nsend() failed");
      tpreturn(TPFAIL, 0, rqst->data, 0L, 0);
    }

    if((rc=recv(sd, rqst->data, 670,0))<0) {
      perror("\nrecv[0]() failed");
      tpreturn(TPFAIL, 0, rqst->data, 0L, 0);
    }

    /* Return the transformed buffer to the requestor. */
    tpreturn(TPSUCCESS, 0, rqst->data, rc, 0);
}
```

PAYMNTSRV.C:

```
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <atmi.h>       /* TUXEDO Header File */
#include <string.h>
#include <recio.h>
#include <xxfdbk.h>
#include <userlog.h>    /* TUXEDO Header File */
#include <decimal.h>
#include <quscrtus.h>
#include <qusptrus.h>
#include <qusgen.h>
#include <qp0wpid.h>
#include "common.h"

/* tpsvrinit is executed when a server is booted, before it begins
   processing requests.  It is not necessary to have this function.
   Also available is tpsvrdone (not used in this example), which is
   called at server shutdown time.
*/


   static int sd, addr=0, port=5000;
   int rc;


tpsvrinit(int argc, char *argv[])
{
       /* Some compilers warn if argc and argv aren't used. */
       argc = argc;
       argv = argv;

       /* userlog writes to the central TUXEDO message log */
    userlog("Welcome to the tpcc server");
    addr=inet_addr("9.5.149.167");
    sd=Connect(addr,port);
    return(0);
}
```

```
/* This function performs the actual service requested by the client.
   Its argument is a structure containing among other things a pointer
   to the data buffer, and the length of the data buffer.
*/



     /*       PAYMENT       */

PAYMNT(TPSVCINFO *rqst)
{


    if(send(sd, rqst->data,230,0)<0) {
      perror("\nsend() failed");
      tpreturn(TPFAIL, 0, rqst->data, 0L, 0);
    }


     if((rc=recv(sd, rqst->data, 670,0))<0) {
      perror("\nrecv[0]() failed");
      tpreturn(TPFAIL, 0, rqst->data, 0L, 0);
    }

    /* Return the transformed buffer to the requestor. */
    tpreturn(TPSUCCESS, 0, rqst->data, rc, 0);
}
```

POSTRESULT.C:

```
#include "Common.h"

#define DATA_ERROR 30
#define SIGNON_ERROR 31
/*********************************************************************/
/*                                                                   */
/*       External functions                                          */
/*                                                                   */
/*********************************************************************/
int getdttmstr(char * timein, char* timestamp, char type);

/*=================================================================*//*  Internal procedures
                                  *//*
         *//*=================================================================*/


char *newOrderResult = {
  "<html>"
    "<head>"
    "<title>TPC-C New Order</title>"
    "</head>"
    "<body>"
    "<FORM ACTION=tpcc.dll METHOD=\"GET\">"
  "<INPUT TYPE=\"hidden\" NAME=\"FORMID\" VALUE=\"3\">"
  "<INPUT TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"%s\">"
  "<INPUT TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"%s\">"
    "<center><h1> TPC-C New Order</h1></center>"
    "<pre>"
    "<br>Warehouse: %4.4d District: %2.2d"
    "                    "
    "Date: %s<br>"
    "Customer:  %4.4d  Name: %15s%c  Credit: %c%c    Disc: %5.2D(5,2)<br>"
    "Order Number: %8d  Number of Lines: %2D(3,0)       W_tax: %4.2D(5,2)"
    "  D_tax:  %4.2D(5,2)<br><br>"
    "<p>"
    "<table>"
    "<tr>"
    "<th>Supp_w  </th>"
    "<th>Item_Id  </th>"
    "<th Colspan=75>Item Name                </th>"
    "                  "
    "<th>             </th>"
    "<th>Qty  </th>"
    "<th>Stock  </th>"
    "<th>B/G  </th>"
    "<th Colspan=35>Price    </th>"
    "<th>Amount</th>"
    "</tr>"
    "</table>"
};

char *newOrderError = {
  "<html>"
    "<head>"
    "<title>TPC-C New Order Error</title>"
    "</head>"
    "<body>"
    "<FORM ACTION=tpcc.dll METHOD=\"GET\">"
  "<INPUT TYPE=\"hidden\" NAME=\"FORMID\" VALUE=\"3\">"
  "<INPUT TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"%s\">"
  "<INPUT TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"%s\">"
    "<center><h1> TPC-C New Order Error</h1></center>"
    "<pre>"
    "<br>Warehouse: %4.4d District: %2.2d"
    "                    "
    "Date: %s<br>"
    "Customer:  %4.4d  <br>"
    "<p>"
    "<table>"
    "<tr>"
    "<th>Supp_w  </th>"
    "<th>Item_Id  </th>"
    "<th>             </th>"
    "<th>Qty  </th>"
    "</tr>"
    "</table>"
};

char *newOrderTail = {
  "Execution Status: %24s            Total:  $%8.2D(11,2)<br>"
  "<br><br>"
  "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..NewOrder..\">"
  "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Payment..\">"  "<INPUT TYPE=\"submit\"
NAME=\"CMD\" VALUE=\"..Delivery..\">"
  "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Order-Status..\">"
  "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Stock-Level..\">"
  "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Exit..\">"
    "</pre>"
    "</body>"
    "</HTML>"
};

char *paymentResult = {
  "<html>"
    "<head>"
    "<title>TPC-C Payment</title>"
    "</head>"
    "<body>"
    "<FORM ACTION=tpcc.dll METHOD=\"GET\">"
  "<INPUT TYPE=\"hidden\" NAME=\"FORMID\" VALUE=\"4\">"
  "<INPUT TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"%s\">"
  "<INPUT TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"%s\">"
    "<center><h1> TPC-C Payment</h1></center>"
    "<pre>"
    "Date: %s<br><br>"
    "<br>Warehouse: %4.4d                          District: %2.2d"
    "<br>%19s%c              /* wrhs address 1  */
    "                    "    /* gap   */
    "    %19s%c              /* dstrct address 1  */
    "<br>%19s%c              /* wrhs address 2  */
    "                    "    /* gap   */
    "    %19s%c              /* dstrct address 2  */
    "<br>%19s%c              /* wrhs city       */
    "    %c%c               /* wrhs state two characters  */
    "    %8s%c               /* wrhs zip        */
    "    %19s%c              /* dstrct city     */
    "    %c%c               /* dstrct state two characters */
    "    %8s%c               /* dstrct zip      */
    "<br>                                                    "
    "<br>Customer: %4.4d  Cust-Warehouse: %4.4d  Cust-District: %2.2d"
    "<br>Name: %15s%c %c%c %15s%c      Since:  %9s "
    "<br>      %19s%c                  Credit: %c%c "
    "<br>      %19s%c                  %Disc: %2.2D(5,2)"
    "<br>      %19s%c  %c%c %10s%c   Phone: %17s%c"
    "<br>"
    "<br>Amount Paid:   $%6.2D(7,2)  New Cust-Balance: $%8.2D(13,2)"    "<br>Credit Limit:
$%10.2D(13,2) "
    "<br>"
    "<br>Cust-data: %49s%c"
    "<br>           %49s%c"
    "<br>           %49s%c"
    "<br>           %49s%c"
    "<br><br>"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..NewOrder..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Payment..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Delivery..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Order-Status..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Stock-Level..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Exit..\">"
    "</pre>"
    "</body>"
    "</HTML>"
};

char *paymentError = {
  "<html>"
    "<head>"
    "<title>TPC-C Payment Error</title>"
    "</head>"
    "<body>"
    "<FORM ACTION=tpcc.dll METHOD=\"GET\">"
  "<INPUT TYPE=\"hidden\" NAME=\"FORMID\" VALUE=\"4\">"
  "<INPUT TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"%s\">"
  "<INPUT TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"%s\">"
    "<center><h1> TPC-C Invalid payment data</h1></center>"
    "<pre>"
    "<br>Warehouse: %4.4d                          District: %2.2d"
    "<br>Customer:  %4.4d   Customer-Warhouse %4.4d  Customer-District: %2.2d"
    "<br>Name:   %15s%c  Type: %c"
    "Amount Paid: %9D(7,2)"
    "<br><br>"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..NewOrder..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Payment..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Delivery..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Order-Status..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Stock-Level..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Exit..\">"
    "</pre>"
    "</body>"
    "</HTML>"
};

char *ordstsResult = {
  "<html>"
    "<head>"
    "<title>TPC-C Order Status</title>"
    "</head>"
    "<body>"
    "<FORM ACTION=tpcc.dll METHOD=\"GET\">"
  "<INPUT TYPE=\"hidden\" NAME=\"FORMID\" VALUE=\"5\">"
  "<INPUT TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"%s\">"
  "<INPUT TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"%s\">"
    "<center><h1> TPC-C Order-Status</h1></center>"
    "<pre>"
    "<br>Warehouse: %4.4d      District: %2.2d"
    "<br>Customer: %4.4d  "
    "    Name: %15s%c"
    "    %c%c %15s%c"
    "<br>Cust-Balance: $%7.2D(13,2)"
    "<br>                "    /* gap   */
    "<br>Order-Number: %8.2d"
    "    Entry-Date: %s"
    "    Carrier-Number: %c%c"
    "<br><br>"
    "<table>"
    "<tr>"
    "<th>Supp-w  </th>"
    "<th>Item-Id  </th>"
    "<th>Qty     </th>"
    "<th>Amount</th>"
    "<th>Delivery-Date</th>"
    "</tr>"
    "</table>"
    "<br><br>"
};

char *ordstsTail = {
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..NewOrder..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Payment..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Delivery..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Order-Status..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Stock-Level..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Exit..\">"
    "</pre>"
    "</body>"
    "</HTML>"
};

char *ordstsError = {
  "<html>"
    "<head>"
    "<title>TPC-C Order Status Error</title>"
    "</head>"
    "<body>"
    "<FORM ACTION=tpcc.dll METHOD=\"GET\">"
  "<INPUT TYPE=\"hidden\" NAME=\"FORMID\" VALUE=\"5\">"
  "<INPUT TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"%s\">"
  "<INPUT TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"%s\">"
    "<center><h1> TPC-C Invalid order status data</h1></center>"
    "<pre>"
    "<br>Warehouse: %4.4d                          District: %2.2d"
    "<br>Customer:  %4.4d"
```

```
    "<br>Name:   %15s%c Type: %c"
    "<br><br>"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..NewOrder..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Payment..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Delivery..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Order-Status..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Stock-Level..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Exit..\">"
    "</pre>"
    "</body>"
    "</HTML>"
};


char *deliveryResult = {
  "<html>"
    "<head>"
    "<title>TPC-C Delivery</title>"
    "</head>"
    "<body>"
    "<FORM ACTION=tpcc.dll METHOD=\"GET\">"
  "<INPUT TYPE=\"hidden\" NAME=\"FORMID\" VALUE=\"6\">"
  "<INPUT TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"%s\">"
  "<INPUT TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"%s\">"
    "<center><h1> TPC-C Delivery</h1></center>"
    "<pre>"
    "<br>Warehouse: %4.4d"
    "<br>                "
    "<br>Carrier Number: %c%c"
    "<br>                "
    "<br>Execution Status: Delivery has been queued"
    "<br><br>"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..NewOrder..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Payment..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Delivery..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Order-Status..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Stock-Level..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Exit..\">"
    "</pre>"
    "</body>"
    "</HTML>"
};

char *stklvlResult = {
  "<html>"
    "<head>"
    "<title>TPC-C Stock Level</title>"
    "</head>"
    "<body>"
    "<FORM ACTION=tpcc.dll METHOD=\"GET\">"
  "<INPUT TYPE=\"hidden\" NAME=\"FORMID\" VALUE=\"7\">"
  "<INPUT TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"%s\">"
  "<INPUT TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"%s\">"
    "<center><h1>TPC-C Stock Level</h1></center>"
    "<pre>"
    "<br>Warehouse: %4.4d      District: %2.2d"
    "<br>                "
    "<br>Stock Level Threshold: %2.2d"
    "<br>                "
    "<br>Low Stock: %3.3d<br>"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..NewOrder..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Payment..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Delivery..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Order-Status..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Stock-Level..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Exit..\">"
    "</pre>"
    "</body>"
    "</HTML>"
};

char *signonResult = {
  "<html>"
    "<head>"
    "<title>TPC-C Sign On Error in CGI</title>"
    "</head>"
    "<body>"
    "<FORM ACTION=tpcc.dll METHOD=\"GET\">"
  "<INPUT TYPE=\"hidden\" NAME=\"FORMID\" VALUE=\"1\">"
  "<INPUT TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"-2\">"
  "<INPUT TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"0\">"
    "<h1>TPC-C Sign error detected!</h1>"
    "<center><h2>Please sign on again</h2></center>"
    "<center><h2>the signon inforation has been lost.</h2></center>"    "Warehouse ID <INPUT
NAME=\"w_id\" SIZE=4 MAXLENGTH=4><BR>"
    "District ID <INPUT NAME=\"d_id\" SIZE=2 MAXLENGTH=2><BR>"
    "<HR>"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"Submit\">"
    "</body>"
    "</HTML>"
};

char *dataErrorResult = {
  "<html>"
    "<head>"
    "<title>TPC-C Data Error in CGI</title>"
    "</head>"
    "<body>"
    "<FORM ACTION=tpcc.dll METHOD=\"GET\">"
  "<INPUT TYPE=\"hidden\" NAME=\"FORMID\" VALUE=\"2\">"
  "<INPUT TYPE=\"hidden\" NAME=\"TERMID\" VALUE=\"%s\">"
  "<INPUT TYPE=\"hidden\" NAME=\"SYNCID\" VALUE=\"%s\">"
    "<h1>TPC-C Data entry error detected!</h1>"
    "<center><h2>Please select a tranaction from the menu</h2></center>"
    "<center><h2>or select Back and correct the data</h2></center>"
    "<br>"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..NewOrder..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Payment..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Delivery..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Order-Status..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Stock-Level..\">"
    "<INPUT TYPE=\"submit\" NAME=\"CMD\" VALUE=\"..Exit..\">"
    "</body>"
    "</HTML>"
};



int postResult(void *res_buf, char * warehouse, char * district,
               int formIdPost, char * scratchBuffer )
{
  int          i;
  unsigned char *toPtr;/* storeage location                    */
  char         wzip_tag;/* ending character for warehouse zip   */
  char         dzip_tag;/* ending character for district zip    */
  char         czip_tag;/* ending character for customer zip    */
  char         clast_tag;/* ending of customers last name       */
  char         cfirst_tag;/* ending of customers first name     */
  char         waddress1_tag;/* ending of warehouse address one */
  char         waddress2_tag;/* ending of warehouse address two */
  char         daddress1_tag;/* ending of district address one  */
  char         daddress2_tag;/* ending of district address two  */
  char         wcity_tag;/* ending of warehouse city            */
  char         dcity_tag;/* ending of district city             */
  char         cphone_tag;/* ending of customers last name      */
  char         caddress1_tag;/* ending for customer address 1   */
  char         caddress2_tag;/* ending for customer address 1   */
  char         ccity_tag;/* ending for customer city            */
  char         cdata1_tag;/* ending for customer data 1         */
  char         cdata2_tag;/* ending for customer data 2         */
  char         cdata3_tag;/* ending for customer data 3         */
  char         cdata4_tag;/* ending for customer data 4         */
  char         item_name_tag;/* last character of the item name */
  char         dlv_time[27];/* time string used for convertion       */
  char         time[27];    /* converted output return 27 characters */
  char         cl_time[27]; /* converted output return              */
  _MI_Time     time_of_day; /* time of dat for new order transaction */

  new_order_display      *t;
  payment_display        *p;
  order_status_display   *o;
  dlv_input              *d;
  stock_display          *s;

  switch(formIdPost) {
    case NEW_ORDER:
    t = ( new_order_display *) res_buf;
    /* conver ordttm to date */
    /* getdtmstr( time, t->new_out->oenttm,   ) */
    mattod ( time_of_day);
    time_of_day[7] = 'B';/* set the period flag  */
    getdtmstr ( time_of_day, time, 'm' );
    /* returns a 27 character null terminated string */
    /* remove the mirosecound */
    time[19] = '\0'; /* terminate the string */
    if (t->new_out->neword_result == 'G')
    {
      clast_tag = t->new_out->clast[15];/* get last character */
      t->new_out->clast[15]= '\0';/* terminate string */
      sprintf(scratchBuffer,newOrderResult,
              warehouse,
              district,
              t->new_inp->cwid,
              t->new_inp->cdid,
              time,
              t->new_inp->cid,
              t->new_out->clast, clast_tag,
              t->new_out->ccredt[0],
              t->new_out->ccredt[1],
              t->new_out->cdct,
              t->new_out->oid,
              t->new_inp->number_of_items,
              t->new_out->wtax,
              t->new_out->dtax
              );

        toPtr = scratchBuffer + strlen(scratchBuffer);
        for (i=0; i<t->new_inp->number_of_items; ++i){
        item_name_tag = t->new_out->output_line[i].output_item_name[23];
        t->new_out->output_line[i].output_item_name[23]='\0';/* terminate */
        sprintf(toPtr,
        "%4d %6d   %23s%c %3D(3,0)  %3d   %c   $%6.2D(5,2)  $%7.2D(7   ,2)<br>",
                t->new_inp->input_line[i].olspwh,
                t->new_inp->input_line[i].oliid,
                t->new_out->output_line[i].output_item_name, item_name_tag,
                t->new_inp->input_line[i].olqty,
                t->new_out->output_line[i].output_stqty,
                t->new_out->output_line[i].output_borg,
                t->new_out->output_line[i].output_item_price,
                t->new_out->output_line[i].output_olamnt);
          toPtr += strlen(toPtr);
        }
        t->execution_status[0] = '\0';/* make blank */
      }
      else
      {
            sprintf(scratchBuffer,newOrderError,
              warehouse,
              district,
               t->new_inp->cwid,
               t->new_inp->cdid,
               time,
               t->new_inp->cid
              );

         toPtr = scratchBuffer + strlen(scratchBuffer);
         for (i=0; i<t->new_inp->number_of_items; ++i){
            sprintf(toPtr,
                " %4d  %6d  %3D(3,0)<br>",
                t->new_inp->input_line[i].olspwh,
                t->new_inp->input_line[i].oliid,
                t->new_inp->input_line[i].olqty );
            toPtr += strlen(toPtr);
          }
      t->new_out->totamt = 0.0;
      if (t->new_out->neword_result == 'I')
          strcpy( t->execution_status, "INVALID INPUT");/* set to ID    */
      else
          strcpy( t->execution_status, "Item number is not valid ");

    } /* end of the else */

      sprintf(toPtr,newOrderTail,
              t->execution_status,
              t->new_out->totamt);

      break;

    case PAYMENT:
    p = ( payment_display *) res_buf;
      /* terminate output strings */
      if(( p->pay_out->output_fmt_num[0] != '7' )||
         ( p->pay_out->output_fmt_num[1] != '7' ))
      {
           waddress1_tag = p->pay_out->waddr1[19];
           daddress1_tag = p->pay_out->daddr1[19];
           waddress2_tag = p->pay_out->waddr2[19];
           daddress2_tag = p->pay_out->daddr2[19];
           wcity_tag     = p->pay_out->wcity[19];
           dcity_tag     = p->pay_out->dcity[19];
           clast_tag     = p->pay_out->clast[15];
           cfirst_tag    = p->pay_out->cfirst[15];
           caddress1_tag = p->pay_out->caddr1[19];
           caddress2_tag = p->pay_out->caddr2[19];
           ccity_tag     = p->pay_out->ccity[19];
           cphone_tag    = p->pay_out->cphone[15];
           cdata1_tag    = p->pay_out->cdat1[49];
           cdata2_tag    = p->pay_out->cdat2[49];
           cdata3_tag    = p->pay_out->cdat3[49];
           cdata4_tag    = p->pay_out->cdat4[49];
           p->pay_out->waddr1[19]='\0';
           p->pay_out->daddr1[19]='\0';
           p->pay_out->waddr2[19]='\0';
           p->pay_out->daddr2[19]='\0';
           p->pay_out->wcity[19]='\0';
           p->pay_out->dcity[19]='\0';
           p->pay_out->clast[15]='\0';
           p->pay_out->cfirst[15]='\0';
```

```
          p->pay_out->caddr1[19]='\0';
          p->pay_out->caddr2[19]='\0';
          p->pay_out->ccity[19]='\0';
          p->pay_out->cphone[15]='\0';
          p->pay_out->cdat1[49]='\0';
          p->pay_out->cdat2[49]='\0';
          p->pay_out->cdat3[49]='\0';
          p->pay_out->cdat4[49]='\0';
          wzip_tag = p->pay_out->wzip[8];/* copy zip */
          dzip_tag = p->pay_out->dzip[8];/* copy zip */
          czip_tag = p->pay_out->czip[8];/* copy zip */
          p->pay_out->wzip[8] = '\0';/* terminate warehouse zip */
          p->pay_out->dzip[8] = '\0';/* terminate district zip  */
          p->pay_out->czip[8] = '\0';/* terminate customer zip  */
          /* convert time to string   */
          /* inputs                   */
          /* p->pay_out->cldate,      */
          getdttmstr ( p->pay_out->cldate, cl_time, 'm' );
          /* returns a 27 character null terminated string */
          /* remove the mirosecound */
          cl_time[19] = '\0'; /* terminate the string */          /*
p->pay_out->time_of_day,  */
          /* output cldate and time   */
          getdttmstr ( p->pay_out->time_of_day, time, 'm' );
          /* returns a 27 character null terminated string */
          /* remove the mirosecound */
          time[19] = '\0'; /* terminate the string */
          sprintf(scratchBuffer,paymentResult,
             warehouse,
             district,
             time,
             p->pay_inp->wid,
             p->pay_inp->did,
             p->pay_out->waddr1, waddress1_tag,
             p->pay_out->daddr1, daddress1_tag,
             p->pay_out->waddr2, waddress2_tag,
             p->pay_out->daddr2, daddress2_tag,
             p->pay_out->wcity,  wcity_tag,
             p->pay_out->wstate[0],
             p->pay_out->wstate[1],
             p->pay_out->wzip, wzip_tag,
             p->pay_out->dcity, dcity_tag,
             p->pay_out->dstate[0],
             p->pay_out->dstate[1],
             p->pay_out->dzip,  dzip_tag,
             p->pay_out->cid,
             p->pay_inp->cwid,
             p->pay_inp->cdid,
             p->pay_out->clast, clast_tag,
             p->pay_out->cinit[0],
             p->pay_out->cinit[1],
             p->pay_out->cfirst, cfirst_tag,
             cl_time,
             p->pay_out->caddr1, caddress1_tag,
             p->pay_out->ccredt[0],
             p->pay_out->ccredt[1],
             p->pay_out->caddr2, caddress2_tag,
             p->pay_out->cdct,
             p->pay_out->ccity, ccity_tag,
             p->pay_out->cstate[0],
             p->pay_out->cstate[1],
             p->pay_out->czip, czip_tag,
             p->pay_out->cphone, cphone_tag,
             p->pay_inp->amount,
             p->pay_out->cbal,
             p->pay_out->ccrdlm,
             p->pay_out->cdat1, cdata1_tag,
             p->pay_out->cdat2, cdata2_tag,
             p->pay_out->cdat3, cdata3_tag,
             p->pay_out->cdat4, cdata4_tag
             );
      }
      else
      {
         clast_tag = p->pay_inp->clast[15];
         p->pay_inp->clast[15] = '\0';
         sprintf(scratchBuffer,paymentError,
              warehouse,
              district,
                 p->pay_inp->wid,
                 p->pay_inp->did,
                 p->pay_inp->cid,
                 p->pay_inp->cwid,
                 p->pay_inp->cdid,
                 p->pay_inp->clast, clast_tag,
                 p->pay_inp->payment_type,
                 p->pay_inp->amount );
      }
      break;

    case ORDER_STATUS:
    o = ( order_status_display *) res_buf;
    if (( o->ordsts_out->ordsts_fmt_num[0]!='6')||
        ( o->ordsts_out->ordsts_fmt_num[1]!='2'))
    {
    /* convert time       o->ordsts_out->oettod,          */
    /* output is in time                                  */
      getdttmstr ( o->ordsts_out->oettod, time, 'm' );
      /* returns a 27 character null terminated string */
      /* remove the mirosecound */
      time[19] = '\0'; /* terminate the string */
    /* set temination for the customer first and last name */
      clast_tag = o->ordsts_out->clast[15];
      cfirst_tag = o->ordsts_out->cfirst[15];
      o->ordsts_out->clast[15] = '\0';
      o->ordsts_out->cfirst[15] = '\0';
      if ( o->ordsts_out->ocarid[0] == '\0' )
        o->ordsts_out->ocarid[0] = ' ';
      if ( o->ordsts_out->ocarid[1] == '\0' )
        o->ordsts_out->ocarid[1] = ' ';
      sprintf(scratchBuffer,ordstsResult,
              warehouse,
              district,
              o->ordsts_inp->wid,
              o->ordsts_inp->did,
              o->ordsts_out->ocid,
              o->ordsts_out->clast, clast_tag,
              o->ordsts_out->cinit[0],
              o->ordsts_out->cinit[1],
              o->ordsts_out->cfirst, cfirst_tag,
              o->ordsts_out->cbal,
              o->ordsts_out->oid,
              time, /* o->ordsts_out->oettm, */
              o->ordsts_out->ocarid[0],
              o->ordsts_out->ocarid[1]
              );

      toPtr = scratchBuffer + strlen(scratchBuffer);
      for (i=0; i<o->ordsts_out->olnenbr; ++i){
      /* conver orderline deliver time to time of day */
      /*                                              */
      /* o->ordsts_out->olineinfo[i].do_oldlvl        */
      /* output as dlv_time  char 15 output array     */
      /*  conver dlv_time to date time string         */
      getdttmstr ( o->ordsts_out->olineinfo[i].do_oldlvl, dlv_time, 'm' );
      /* returns a 27 character null terminated string */
      /* remove the mirosecound */
      dlv_time[19] = '\0'; /* terminate the string */
        sprintf(toPtr," %4d    %6d %2s %2d %6.2D(7,2) %14s<br>",
                o->ordsts_out->olineinfo[i].do_olspwh,
                o->ordsts_out->olineinfo[i].do_oliid,
                SPACE,
                o->ordsts_out->olineinfo[i].do_olqty,
                o->ordsts_out->olineinfo[i].do_olamnt,
                dlv_time );
        toPtr += strlen(toPtr);
      }

      sprintf(toPtr,ordstsTail);
      }
      else
      {
         clast_tag = o->ordsts_inp->clast[15];
         o->ordsts_inp->clast[15] = '\0';
         sprintf(scratchBuffer,ordstsError,
                 warehouse,
                 district,
                 o->ordsts_inp->wid,
                 o->ordsts_inp->did,
                 o->ordsts_inp->cid,
                 o->ordsts_inp->clast, clast_tag,
                 o->ordsts_inp->ordsts_type );
      }

      break;

    case DELIVERY:
    d = ( dlv_input *) res_buf;
      sprintf(scratchBuffer,deliveryResult,
              warehouse,
              district,
              d->wid,
              d->carrier[0],
              d->carrier[1]
              );
      break;

    case STOCK_LEVEL:
    s = ( stock_display *) res_buf;
      sprintf(scratchBuffer,stklvlResult,
              warehouse,
              district,
              s->stk_inp->wid,
              s->stk_inp->did,
              s->stk_inp->threshold,
              s->stk_out->blwstk
              );
      break;

    case SIGNON_ERROR:
      sprintf(scratchBuffer,signonResult );
      break;

    case DATA_ERROR:
      sprintf(scratchBuffer,dataErrorResult,
              warehouse, district );
      break;

    default:
      printf("postResult: Invalid form ID = %d", formIdPost );
      return (-1);
      break;
  } /* end of switch */


  return 0;
}
```

SCANINPUT.C:

```
/*****************************************************************************/
/* Change activity                                                           */
/*  9/5/97 - fix for empty string (ip->inptr++ source: 180,181)              */
/*  9/5/97 - fix for date conversion (removed ; on if) source 960            */
/*****************************************************************************/

#pragma comment (copyright, \
"TPCDIMP LIBRARY\
(C) Copyright IBM Corp. 1996,1997,1998.\
All rights reserved.\
US Government Users Restricted Rights-\
Use, duplication or disclosure restricted \
by GSA ADP Schedule Contract with IBM Corp.\
Licensed Materials-Property of IBM")

/*                                                           */
/* OCO Source Materials                                      */
/*                                                           */
/* The Source code for this program is not published or otherwise */
/* divested of its trade secrets, irrespective of what has been   */
/* deposited with the U.S. Copyright office                  */
/*                                                           */
/* TPCDIMP LIBRARY                                           */
/* (C) Copyright IBM Corp. 1996,1997,1998                    */
/*                                                           */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <stddef.h>
#include <math.h>
#include <ctype.h>
#include <decimal.h>
#include <recio.h>
#include <errno.h>
#include <time.h>
#include <xxfdbk.h>
#include <xxcvt.h>
#include <micomput.h>
#include <QSYSINC/MIH/MIDTTM>
#include <qsysinc/mih/cvtch>
#include "QUSLRCD.H"
#include "QUSLFLD.H"
#include "QUSGEN.H"
#include "QUSCRTUS.H"
#include "QUSDLTUS.H"
#include "QUSPTRUS.H"
#include "IMPORT.H"
#include "common.h"

typedef _Packed struct {
    short wid;
    short did;
```

```
    char carr_id[2];
} delivery_cgi_input;

typedef _Packed struct {
    short wid;
    short did;
    int   cid;
    char last[16];
} order_cgi_input;

typedef _Packed struct {
    short wid;
    short did;
    int   cid;
    char last[16];
    decimal(7,2) amount;
} pay_cgi_input;

static char chk_flag = 'R';

/*****************************************************************/
/*                                                               */
/*    Functions used by CGIBIN                                   */
/*                                                               */
/*****************************************************************/

int scan_new_order(char * input, new_order_input * output);
int scan_payment(char * input, payment_input * output);
int scan_order_status(char * input, order_status_input * output);
int scan_delivery(char * input, dlv_input * output);
int scan_stock_level(char * input, stock_input * output);
 /******************* ImportFields ********************************/
 /* Function: ImportFields ()  - Delimited ASCII format           */
 /*                                                               */
 /* Description: Parse import data to find fields                 */
 /*                                                               */
 /*****************************************************************/
void  ImportFields(ImportParms *ip)
{
 char * marker;
 /* input file action */
 ip->inptr = ip->colptr;              /* Start of first field         */
 *ip->outstart = ip->tran_type;       /* set the tranaction type      */
 ip->fldcnt  = 0;                     /* initialize field count to 0  */
 ip->fldlen  = 0;                     /* initialize field length to 0 */
 ci = ci_str;                         /* Set ptr to column info       */
 ip->outptr = ip->outstart+ci->offset;/* set first output item        */
 ip->header='Y';                      /*  This is header data         */
 ip->sign_on_error='N';               /* set sign on error flag       */
 ip->tran_error = 'N';                /* reset transaction error      */
 ip->end_trans='N';                   /*  Not the end transaction     */
 for (ip->ix=0; ip->ix <= ip->buflen; ip->ix++) /* Delimiter scanning */
 {
    if(ip->header!='N')
    {
       if (*ip->colptr == '=')/* check for start of data */
       {
          ip->header='N';
          ip->Blankcnt   = 0;       /* reset blank count            */
          ip->inptr = ip->colptr+1;/* Set start of next field       */
          ip->fldlen  = 0;          /* initialize field length to 0 */
       }
    }
    else /* normal path if not header */
    {
       if(ip->end_trans!='N')/* process the end of new order transact  i on */
       {
          if((*ip->colptr==ip->row_delimit)/* No, Row delimiter found?  */
          ||(*ip->colptr==ip->col_delimit))/* No, Column delimter found?*/
          {
            ip->header='Y';/* data is ok continue next data is header */
            memset ( ip->outptr, 0, ci->fldlen );/* clear the data field */
            if ( ci->MoveField_code == PACKED)
            {
               marker = ip->outptr;/* get a pointer to the output */
               marker[ci->fldlen-1]=0xf;/* set the sign byte    */
            }
            ++ip->fldcnt;   /* add one to field count            */
            if (ip->fldcnt !=ip->columnCnt) /* Processed all columns? */
            {                    /* No, then .....              */              ip->inptr =
ip->colptr+1;/* Set start of next field  */
               ip->fldlen  = 0;  /* initialize field length to 0     */
               ci++;        /* Advance to next column info          */
               ip->outptr = ip->outstart+ ci->offset;/* move to next */
            }
            else              /* If all columns processed....        */
              break;          /* Yes, set to exit for loop            */
            if(*ip->colptr==ip->col_delimit)/* Column delimter found ?*/
              break;          /* Yes, set to exit for loop            */
          }
          else  /* data  found */
          {
             ip->tran_error = 'Y';    /* set the error flag     */
             break; /* exit */
          } /* end of else */
       }/* end of if end_tran ending tranaction action */
       else /* normal path if not header */
       {
   if (*ip->colptr!=ip->col_delimit)  /* No, Column delimter found?   */
    if (*ip->colptr!=ip->row_delimit) /* No, Row delimiter found?     */
     if (*ip->colptr != '%')          /* not a specal character       */
      if (*ip->colptr != '+')         /* No, blank found?             */
       if (ci->number_chk!='1')       /* No, Integer scan?            */
         ip->Blankcnt   = 0;          /* Reset trailing blank count   */
       /****************/
       /* Integer scan */
       /****************/
       else                           /* Yes, integer scan required   */
       {                              /* Valid number, + or - ?       */
         switch (*ip->colptr)
          {
           case '0':                  /* For numbers, make sure ..... */
           case '1':                  /* that there is no imbedded..  */
           case '2':                  /* blank characters.            */
           case '3':
           case '4':
           case '5':
           case '6':
           case '7':
           case '8':
           case '9':
            if (ip->Blankcnt==0)      /* Trailing blank found?        */
               ip->Numeric.Data=1;    /* No, indicate data present    */
            else                      /* Yes, then....                */
            {
               ip->tran_error='Y';    /* digit after a triling blank  */
               ip->ix=ip->buflen;     /* set end condition            */
            }
            break;
           case '.':                  /* Decimal point detected?      */
            if (ip->decmpt == '.')    /* Period allowed? yes...       */
               ip->decptr = ip->colptr;/* Save pointer to decimal pt  */
            else
            {
               ip->tran_error='Y';    /* decimal point not allowed    */
               ip->ix=ip->buflen;     /* set end condition            */
            }
            break;
           default:                   /* Not a number or decimal point*/
              ip->tran_error='Y';     /* no digit in number field     */
              ip->ix=ip->buflen;      /* set end condition            */
            break;
          } /* end of switch */
       } /* end else integer scan */
       /***************/
       /* Blank found */
       /***************/
       else                           /* Blank found                  */
        {                             /* Yes, blank found then ...    */
            if (ip->inptr==ip->colptr)  /* Leading blank?             */
              ip->inptr++;            /* Yes, advance start address   */
            else                      /* No, possible trailing blank  */
            {
               ip->Blankcnt++;        /* Increment trailing blank cnt */
               *ip->colptr = ' ';     /* replace + with a blank       */
            }
       } /* end else blank */
       /************************/
       /* Special character % */
       /************************/
        else
        {
          ip->tran_error = 'Y';       /* set the error flag           */
          break; /* exit */           /* no special character are     */
                                      /* allowed.                     */
        }
       /*****************/
       /* Row delimiter */
       /*****************/
       else                           /* Row delimiter                */
       {
         ColRowDelimiter(ip);         /* Process row delimiter        */
         ip->header = 'Y';           /* next part is header           */
       } /* end of the else */
       /*********************/
       /* Column delimiter */
       /*********************/
       else                           /*                              */
       {
         ColRowDelimiter(ip);         /* Process column delimiter     */
         ip->header = 'Y';            /* first part of data is header */
       }/* end of the else Column delimiter */
       } /* end else header          */
       } /* end else end_trans       */


        ++ip->colptr;                 /* Point to next buffer char    */
    }  /* end for loop  */
    if (ip->tran_error == 'N')/* if there are no transaction errors */
    {
       if (ip->fldcnt != ip->columnCnt)  /* Error if number of columns   */
       {
           ip->tran_error = 'Y';      /* set the error flag           */
           ip->ix=ip->buflen;         /* set end condition            */
       }
                                      /*  doesn't match output file   */
    }
    else
    {
        if(ip->fldcnt<2)
        {
           ip->sign_on_error = 'Y';
        }
    }
    return;
}
 /******************** ColRowDelimiter *********************************/
 /* Function: ColRowDelimter (ICB)                                     */
 /*                                                                    */
 /* Description: Find input fields and create output columns           */
 /*                                                                    */
 /**********************************************************************/
void   ColRowDelimiter ( ImportParms *ip)
{
    check_input_string( ip );       /* check the input string for     */
                                    /* valid tranactions              */
    ++ip->fldcnt;                   /* add one to field count         */
    if (ip->fldcnt !=ip->columnCnt) /* Processed all columns?         */
    {                               /* No, then .....                 */
       ip->inptr =  ip->colptr+1;   /* Set start of next field        */
       ip->fldlen  = 0;             /* initialize field length to 0   */
       ci++;                        /* Advance to next column info    */
       ip->outptr = ip->outstart + ci->offset;/* set output pointer   */
    }
    else                            /* If all columns processed....   */
       ip->ix = ip->buflen;         /* Yes, set to exit for loop      */
 return;
 }
 /******************** MoveField ***********************************/
 /* Function: MoveField (ip   )                                    */
 /*                                                                */
 /* Description: Build output database records                     */
 /*                                                                */
 /******************************************************************/
void   MoveField(ImportParms *ip)
{
 int n;                            /* Array index                  */
 short small_int;                  /* Small integer hold area      */
 short difflen;                    /* Length of current input field */
 short left;                       /* Data length left of decimal  */
 short right;                      /* Data length right of decimal  */
 short offset;                     /* Offset to data start         */
 long long_int;                    /* Long integer hold area       */
 int total_digits;                 /* Precision for decimal numbers */
 int fractional_digits;            /* Scale for decimal numbers    */
 char number[31] = "0";            /* A number hold area           */
 char blank = ' ';                 /* A blank character            */
 char  dash = '-';
 char  period = '.';
 char  hilow = 0;
 char  exitflg;
 char  * nmbptr;
 exitflg  = 0;                      /* Initialize continue flag     */
                                    /* Column data present?         */
 n = ip->fldcnt;                    /* get current field number     */
 if (!exitflg)  {
   switch (ci->MoveField_code)
   {
     /********************/
     /* CHAR and GRAPHIC */
     /********************/
     case NUMARIC_CHAR:              /* CHARACTER output numaric     */
       if ( ip->Numeric.Data!=1)     /* no data present              */
       {
          ip->tran_error='Y';        /* numeric data not present     */
          ip->ix=ip->buflen;         /* set end condition            */
          break;
       }
```

```
  else
   ip->Numeric.Data = 0;              /* reset numeric flags       */
                                      /* proesss as normal alph    */
                                      /* character strings         */
case CHAR_GRAPH:                      /* CHARACTER and GRAPHIC     */
 /* Copy data to  buffer and pad with blanks                       */
 difflen = ci->fldlen - ip->fldlen;/* Column len-input len         */
 if (difflen == 0 )                   /* Blank pad required?       */
                                      /* No, just move data        */
    memmove(ip->outptr,ip->inptr,ip->fldlen);
 else                                 /* Check for possible truncate */
 {
    if (difflen > 0 )                 /* Input length < column len?  */
    {                                 /* yes, Move data and then .... */
                                      /* Pad with blanks           */
       memmove(ip->outptr,ip->inptr,ip->fldlen);
       memset(ip->outptr + ip->fldlen, blank, difflen);
    }
    else                              /* Else, truncate data       */
    {                                 /* and send error message    */
       memset(ip->outptr,' ',ci->fldlen);/* set to blanks */
       ip->tran_error = 'Y';          /* set error flag            */
       ip->ix=ip->buflen;             /* end the scaning           */
    }
 }
 break;
/*****************/
/* SMALL INTEGER */
/*****************/
  case SMALLINT:                      /* SMALLINT                  */
   small_int = atoi(ip->inptr);       /* Convert char to integer   */
   memmove(ip->outptr,&small_int,2); /* Move integer to buffer    */
   if (ip->fldlen >  4 )              /* Valid number exists       */
    {
      if (ip->fldlen == 5 )
       {
         if (strcmp(ip->inptr, "32767") > 0)
            ip->tran_error = 'Y';     /* Set errors indicator     */
            memset(ip->outptr,0,2);   /* Move integer to buffer   */
            ip->ix=ip->buflen;        /* end the scaning          */
       }
      else
       {
            ip->tran_error = 'Y';     /* Set errors indicator     */
            memset(ip->outptr,0,2);   /* Move integer to buffer   */
            ip->ix=ip->buflen;        /* end the scaning          */
       }
    }
   ip->Numeric.Data = 0;              /* reset numeric flags       */
   ip->decptr = NULL;                 /* Null decimal point ptr    */
   break;
/************/
/* INTEGER */
/************/
  case INTEGER:                       /* INTEGER                   */
   long_int = atol(ip->inptr);        /* Convert data to integer   */
   memmove(ip->outptr, &long_int,4); /* Move integer to buffer    */
   if (ip->fldlen >  9 )              /* Valid number exists       */
    {
      if (ip->fldlen == 10)
       {
         if (strcmp(ip->inptr, "2147483647") > 0)
            ip->tran_error = 'Y';     /* Set errors indicator     */
            memset(ip->outptr,0,4);   /* Move integer to buffer   */
            ip->ix=ip->buflen;        /* end the scaning          */
       }
      else
       {
            ip->tran_error = 'Y';     /* Set errors indicator     */
            memset(ip->outptr,0,4);   /* Move integer to buffer   */
            ip->ix=ip->buflen;        /* end the scaning          */
       }
    }
   ip->Numeric.Data = 0;              /* reset numeric flags       */
   ip->decptr = NULL;                 /* Null decimal point ptr    */
   break;
/******************/
/* PACKED DECIMAL */
/******************/
  case PACKED:                    /* Packed DECIMAL              */
      memset(ip->outptr,0,ci->fldlen);
      if (ip->decptr!=NULL)       /* Decimal point found?        */
       {                          /* Yes, determine length ....  */
        left=ip->decptr-ip->inptr;/* left of decimal and..       */
        right=ip->fldlen-(left+1);/* data right of decimal       */
        if (right > ci->scale)  /* right len cannot exceed...   */
        {
            ip->tran_error='Y';  /* too many digits in input fiel d*/
            ip->ix = ip->buflen; /* reset the for loop index      */
            nmbptr = ip->outptr+ci->fldlen-1;/* Point to last byte */
            *nmbptr = *nmbptr | 0x0F;/* set posivitive sign        */
            ip->Numeric.Data = 0;   /* reset numeric flags        */
            ip->decptr = NULL;      /* Null decimal point ptr     */
            break;
        }
       }
      else                        /* If no decimal, then ...     */
       {                          /* there is no right of decimal */
        right = 0;                /* Set right data length to 0  */
        left = ip->fldlen;        /* Set left data length to...  */
        if ( left > ci->precision)
        {
          ip->tran_error='Y';     /* too many digits in input field*/
          ip->ix = ip->buflen;    /* reset the for loop index     */
          nmbptr = ip->outptr+ci->fldlen-1;/* Point to last byte   */
          *nmbptr = *nmbptr | 0x0F;/* set posivitive sign         */
          ip->Numeric.Data = 0;   /* reset numeric flags          */
          ip->decptr = NULL;      /* Null decimal point ptr       */
          break;
        }
       }
      total_digits=left+right;
      offset = (ci->fldlen*2-(left+ci->scale+1))/2;
      hilow  = (ci->fldlen*2-(left+ci->scale+1)) %2;
      if (hilow==0)
       {
         if (left > 0)            /* Move left data if present    */
            memcpy(&number[0],ip->inptr,left);
         if (right > 0)           /* Move right data if present   */
            memcpy(&number[left],ip->decptr+1,right);
       }
      else
       {
         if (left > 0)            /* Move left data if present    */
             memcpy(&number[1],ip->inptr,left);
         if (right > 0)           /* Move right data if present   */
             memcpy(&number[left+1],ip->decptr+1,right);
         total_digits++;          /* Adjust conversion digits     */
       }
      nmbptr = ip->outptr+offset; /* Point to start byte          */
      cvtch(nmbptr,number,total_digits);
      nmbptr = ip->outptr+ci->fldlen-1; /* Point to last byte     */
      *nmbptr = *nmbptr | 0x0F; /* set posivitive sign           */
   ip->Numeric.Data = 0;              /* reset numeric flags       */
         ip->decptr = NULL;           /* Null decimal point ptr    */
         break;


      }
     }
  return;
 }
 /******************* scan_new_order ********************************/
 /* Function: scan_new_order( char *, new_order_input)               */
 /*                                                                  */
 /* Description: scan the new order input stream                     */
 /*                                                                  */
 /********************************************************************/
int scan_new_order(char * input, new_order_input * output)
{
 static ImportParms   ipcb;
 static ImportParms  *ip = &ipcb;
 static column_info new_order_info[49];/* fixed column info          */
 static column_info * ci;         /* data pointer                    */
 static char first_pass = '0';      /* first pass                    */
 int count;                       /* for loop counter                */
 int cnt;                         /* Count characters in field       */
 long i;                          /* Loop variable                   */
 char * numptr;                   /* pointer to number of lines count */

 /********************************************************************/
 /*  Read the Import file and build the database records             */
 /*  Write the output file using arrival sequence                    */
 /*  (uses locate mode)                                              */
 /********************************************************************/
 /********************************************************************/
 /*  Delimiter options                                               */
 /*                                                                  */
 /*   - Set column, row and string delimiter values                  */
 /********************************************************************/
 if ( first_pass == '0' )
 {
   ip->buflen = BufferSize;             /* Get import buffer length   */
   ip->columnCnt=49;                    /* Set number of columns      */
   ip->col_delimit='\0';                /*  Set col delimiter         */
   ip->tran_type = 'N';                 /* set transaction type       */
   ip->row_delimit='&';                 /*  Get row delimiter         */
   ip->decmpt='\0';                     /* Set the decimal point      */
                                        /* decimal point not allowed */
   for ( count=0;count<3;count++)
   {
      ci= &new_order_info[count];     /* Setup data inputs            */
      ci->fldlen=2;                   /* Set field length             */
      ci->fldtype='B';                /* Set field data type          */
      ci->number_chk = '1';           /* Enable checking for numbers  */
      ci->MoveField_code=SMALLINT;    /* Yes, set code to small int   */
   }
   new_order_info[0].offset = 11;     /* warehouse ID input           */
   new_order_info[0].skip = 'N';      /* do not skip warehouse id     */
   new_order_info[1].offset = 1;      /* district ID input            */
   new_order_info[1].skip = 'Y';      /* skip district input          */
   new_order_info[2].offset = 13;     /* customer district ID input   */
   new_order_info[2].skip = 'N';      /* do not skip customer district input */
   new_order_info[3].offset = 15;     /* customer ID input            */
   new_order_info[3].skip = 'N';      /* do not skip customer ID input*/
   ci= &new_order_info[3];          /* Setup data inputs              */
   ci->fldlen=4;                    /* Set field length               */
   ci->fldtype='B';                 /* Set field data type            */
   ci->MoveField_code=INTEGER;      /* Yes, set code to small int     */
   ci->number_chk = '1';            /* Enable checking for numbers    */
   for ( count=0;count<15;count++)
   {
      ci= &new_order_info[count*3+4]; /* Setup data inputs            */
      ci->fldlen=2;                   /* Set field length             */
      ci->offset = 21 + ( count * 8); /* set the offset               */
      ci->skip = 'N';                 /* do not skip                  */
      ci->fldtype='B';                /* Set field data type          */
      ci->number_chk = '1';           /* Enable checking for numbers  */
      ci->MoveField_code=SMALLINT;    /* Yes, set code to small int   */
      ci= &new_order_info[count*3+5]; /* Setup data inputs            */
      ci->fldlen=4;                   /* Set field length             */
      ci->offset = 23 + ( count * 8); /* set the offset               */
      ci->skip = 'N';                 /* do not skip                  */
      ci->fldtype='B';                /* Set field data type          */
      ci->number_chk = '1';           /* Enable checking for numbers  */
      ci->MoveField_code=INTEGER;     /* Yes, set code to small int   */
      ci= &new_order_info[count*3+6]; /* Setup data inputs            */
      ci->fldlen =2;                  /* Set field length             */
      ci->offset = 27 + ( count * 8); /* set the offset               */
      ci->skip = 'N';                 /* do not skip                  */
      ci->fldtype ='P';               /* Set field data type          */
      ci->precision=3;                /* Set precision value          */
      ci->scale=0;                    /* Set scale value              */
      ci->left=ci->precision-ci->scale;/* data left of decimal        */
     ci->number_chk = '1';            /* Enable checking for numbers  */
     ci->MoveField_code = PACKED;      /* Set packed decimal code     */
   }
   first_pass = '1';                  /* set first pass               */
 }/* end of if first pass */
 ci_str = &new_order_info[0];
 ip->outstart = (char*)output;      /* set the output location       */
 ip->colptr = input;                /* set the start of the input    */
 ip->new_count = 15;                /* set number of lines for full  */
                                    /* input lines                   */
 ImportFields(ip);                  /* Find import fields            */
 if (ip->tran_error == 'N')
 {
    numptr = ip->outstart + 19;
    memcpy( numptr, &(ip->new_count), 2 );/* save number of lines */
    return (0);
 }
 else
 {
    if (ip->sign_on_error == 'N' )
    {
       return (80);
    }
    else
       return (100);
 }
}
 /******************* scan_payment **********************************/
 /* Function: scan_payment( char *, payment_input)                   */
 /*                                                                  */
 /* Description: scan the payment input stream                       */
 /*                                                                  */
 /********************************************************************/
int scan_payment(char * input, payment_input * output)
{
 static ImportParms   pay_ipcb;
 static ImportParms  *pay_ip = &pay_ipcb;
 static column_info payment_info[8];/* fixed column info             */
 static column_info * ci;           /* data pointer                  */
 static char first_pass = '0';      /* first pass                    */
 int count;                         /* for loop counter              */
 char * typeptr;                    /* payment type pointer          */
 /********************************************************************/
 /*  Read the Import file and build the database records             */
```

```
 /*  Write the output file using arrival sequence                    */
 /*  (uses locate mode)                                              */
 /********************************************************************/
 if ((strstr(input, "&DID*="))==NULL) chk_flag = 'B';
 else chk_flag = 'R';
 if ( first_pass == '0' )
 {
   pay_ip->buflen = BufferSize;          /* Get import buffer length   */
   pay_ip->columnCnt=8;                  /* Set number of columns      */
   pay_ip->col_delimit='\0';             /* Set col delimiter          */
   pay_ip->tran_type = 'P';              /* set transaction type       */
   pay_ip->row_delimit='&';              /* Get row delimiter          */
   pay_ip->decmpt='.';                   /* Set the decimal point      */
   pay_ip->tran_error='N';               /* Set error flag tp none     */
   for ( count=0;count<=6;count++)
   {
        ci = &payment_info[count];    /* Setup data inputs           */
        ci->fldlen =2;                /* Set field length            */
        ci->fldtype ='B';             /* Set field data type         */
        ci->number_chk = '1';         /* Enable checking for numbers  */
        ci->MoveField_code=SMALLINT;  /* Yes, set code to small int   */
        if (count==3)
        {
          ci->fldlen =4;              /* Set field length            */
          ci->MoveField_code=INTEGER;/* Yes, set code to integer     */
        }
    }
    if (chk_flag == 'B' ) {
    ci= &payment_info[6];             /* Setup data inputs           */
    ci->fldlen =16;                   /* Set field length            */
    ci->fldtype ='A';                 /* Set field data type         */
    ci->MoveField_code=CHAR_GRAPH;    /* Set MoveField code          */
    ci->number_chk = '0';             /* Disable checking for numbers */
    }
    else {
    ci= &payment_info[4];             /* Setup data inputs           */
    ci->fldlen =16;                   /* Set field length            */
    ci->fldtype ='A';                 /* Set field data type         */
    ci->MoveField_code=CHAR_GRAPH;    /* Set MoveField code          */
    ci->number_chk = '0';             /* Disable checking for numbers */
    }
    ci= &payment_info[7];             /* Setup data inputs           */
    ci->fldlen =4;                    /* Set field length            */
    ci->fldtype ='P';                 /* Set field data type         */
    ci->precision=7;                   /* Set precision value        */
    ci->scale=2;                       /* Set scale value            */
    ci->left=ci->precision-ci->scale; /* data left of decimal        */
    ci->number_chk = '1';              /* Enable checking for numbers */
    ci->MoveField_code = PACKED;      /* Set packed decimal code     */
    payment_info[0].offset = 14;    /* first record warehouse        */
    payment_info[1].offset = 1;     /* 2nd record sign on district   */
    payment_info[2].offset = 12;    /* 3rd record district           */
    payment_info[3].offset = 16;    /* 4th record customer number    */
    if (chk_flag == 'B') {
    payment_info[4].offset = 22;    /* 5th record customer warehouse id*/
    payment_info[5].offset = 20;    /* 6th record customer district  */
    payment_info[6].offset = 28;    /* 7th record customer last name */
    }
    else {
    payment_info[4].offset = 28;    /* 5th record customer last name*/
    payment_info[5].offset = 22;    /* 6th record customer warehouse */
    payment_info[6].offset = 20;    /* 7th record customer district  */
    }
    payment_info[7].offset = 24;    /* 8th record amount paid        */
    payment_info[0].skip = 'N';     /* first record district         */
    payment_info[1].skip = 'Y';     /* first record warehouse        */
    payment_info[2].skip = 'N';     /* secound record                */
    payment_info[3].skip = 'N';     /* third record                  */
    payment_info[4].skip = 'N';     /* fourth record                 */
    payment_info[5].skip = 'N';     /* fifth record                  */
    payment_info[6].skip = 'N';     /* sixth record                  */
    payment_info[7].skip = 'N';     /* sixth record                  */
    first_pass = '1';               /* set first pass                */
 } /* end of if first pass == '0' */
   ci_str = &payment_info[0];          /* set the starting point     */
   pay_ip->outstart = (char *) output;  /* set the output location    */
   pay_ip->colptr = input;             /* set the start of the input */
   pay_ip->last_name = 'Y';            /* There name be a last name  */
   pay_ip->cust_number = 'Y';          /* There may be a cust number */
   ImportFields(pay_ip);               /* Find import fields         */
   if (pay_ip->tran_error == 'N')
   {
      typeptr = pay_ip->outstart + 11;  /* location of payment type   */
      if ((pay_ip->last_name=='Y')&&(pay_ip->cust_number!='Y'))
      {
           *typeptr = 'L';/* payment type by last name */           return 0;/* go back done
*/

      }
      else
      {
         if ((pay_ip->last_name!='Y')&&(pay_ip->cust_number=='Y'))
         {
              *typeptr = 'C';/* payment type by customer number      */
              return 0;/* go back done */
         }
         else
         {
             return (80); /* both last name and cuctomer set */
         }
      }
   } /* end if tran_error == 'N' */
   else
   {
    if (pay_ip->sign_on_error == 'N' )
    {
      return (80);
    }
    else
       return (100);
   }/* end of else */
   return 0;
}
 /******************* scan_order_status ****************************/
 /* Function: scan_order_status( char *, order_status_input *)       */
 /*                                                                  */
 /* Description: scan the order status input stream                  */
 /*                                                                  */
 /********************************************************************/
int scan_order_status(char * input, order_status_input * output)
{
 static ImportParms   ipcb;
 static ImportParms  *ip = &ipcb;
 static column_info order_status_info[5];/* fixed column info        */
 static column_info * ci;           /* data pointer                  */
 static char first_pass = '0';      /* first pass                    */
 int count;                         /* for loop counter              */
 char * typeptr;                   /* payment type pointer           */
 /********************************************************************/
 /*  Read the Import file and build the database records             */
 /*  Write the output file using arrival sequence                    */
 /*  (uses locate mode)                                              */
 /********************************************************************/
 if ( first_pass == '0' )
 {
   ip->buflen = BufferSize;             /* Get import buffer length   */
   ip->columnCnt=5;                     /* Set number of columns      */
   ip->col_delimit='\0';                /*  Set col delimiter         */
   ip->tran_type = 'O';                 /* set transaction type       */
   ip->row_delimit='&';                 /*  Get row delimiter         */
   ip->decmpt='\0';                     /* Set the decimal point      */
   for ( count=0;count<4;count++)
   {
        ci = &order_status_info[count];/* Setup data inputs           */
        ci->fldlen =2;                /* Set field length            */
        ci->fldtype ='B';             /* Set field data type         */
        ci->number_chk = '1';         /* Enable checking for numbers  */
        ci->MoveField_code=SMALLINT;  /* Yes, set code to small int   */
        if (count==3)
        {
          ci->fldlen =4;              /* Set field length            */
          ci->MoveField_code=INTEGER;/* Yes, set code to intiger     */
        }
    }
    ci= &order_status_info[4];       /* Setup data inputs           */
    ci->fldlen =16;                  /* Set field length            */
    ci->fldtype ='A';                /* Set field data type         */
    ci->MoveField_code=CHAR_GRAPH;   /* Set MoveField code          */
    ci->number_chk = '0';            /* Disable checking for numbers */
    order_status_info[0].offset = 18;/* warehouse from sign on      */
    order_status_info[0].skip  = 'N';/* warehouse from sign on      */
    order_status_info[1].offset = 1; /* district from sign on       */
    order_status_info[1].skip   ='Y';/* district from sign on       */
    order_status_info[2].offset = 16;/* district entered on screen  */
    order_status_info[2].skip   ='N';/* district entered on screen  */
    order_status_info[3].offset = 12;/* customer id offset          */
    order_status_info[3].skip   ='N';/* customer id skip            */
    order_status_info[4].offset = 20;/* customer last name offset   */
    order_status_info[4].skip   ='N';/* customer last name skip     */
    first_pass = '1';                /* set first pass              */
 } /* end of if first pass == '0' */
 ci_str = &order_status_info[0];       /* set the starting point     */
 ip->outstart = (char *)output;        /* set the output location    */
 ip->colptr = input;                   /* set the start of the input */
 ip->last_name = 'Y';                  /* There name be a last name  */
 ip->cust_number = 'Y';                /* There may be a cust number */
 ImportFields(ip);                     /* Find import fields         */
 if (ip->tran_error == 'N')
 {
    typeptr = ip->outstart + 11;  /* location of order status type    */
    if ((ip->last_name=='Y')&&(ip->cust_number!='Y'))
    {
         *typeptr = 'L';/* order status type by last name            */
         return 0;
      }
     else
     {
        if ((ip->last_name!='Y')&&(ip->cust_number=='Y'))
        {
           *typeptr = 'C';/* order status type by customer number    */
           return 0;
        }
        else
          return (95);/* return with an error */
       }
  }
  else /* tran_error not 'N' */
  {
     if (ip->sign_on_error == 'N' )
     {
        return (80);
     }
     else
        return (100);
  }/* end of else */
  return 0;
}
 /******************* scan_delivery ****************************/
 /* Function: scan_delivery( char *, dlv_input *)                    */
 /*                                                                  */
 /* Description: scan the delivery input stream                      */
 /*                                                                  */
 /********************************************************************/
int scan_delivery(char * input, dlv_input * output)
{
 static ImportParms   ipcb;
 static ImportParms  *ip = &ipcb;
 static char first_pass = '0';      /* first pass                    */
 static column_info delivery_info[4];/* fixed column info            */
 static column_info * ci;           /* data pointer                  */
 int cnt;                           /* Count characters in field     */
 int count;                         /* for loop counter              */
 long i;                            /* Loop variable                 */
 /********************************************************************/
 /*  Read the Import file and build the database records             */
 /*  Write the output file using arrival sequence                    */
 /*  (uses locate mode)                                              */
 /********************************************************************/
 if ( first_pass == '0' )
 {
   ip->buflen = BufferSize;             /* Get import buffer length   */
   ip->columnCnt=3;                     /* Set number of columns      */
    ip->col_delimit='\0';                    /*  Set col delimiter    */
    ip->tran_type = 'D';                     /* set transaction type  */
    ip->row_delimit='&';                     /*  Get row delimiter    */
    ip->decmpt='\0';                         /* Set the decimal point */
    for ( count=0;count<2;count++)
    {
       ci= &delivery_info[count];        /* Setup data inputs         */
       ci->fldlen=2;                     /* Set field length          */
       ci->fldtype='B';                  /* Set field data type       */
       ci->number_chk = '1';             /* Enable checking for numbers  */
       ci->MoveField_code=SMALLINT;   /* Yes, set code to small int   */
    }
    ci= &delivery_info[2];            /* Setup data inputs          */
    ci->fldlen=2;
    ci->fldtype='A';                  /* set for alph inputs        */
    ci->MoveField_code=NUMARIC_CHAR;  /* Set MoveField code         */
    ci->number_chk = '1';             /* Enable checking for numbers  */
    delivery_info[0].offset=11;       /* Setup data warehouse       */
                                      /* distric data not used      */
    delivery_info[1].offset=11;       /* Setup district             */
    delivery_info[2].offset=13;       /* Setup data carrier number  */
    delivery_info[0].skip ='N';       /* Setup data district        */
                                      /* distric data not used      */
    delivery_info[1].skip ='Y';       /* Setup data warehouse       */
    delivery_info[2].skip ='N';       /* Setup data carrier number  */
    first_pass = '1';                 /* set first pass             */
 }
 ci_str = &delivery_info[0];
 ip->outstart = (char*) output;       /* set the output location    */
 ip->colptr = input;                  /* set the start of the input */
 ImportFields(ip);                    /* Find import fields         */
 if( ip->tran_error != 'N') /* tran_error not 'N' */
 {
   if (ip->sign_on_error == 'N' )
   {
      return (80);
```

```
    }
    else
       return (100);
  }
  return 0;
}
 /******************* scan_stock_level ******************************/
 /* Function: scan_stock_level( char *, stock_input *)               */
 /*                                                                  */
 /* Description: scan the delivery input stream                      */
 /*                                                                  */
 /********************************************************************/
int scan_stock_level(char * input, stock_input * output)
{
 static ImportParms   ipcb;
 static ImportParms  *ip = &ipcb;
 static column_info stock_info[4];/* fixed column info               */
 static column_info * ci;        /* data pointer                     */
 static char first_pass = '0';      /* first pass                    */
 int count;                      /* for loop counter                 */
 int cnt;                        /* Count characters in field        */
 long i;                         /* Loop variable                    */
 char * belowptr;                 /* pointer to the below field      */

 /********************************************************************/
 /*  Read the Import file and build the database records             */
 /*  Write the output file using arrival sequence                    */
 /*  (uses locate mode)                                              */
 /********************************************************************/
 /********************************************************************/
 /*  Delimiter options                                               */
 /*                                                                  */
 /*   - Set column, row and string delimiter values                  */
 /********************************************************************/
 if ( first_pass == '0' )
 {
   ip->buflen = BufferSize;             /* Get import buffer length   */
   ip->columnCnt=3;                     /* Set number of columns      */
   ip->col_delimit='\0';                 /*  Set col delimiter        */
   ip->tran_type = 'S';                  /* set transaction type      */
   ip->row_delimit='&';                  /*  Get row delimiter        */
   ip->decmpt='\0';                      /* Set the decimal point     */
   for ( count=0;count<3;count++)
   {
      ci= &stock_info[count];          /* Setup data inputs            */
      ci->fldlen=2;                    /* Set field length             */
      ci->fldtype='B';                 /* Set field data type          */
      ci->number_chk = '1';            /* Enable checking for numbers  */
      ci->skip = 'N';                  /* Do not skip this item        */
      ci->offset = 11 + (2 * count);   /* set the off set              */
      ci->MoveField_code=SMALLINT;     /* Yes, set code to small int   */
   }
   first_pass = '1';                   /* set first pass               */
 }/* end of if first pass */
 ci_str = &stock_info[0];
 ip->outstart = (char*)output;        /* set the output location      */
 ip->colptr = input;                  /* set the start of the input   */
 ImportFields(ip);                    /* Find import fields           */
 if (ip->tran_error == 'N')
 {
    belowptr = ip->outstart + 17;     /* set the below level pointer  */
    memset ( belowptr, 0, 2);         /* clear the below stock level  */
    return 0;
 }
 else
 {
    if (ip->sign_on_error == 'N' )
    {
       return (80);
    }
    else
       return (100);
 }/* end of else */
}
void check_input_string(ImportParms * ip)/* check the input string     */
{
  if (ip->fldlen == 0)              /* Field length 0?                */
                                    /* Yes, calculate fldlen          */
       ip->fldlen  = ip->colptr-ip->inptr-ip->Blankcnt;
  ip->Blankcnt   = 0;               /* reset blank count              */
  if (ip->fldlen == 0)              /* if input field is empty        */
  {
     if ( ip->fldcnt<2)             /* zero length sign on data       */
     {
        ip->tran_error = 'Y';       /* no empty fileds allowed        */
        ip->sign_on_error = 'Y';    /* report sign on error           */
        ip->ix=ip->buflen;          /* set end condition              */
     }
     else
     {
       switch ( ip->tran_type )
       {
        case 'D':                   /* delivery transactions          */
        case 'S':                   /* stock level tranactions        */
           ip->tran_error = 'Y';    /* no empty fileds allowed        */
        break;
        case 'N':                   /* new order transaction          */
          do_new_order(ip);
        break;                        /* end of case for new order    */
        case 'O':
          do_order_status(ip);
        break;
        case 'P':
          do_payment(ip);
        break;
        default:
        break;
       } /* end of switch */
     } /* end of else sign on error */
  } /* end of if zero string */
  else
  {
      *ip->colptr = '\0';           /* Change delimiter to null       */
      if ( ip->tran_error == 'N')   /* no erors were detected by      */
                                    /* scanner.                       */
      {
        if ( ci->skip=='N')
        MoveField(ip);              /* Go move data to out buffer     */
      }
      else
      {
         if ( ip->fldcnt<2)           /* zero length sign on data     */
         {
             ip->sign_on_error = 'Y';    /* report sign on error      */
         }
         ip->ix=ip->buflen;         /* set end condition              */
      }
  }
}/* end of check input string */
void do_new_order(ImportParms * ip)
{
  int check;
           if( ip->fldcnt < 7)      /* check for first data fields    */
                                    /* must be at lest one item ordered*/
           {
              ip->tran_error = 'Y'; /* clear format error             */
              ip->new_count = 0;    /* no items were ordered          */
              ip->ix=ip->buflen;    /* set end condition              */
           }
           else
           {
              check = (ip->fldcnt - 7 )%3;
              if ( check != 0 )
              {
                 ip->tran_error = 'Y';/* set the error flag           */
                 ip->new_count = 0;   /* no valid items were ordered  */
                 ip->ix=ip->buflen;   /* set end condition            */
              }
              else
              {
                 ip->end_trans = 'Y';
                 memset(ip->outptr,0,ci->fldlen);/* set to zero       */
                 ip->new_count = ((ip->fldcnt - 7 )/3) + 1;
              }                        /* end of else check == 0       */
           }                           /* end of else ip->fldcnt >= 7  */
}
void do_order_status(ImportParms * ip)
{
   if( ip->fldcnt == 3)     /* check for customer number    */
   {
        memset(ip->outptr,0,ci->fldlen);/* set to zero */
        ip->cust_number = 'N';
   }
   else
   {
        if(ip->fldcnt ==4)    /* last name                    */
        {
            memset(ip->outptr,' ', ci->fldlen);/* set  blanks */
            ip->last_name = 'N';
        }
        else
        {
            ip->tran_error = 'Y';/* only blank fields are     */
            ip->ix = ip->buflen;/* set the end for loop flag  */
        }
   }/* end of else fldcnt not equal 3 */
}
void do_payment(ImportParms * ip)
{
   if( ip->fldcnt == 3)/* check for customer number */
   {
         memset(ip->outptr,0,ci->fldlen);/* set to zero */
         ip->cust_number = 'N';
   }
   else
   {
     if (chk_flag == 'B' ) {
        if(ip->fldcnt ==6)         /* last name                        */
        {
            memset(ip->outptr,' ', ci->fldlen);/* set  blanks */
            ip->last_name = 'N';
        }
        else                            /* any other position is an error    */
        {
                  ip->tran_error = 'Y';/* clear format error         */
                  ip->ix = ip->buflen;/* set the end for loop flag   */
        }
       }
       else {
        if(ip->fldcnt ==4)         /* last name                        */
        {
            memset(ip->outptr,' ', ci->fldlen);/* set  blanks */
            ip->last_name = 'N';
        }
        else                            /* any other position is an error    */
        {
                  ip->tran_error = 'Y';/* clear format error         */
                  ip->ix = ip->buflen;/* set the end for loop flag   */
        }
       }
   }                                    /* end of else fldcnt not equal to 3 */
 }/* end of do payment */
```

STKLVLSRV.C:

```
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <atmi.h>       /* TUXEDO Header File */
#include <string.h>
#include <recio.h>
#include <xxfdbk.h>
#include <userlog.h>    /* TUXEDO Header File */
#include <decimal.h>
#include <quscrtus.h>
#include <qusptrus.h>
#include <qusgen.h>
#include <qp0wpid.h>
#include "common.h"

/* tpsvrinit is executed when a server is booted, before it begins
   processing requests.  It is not necessary to have this function.
   Also available is tpsvrdone (not used in this example), which is
   called at server shutdown time.
*/



   static int sd, addr=0, port=8000;
   int rc;


tpsvrinit(int argc, char *argv[])
{
        /* Some compilers warn if argc and argv aren't used. */
        argc = argc;
        argv = argv;

        /* userlog writes to the central TUXEDO message log */
        userlog("Welcome to the tpcc server");
    addr=inet_addr("9.5.149.167");
    sd=Connect(addr,port);
        return(0);
}

/* This function performs the actual service requested by the client.
   Its argument is a structure containing among other things a pointer
   to the data buffer, and the length of the data buffer.
*/



     /*       STOCK LEVEL    */
```

```
STKLVL(TPSVCINFO *rqst)
{


    if(send(sd, rqst->data,230,0)<0) {
      perror("\nsend() failed");
      tpreturn(TPFAIL, 0, rqst->data, 0L, 0);
    }


     if((rc=recv(sd, rqst->data, 670,0))<0) {
      perror("\nrecv[0]() failed");
      tpreturn(TPFAIL, 0, rqst->data, 0L, 0);
    }

    /* Return the transformed buffer to the requestor. */
    tpreturn(TPSUCCESS, 0, rqst->data, rc, 0);
}
```

TUXCONFIG File:

```
5769PW1 V4R2M0  980228                SEU SOURCE LISTING                          05/24/99
10:05:01                  PAGE    1
  SOURCE FILE . . . . . .  TPCCKVCLNT/TUXSRC
  MEMBER  . . . . . .  UBBMSRV
  SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8
...+... 9 ...+... 0
    100 #ident  "@(#)apps:simpapp/ubbsimple     60.3"
    200
    300 #Skeleton UBBCONFIG file for the TUXEDO Simple Application.
    400 #Replace the <bracketed> items with the appropriate values.
    500
    600 *RESOURCES
    700 IPCKEY                  123235
    800
    900 DOMAINID                tpccapp
   1000 MASTER                  HORSEFLY
                        05/04/99
   1100 MAXACCESSERS            6500
   1200 MAXSERVERS              240
   1300 MAXSERVICES             240
   1400 MODEL                   SHM
   1500 LDBAL                   Y
   1600 SCANUNIT                60
   1700 SANITYSCAN              60
   1800 BLOCKTIME               60
   1900 BBLQUERY                60
   2000 DBBLWAIT                1
   2100
   2200 *MACHINES
   2300 "HORSEFLY.RCHLAND.IBM.COM"
                        05/04/99
   2400                         LMID=HORSEFLY
                        05/04/99
   2500                         APPDIR="/home/tpcckvclnt"
   2600                         TUXCONFIG="/home/tpcckvclnt/tuxconfig"
   2700                         TUXDIR="/qopensys/tuxsdk"
   2800 SPINCOUNT=2000
   2900
   3000 *GROUPS
   3100 DEFAULT:                LMID=HORSEFLY
                        05/04/99
   3200 srvr1                   GRPNO=1
   3300 srvr2                   GRPNO=11
   3400 srvr3                   GRPNO=21
   3500 srvr4                   GRPNO=31
   3600 srvr5                   GRPNO=41
   3700 srvr6                   GRPNO=51
   3800 srvr7                   GRPNO=61
   3900 srvr8                   GRPNO=71
   4000 srvr9                   GRPNO=81
   4100 srvr10                  GRPNO=91
   4200
   4300 *SERVERS
   4400 DEFAULT:                REPLYQ=Y
   4500 newordsrv               SRVGRP=srvr1 SRVID=1    MIN=30  RQADDR=noq1
   4600 newordsrv               SRVGRP=srvr6 SRVID=33   MIN=30  RQADDR=noq2
   4700 newordsrv               SRVGRP=srvr8 SRVID=68   MIN=30  RQADDR=noq3
   4800 paymntsrv               SRVGRP=srvr2 SRVID=101  MIN=25  RQADDR=pyq1
   4900 paymntsrv               SRVGRP=srvr7 SRVID=131  MIN=25  RQADDR=pyq2
   5000 paymntsrv               SRVGRP=srvr9 SRVID=161  MIN=25  RQADDR=pyq3
   5100 ordstssrv               SRVGRP=srvr3 SRVID=201  MIN=15  RQADDR=orq1
   5200 delvrysrv               SRVGRP=srvr4 SRVID=301  MIN=15  RQADDR=dlq1
   5300 delvrysrv               SRVGRP=srvr10 SRVID=351  MIN=15  RQADDR=dlq2
  5769PW1 V4R2M0  980228                SEU SOURCE LISTING
05/24/99 10:05:01                  PAGE    2
  SOURCE FILE . . . . . .  TPCCKVCLNT/TUXSRC
  MEMBER  . . . . . .  UBBMSRV
  SEQNBR*...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7 ...+... 8
...+... 9 ...+... 0
   5400 stklvlsrv               SRVGRP=srvr5 SRVID=401  MIN=15  RQADDR=slq1
   5500
   5600 *SERVICES
   5700 DEFAULT:
   5800 NEWORDER
   5900 PAYMNT
   6000 DELIVRY
   6100 STKLVL
   6200 ORDSTS
                               * * * *  E N D  O F  S O U R C E  * * * *
```

SERVER:

DELIVERY.CBL:

```
      PROCESS NOTRUNC NORANGE.
      IDENTIFICATION DIVISION.
        PROGRAM-ID. DELIVERY.
        AUTHOR. PPCOC.
        INSTALLATION. IBM-ROCHESTER.
        DATE-WRITTEN.
        DATE-COMPILED.
      *****************************************************************
      *   C H A N G E S     &     U P D A T E S
      *
      *   LATEST CHANGES LISTED FIRST
      *
*   MM/DD/YY   AUTHOR NAME          LINES CHANGED/ADDED: NNN
*
*
*     DESCRIPTION OF CHANGE:
*
*
*
*****************************************************************
 ENVIRONMENT DIVISION.
 CONFIGURATION SECTION.
   SOURCE-COMPUTER. IBM-AS400.
*  SOURCE-COMPUTER. IBM-AS400 WITH DEBUGGING MODE.
   OBJECT-COMPUTER. IBM-AS400.
 INPUT-OUTPUT SECTION.
 FILE-CONTROL.
     SELECT NEWORD
            ASSIGN TO DATABASE-NEWORDLF
            ORGANIZATION IS INDEXED
            ACCESS MODE IS DYNAMIC
            RECORD KEY IS EXTERNALLY-DESCRIBED-KEY
            WITH DUPLICATES
            FILE STATUS IS NEWORD-FILE-STATUS.
     SELECT DISTRICT-FILE
           ASSIGN TO DATABASE-DSTRCT
           ORGANIZATION IS INDEXED
           ACCESS MODE IS RANDOM
           RECORD KEY IS EXTERNALLY-DESCRIBED-KEY
           WITH DUPLICATES.
     SELECT ORDERS
            ASSIGN TO DATABASE-ORDERSLF
            ORGANIZATION IS INDEXED
            ACCESS MODE IS RANDOM
            RECORD KEY IS EXTERNALLY-DESCRIBED-KEY
            WITH DUPLICATES.
     SELECT ORDERLINE
            ASSIGN TO DATABASE-ORDLINLF
            ORGANIZATION IS INDEXED
            ACCESS MODE IS DYNAMIC
            RECORD KEY IS EXTERNALLY-DESCRIBED-KEY
            WITH DUPLICATES.
     SELECT CSTMR
            ASSIGN TO DATABASE-CSTMR
            ORGANIZATION IS INDEXED
            ACCESS MODE IS RANDOM
            RECORD KEY IS EXTERNALLY-DESCRIBED-KEY
            WITH DUPLICATES.
     SELECT DLVRYLOG
            ASSIGN TO DATABASE-DLVRYLOG
            ORGANIZATION IS SEQUENTIAL.
D      SELECT ISO-IFILE
D           ASSIGN TO DATABASE-ISOIFILE
D           ORGANIZATION IS SEQUENTIAL
D           ACCESS IS SEQUENTIAL.
D      SELECT DBUG-FILE
D           ASSIGN TO FORMATFILE-DBUGPRT2
D           ORGANIZATION IS SEQUENTIAL
D           ACCESS IS SEQUENTIAL.
 I-O-CONTROL.
     COMMITMENT CONTROL FOR ORDERS ORDERLINE
                            DISTRICT-FILE CSTMR NEWORD.

 DATA DIVISION.
 FILE SECTION.
 FD  NEWORD
     LABEL RECORDS ARE STANDARD
     DATA RECORD IS NEWORD-RECORD.
 01  NEWORD-RECORD.
     COPY DDS-NORCD OF NEWORDLF.
 FD DISTRICT-FILE
      LABEL RECORDS ARE STANDARD.
 01 DISTRICT-RECORD.
      COPY DDS-DSRCD OF DSTRCT.
 FD  ORDERS
     LABEL RECORDS ARE STANDARD
     DATA RECORD IS ORDERS-RECORD.
 01  ORDERS-RECORD.
     COPY DDS-ORRCD OF ORDERSLF.
 FD  ORDERLINE
     LABEL RECORDS ARE STANDARD.
 01  ORDERLINE-RECORD.
     COPY DDS-OLRCD OF ORDLINLF.
 FD  CSTMR
     LABEL RECORDS ARE STANDARD
     DATA RECORD IS CSTMR-RECORD.
 01  CSTMR-RECORD.
     COPY DDS-CSRCD OF CSTMR.
 FD  DLVRYLOG
     LABEL RECORDS ARE STANDARD
     DATA RECORD IS DLVRYLOG-RECORD.
 01  DLVRYLOG-RECORD.
     COPY DDS-LGRCD OF DLVRYLOG.
     05 LOG-REDEF REDEFINES LGRCD.
         06        TIMEB    PIC S9(8).
         06        TIMEA    PIC S9(8).
         06        UNDEL    PIC S9.
         06        WID      PIC S9(4) COMP-4.

DFD  DBUG-FILE
D    LABEL RECORDS ARE STANDARD
D    DATA RECORD IS DBUG-REC2.
D   01  DBUGPRT-REC2.
D    COPY DDS-DBUGRCD2 OF DBUGPRT2.
D       05  DBUG-REC2 REDEFINES DBUGRCD2-O.
D           06  DEBUG-DATA.
D               07  ISONUMBER              PIC 99.
D               07  TXTDATA                PIC X(40).
D               07  WID                    PIC S9(4).
D               07  DID                    PIC S9(2).
D               07  TOD                    PIC X(8).
D               07  TIME6                  PIC 9(6).
D               07  CID                    PIC S9(4).
D               07  CWID                   PIC S9(4).
D               07  CDID                   PIC S9(2).
D               07  CBAL                   PIC S9(11)V99.
D               07  PAYMNT                 PIC S9(7)V99.
DFD  ISO-IFILE
D    LABEL RECORDS ARE STANDARD.
D01  ISOIFILE-RECORD.
D    COPY DDS-ISOIRCD OF ISOIFILE.
D    05 ISO-IREC REDEFINES ISOIRCD.
D      06 ISOINUM PIC S9(4) COMP-4.


 WORKING-STORAGE SECTION.

 01 CSTMR-KEY.
   03 CSTMR-KEY-DATA OCCURS 5 TIMES.
      05  KNAM   PIC X(10).
      05  KVAL   PIC S9(9) COMP-4.

 01 CSTMR-HASH-PTR   USAGE POINTER.

 01 CSTMR-HASH-NAME  PIC X(10).
```

```
 01 FUNCT            PIC X(1).
 01 KEYS             PIC S9(9) COMP-4.
 01 RET-CODE         PIC S9(9) COMP-4.

 01  RET-CODE-PTR    USAGE POINTER.
 01  CSTMR-DEF-PTR   USAGE POINTER.
 01  CSTMR-KEY-PTR   USAGE POINTER.

D01  CBAL2              PIC S9(11)V99 COMP-3.
D01  ISOVALUE           PIC S999 COMP-3 VALUE 0.

 01 UPDATER              PIC X(1) VALUE "3".
 01 FETCHR               PIC X(1) VALUE "1".
 01 FETCHUPDATE          PIC X(1) VALUE "2".

 01 CUST-KEYS            pic 9(1) comp-4 vALUE 3.
 01 STOCK-KEYS           pic 9(1) comp-4 VALUE 2.
 01 ITEM-KEYS            pic 9(1) comp-4 VALUE 1.



  01   TRANSACTION-INPUT.
         06    TXN-TYPE          PIC X.
         06    JOBNAME           PIC X(10).
         06    CLIENT-INPUT      PIC X(202).
         06 DLVRY-I REDEFINES CLIENT-INPUT.
           08 WID               PIC S9(4) COMP-4.
           08 CARRIER           PIC XX.
           08 D-TIME-OF-DAY     PIC X(8).
*          08 D-DATE            PIC 9(6).

    01  SWITCH-AREA.
        05  SW03                          PIC 1.
          88 LOCK-OCCURRED              VALUE B"1".
          88 LOCK-OFF                   VALUE B"0".

 77  STATUS-IND             PIC X(2).
 77  MAJ-MIN-SAV            PIC X(4).
 77  INDON                  PIC 1 VALUE B"1".
 77  INDOFF                 PIC 1 VALUE B"0".
 77  LEN                    PIC 9(10)V9(5) COMP.

 01 CMNF-INDIC-AREA.
    05  CMNF-INDIC          PIC 1 OCCURS 99 TIMES
                            INDICATOR 1.
 01 MAJ-MIN.
    05 MAJ                 PIC X(2).
    05 MIN                 PIC X(2).

*01  TPCDATE.
*   05 TPCDATEDD        PIC  99.
*   05 TPCDATEMM        PIC  99.
*   05 TPCDATECC        PIC  99.
*   05 TPCDATEYY        PIC  99.
*01  CURTIME.
*   05 CURTIME6        PIC 9(6).
*   05 FILLER          PIC 9(2).
 01  TIME-OF-DAY       PIC X(8).
 01  TIME-PTR          USAGE POINTER.
*   MI Time of day from call to mattod
*   05 DATEIYY          PIC  99.
*   05 DATEIMM          PIC  99.
*   05 DATEIDD          PIC  99.

 01 DQDATA-FRM-CLNT.
    05 DQDATA-WHS  PIC S9(4) COMP-4.
    05 DQDATA-CARR PIC X(2).
    05 DQDATA-BTIM PIC S9(8).
    05 DQDATA-fill PIC S9(6).

D  01 TESTNUM                 PIC 99 VALUE 1.

 01 OLINES-LEFT  PIC S9(2) COMP-4.

 77  ORDAMT PIC 9(9)V99.

 01  FILE-STATUS-ERROR-CDS.
    05  NEWORD-FILE-STATUS    PIC X(2).
 01  ORDERS-DELIVERED-TABLE.
       10 ORDER-LINE OCCURS 10 TIMES.
         15 ELEMENT-DISTRICT     PIC S99.
         15 ELEMENT-ORDER        PIC S9(6).
 77  WHS   PIC X(4).

 LINKAGE SECTION.
   01  FIRST-TIME PIC S9(4) COMP-4.
   01  INP-TRAN PIC X(230).

 PROCEDURE DIVISION USING FIRST-TIME INP-TRAN.

 DELIVERY-PROGRAM.
     IF FIRST-TIME = 001 THEN
         OPEN I-O NEWORD
         OPEN I-O DISTRICT-FILE
         OPEN I-O ORDERS
         OPEN I-O CSTMR
         OPEN I-O ORDERLINE
         OPEN EXTEND DLVRYLOG
D        OPEN OUTPUT DBUG-FILE
D        OPEN INPUT ISO-IFILE

         MOVE "CSTMR"  TO CSTMR-HASH-NAME
         MOVE "CID"    TO KNAM OF CSTMR-KEY-DATA(1)
         MOVE "CDID"   TO KNAM OF CSTMR-KEY-DATA(2)
         MOVE "CWID"   TO KNAM OF CSTMR-KEY-DATA(3)

         SET CSTMR-HASH-PTR TO ADDRESS OF CSTMR-HASH-NAME
         SET RET-CODE-PTR   TO ADDRESS OF RET-CODE
         SET CSTMR-DEF-PTR  TO ADDRESS OF CSTMR-RECORD
         SET CSTMR-KEY-PTR  TO ADDRESS OF CSTMR-KEY
         SET TIME-PTR       TO ADDRESS OF TIME-OF-DAY
     END-IF.
 PROCESS-DELIVERY.

     MOVE INP-TRAN TO  TRANSACTION-INPUT.

     MOVE DLVRY-I TO DQDATA-FRM-CLNT.

     MOVE D-TIME-OF-DAY OF DLVRY-I TO
                     LOGTIMEB OF DLVRYLOG-RECORD.

D    READ ISO-IFILE.
D    MOVE ISOINUM TO ISONUMBER, ISOVALUE.
     MOVE   0 TO NODID.
     MOVE   0 TO NOOID.
     MOVE "0" TO UNDELDST.
     MOVE DQDATA-WHS TO NOWID, OWID, OLWID, CWID OF CSTMR-RECORD.
     MOVE DQDATA-WHS TO DWID.

     IF DQDATA-CARR = "99" THEN
       STOP RUN.

     MOVE DQDATA-CARR TO OCARID.

*    ACCEPT DATEINT     FROM DATE.
     CALL "gettime" USING BY VALUE TIME-PTR.
*    Move "19" to TPCDatecc.
*    If dateiyy < "70" then
*      Move "20" to TPCDatecc.
*    Move   DATEIYY      TO TPCDATEYY.
*    Move   DATEIMM      TO TPCDATEMM.
*    Move   DATEIDD    TO TPCDATEDD.

*    ACCEPT CURTIME FROM TIME.

* THE FOLLOWING CODE WILL ONLY BE INCLUDED WHEN THE
* FLAG IS SET AND IS FOR ISOLATION TEST USE
*
D    MOVE "DEL PRE-READ  " TO TXTDATA.
D    MOVE 1 TO CDID OF DBUG-REC2.
D    MOVE DQDATA-WHS TO CWID OF DBUG-REC2.
D    MOVE 0 TO CID OF DBUG-REC2.
D    MOVE 1 TO DID OF DBUG-REC2.
D    MOVE 0 TO PAYMNT OF DBUG-REC2.
D    MOVE DQDATA-WHS TO WID OF DBUG-REC2.
D    MOVE 0 TO CBAL OF DBUG-REC2.
*    MOVE CURTIME6 TO TIME6.
D    MOVE TIME-OF-DAY TO TOD.
D    MOVE TPCDATE TO DATE6.
D    WRITE DBUGPRT-REC2 FORMAT IS "DBUGRCD2".
  RE-DO-DLVRY.
     PERFORM 10 TIMES

       MOVE 0 TO ORDAMT
       ADD 1 TO NODID
       READ NEWORD
       IF NEWORD-FILE-STATUS = "23"
          MOVE NODID TO DID OF DISTRICT-RECORD
          READ DISTRICT-FILE
          READ NEWORD
          END-IF
D      IF NEWORD-FILE-STATUS = "23"
*         ACCEPT CURTIME FROM TIME
D         CALL "gettime" USING BY VALUE TIME-PTR
*         MOVE CURTIME6 TO TIME6
D         MOVE TIME-OF-DAY TO TOD
D         MOVE TPCDATE TO DATE6
D         MOVE NODID TO DID OF DBUG-REC2
D         MOVE "DEL NO NEWORDER" TO TXTDATA
D         WRITE DBUGPRT-REC2 FORMAT IS "DBUGRCD2"
D         CALL "delayme"
D         READ NEWORD
*         ACCEPT CURTIME FROM TIME
D         CALL "gettime" USING BY VALUE TIME-PTR
*         MOVE CURTIME6 TO TIME6
D         MOVE TIME-OF-DAY TO TOD
D         MOVE TPCDATE TO DATE6
D         MOVE "DEL NO NEWORD2" TO TXTDATA
D         WRITE DBUGPRT-REC2 FORMAT IS "DBUGRCD2"
D         END-IF
          IF NEWORD-FILE-STATUS = "23"
            MOVE "1" TO UNDELDST
          ELSE
            MOVE NODID TO ODID, OLDID, CDID OF CSTMR-RECORD
            MOVE NOOID TO OID, OLOID

            READ ORDERS
            MOVE DQDATA-CARR TO OCARID
            REWRITE ORDERS-RECORD

            READ ORDERLINE
*           MOVE TPCDATE TO OLDLVD
*           MOVE CURTIME6 TO OLDLVT
            MOVE TIME-OF-DAY TO OLDLVTOD
            REWRITE ORDERLINE-RECORD
            ADD OLAMNT TO ORDAMT

            SUBTRACT 1 FROM OLINES GIVING OLINES-LEFT

            PERFORM OLINES-LEFT TIMES
                 READ ORDERLINE NEXT
*                MOVE TPCDATE TO OLDLVD
*                MOVE CURTIME6 TO OLDLVT
                 MOVE TIME-OF-DAY TO OLDLVTOD
                 REWRITE ORDERLINE-RECORD
                 ADD OLAMNT TO ORDAMT
            END-PERFORM


          MOVE OCID TO CID OF CSTMR-RECORD
          MOVE CID  OF CSTMR-RECORD TO KVAL OF CSTMR-KEY-DATA(1)
          MOVE CDID OF CSTMR-RECORD TO KVAL OF CSTMR-KEY-DATA(2)
          MOVE CWID OF CSTMR-RECORD TO KVAL OF CSTMR-KEY-DATA(3)

            CALL "qdbrunha" USING BY VALUE
            CSTMR-HASH-PTR FETCHUPDATE CUST-KEYS CSTMR-KEY-PTR
                     CSTMR-DEF-PTR RET-CODE-PTR
               IF RET-CODE = 812
                   SET LOCK-OFF TO TRUE
                   ROLLBACK
                   GO TO RE-DO-DLVRY
               END-IF
            ADD ORDAMT TO CBAL OF CSTMR-RECORD
            ADD 1 TO CDELCNT

            CALL "qdbrunha" USING BY VALUE
                 CSTMR-HASH-PTR UPDATER CUST-KEYS CSTMR-KEY-PTR
                      CSTMR-DEF-PTR RET-CODE-PTR

               IF RET-CODE = 812
                   SET LOCK-OFF TO TRUE
                   ROLLBACK
                   GO TO RE-DO-DLVRY
               END-IF

            DELETE NEWORD
            MOVE OID  OF ORDERS-RECORD
                     TO ELEMENT-ORDER(NODID)

         END-IF
       END-PERFORM.

* THE FOLLOWING CODE WILL ONLY BE INCLUDED WHEN THE
* FLAG IS SET AND IS FOR ISOLATION TEST USE
*
D    IF ISOVALUE = 8 THEN GO TO DBG-SKP1.
D    IF ISOVALUE = 6 THEN
D       MOVE "DEL PRE-ROLLBK" TO TXTDATA
D    ELSE IF ISOVALUE = 5 THEN
D       MOVE "DEL PRE-COMMIT" TO TXTDATA.
D    MOVE 1 TO CDID OF DBUG-REC2.
D    MOVE DQDATA-WHS TO CWID OF DBUG-REC2.
D    MOVE 1 TO DID OF DBUG-REC2.
D    MOVE DQDATA-WHS TO WID OF DBUG-REC2.
*    MOVE CURTIME6 TO TIME6.
*    MOVE TPCDATE TO DATE6.
D    MOVE TIME-OF-DAY TO TOD.
```

```
      D    WRITE DBUGPRT-REC2 FORMAT IS "DBUGRCD2".
      D    CALL "delayme".
      D    IF ISOVALUE = 6 THEN
      D     ROLLBACK.
      DDBG-SKP1.
                COMMIT.

      * THE FOLLOWING CODE WILL ONLY BE INCLUDED WHEN THE
      * FLAG IS SET AND IS FOR ISOLATION TEST USE
      *
      D    IF ISOVALUE = 8 THEN GO TO DBG-SKP2.
      D    IF ISOVALUE = 6 THEN
      D       MOVE "DEL POST-ROLLBK" TO TXTDATA
      D    ELSE IF ISOVALUE = 5 THEN
      D       MOVE "DEL POST-COMMIT" TO TXTDATA.
      D    MOVE 1 TO CDID OF DBUG-REC2.
      D    MOVE DQDATA-WHS TO CWID OF DBUG-REC2.
      D    MOVE 1 TO DID OF DBUG-REC2.
      D    MOVE 0 TO PAYMNT OF DBUG-REC2.
      D    MOVE DQDATA-WHS TO WID OF DBUG-REC2.
      D    MOVE CBAL2 TO CBAL OF DBUG-REC2.
      *    ACCEPT DATEINT     FROM DATE.
      D    CALL "gettime" USING BY VALUE TIME-PTR.
      *    Move "19" to TPCDatecc.
      *    If dateiyy < "70" then
      *      Move "20" to TPCDatecc.
      *    Move   DATEIYY      TO TPCDateYY.
      *    Move   DATEIMM      TO TPCDateMM.
      *    Move   DATEIDD      TO TPCDateDD.
      *    ACCEPT CURTIME FROM TIME.
      *    MOVE CURTIME6 TO TIME6.
      *    MOVE TPCDATE TO DATE6.
      D    MOVE TIME-OF-DAY TO TOD.
      D    WRITE DBUGPRT-REC2 FORMAT IS "DBUGRCD2".
      DDBG-SKP2.
           MOVE DQDATA-WHS TO WH OF DLVRYLOG-RECORD.
      *    ACCEPT LOGTIMEA FROM TIME.
           CALL "gettime" USING BY VALUE TIME-PTR.
           MOVE TIME-OF-DAY TO LOGTIMEA.
           WRITE DLVRYLOG-RECORD.

           GOBACK.

           GO TO PROCESS-DELIVERY.
```

DLVFEND.CL:

```
/* DLVFEND:  Delivery Job Start Program         */
             PGM

             DCL        VAR(&DATALIB) TYPE(*CHAR) LEN(10)
             DCL        VAR(&OBJLIB) TYPE(*CHAR) LEN(10)
             MONMSG CPF0000
             RTVDTAARA  DTAARA(TPCCINFO/DATALIB) RTNVAR(&DATALIB)
             RTVDTAARA  DTAARA(TPCCINFO/OBJECTLIB) RTNVAR(&OBJLIB)
             ADDLIBLE   LIB(&DATALIB)
             ADDLIBLE   LIB(&OBJLIB)
             MONMSG CPF2103
             CHGJOB     RUNPTY(34)
             STRCMTCTL  LCKLVL(*ALL)
             RETURN
```

DLVRSRVR.C:

```
#include "Common.h"
char *DELIVERY(short *, char *);

main(  ) {
  char          rcv_buf[BufferSize],  snd_buf[BufferSize];
  int           sd, snd_len=670, rcv_len=230, rc;
  static short  first_time=1;


      for(;;) {

      /* recv() client's request */
      if((rc=recv(sd,rcv_buf,rcv_len,0)) < 0) {
        perror("\nrecv() failed");
        exit(-1);
        }
      else {            /* else 2  */
        if(rc==0)/* if client was closed shut down operation  */
        {
          close(sd);
          exit(0);
        }
        rcv_buf[rcv_len]='\0';
          if(first_time==1) {
                          DLVFEND();
             system(
   "OVRDBF FILE(ORDERSLF) TOFILE(ORDERSLF) LVLCHK(*NO) SHARE(*NO)");

             system(
   "OVRDBF FILE(NEWORDLF) TOFILE(NEWORDLF) LVLCHK(*NO) SHARE(*NO) NBRRCDS(10)");

             system(
   "OVRDBF FILE(CSTMR) TOFILE(CSTMR) WAITRCD(5) LVLCHK(*NO) SHARE(*NO)");

             system(
   "OVRDBF FILE(CSTMRPF) TOFILE(CSTMRPF) WAITRCD(5) LVLCHK(*NO) SHARE(*NO)");

             system(
      "OVRDBF FILE(DLVRYLOG) TOFILE(DLVRYLOG) LVLCHK(*NO) SHARE(*NO)");

             system(
"OVRDBF FILE(ORDLINLF) TOFILE(ORDLINLF) LVLCHK(*NO) SHARE(*NO) \
 SEQONLY(*NO) NBRRCDS(120)");

                          DELIVERY(&first_time, rcv_buf);
                          first_time = 0;
                        }
          else DELIVERY(&first_time, rcv_buf);
          }
   }  /* for loop  */
  exit(0);
}
```

NOPAYOSMOD.CBL:

```
       PROCESS NOTRUNC NORANGE.

 IDENTIFICATION DIVISION.
   PROGRAM-ID. NOPAYOSMOD.
   AUTHOR.  P/P COC
   INSTALLATION. IBM-ROCHESTER.
   DATE-WRITTEN.
   DATE-COMPILED.
*                                                                *
*   C  H  A  N  G  E  S     &      U  P  D  A  T  E  S
*
*   LATEST CHANGES LISTED FIRST
*
*   MM/DD/YY   AUTHOR NAME           LINES CHANGED/ADDED: NNN
*
*
*
*                                                                *
 ENVIRONMENT DIVISION.
 CONFIGURATION SECTION.
   SOURCE-COMPUTER. IBM-AS400.
*  SOURCE-COMPUTER. IBM-AS400 WITH DEBUGGING MODE.
   OBJECT-COMPUTER. IBM-AS400.
   INPUT-OUTPUT SECTION.
   FILE-CONTROL.
       SELECT WAREHOUSE-FILE ASSIGN TO DATABASE-WRHS
            ORGANIZATION IS INDEXED
            ACCESS MODE IS RANDOM
            RECORD KEY IS EXTERNALLY-DESCRIBED-KEY
            WITH DUPLICATES.
       SELECT DISTRICT-FILE ASSIGN TO DATABASE-DSTRCT
            ORGANIZATION IS INDEXED
            ACCESS MODE IS RANDOM
            RECORD KEY IS EXTERNALLY-DESCRIBED-KEY
            WITH DUPLICATES
            FILE STATUS IS DISTRICT-FILE-STATUS.
       SELECT CUSTOMER-FILE ASSIGN TO DATABASE-CSTMR
            ORGANIZATION IS INDEXED
            ACCESS MODE IS RANDOM
            RECORD KEY IS EXTERNALLY-DESCRIBED-KEY
            WITH DUPLICATES
            FILE STATUS IS CUSTOMER-FILE-STATUS.
       SELECT ITEM-FILE ASSIGN TO DATABASE-ITEM
            ORGANIZATION IS INDEXED
            ACCESS MODE IS RANDOM
            RECORD KEY IS EXTERNALLY-DESCRIBED-KEY
            WITH DUPLICATES
            FILE STATUS IS ITEM-FILE-STATUS.
       SELECT STOCK-FILE ASSIGN TO DATABASE-STOCK
            ORGANIZATION IS INDEXED
            ACCESS MODE IS RANDOM
            RECORD KEY IS EXTERNALLY-DESCRIBED-KEY
            WITH DUPLICATES
            FILE STATUS IS STOCK-FILE-STATUS.
       SELECT NEWORDER-FILE ASSIGN TO DATABASE-NEWORD
            ORGANIZATION IS SEQUENTIAL
            FILE STATUS IS NEWORDER-FILE-STATUS.
       SELECT ORDERS-FILE ASSIGN TO DATABASE-ORDERSLF
            ORGANIZATION IS SEQUENTIAL.
       SELECT ORDERS-VIEW ASSIGN TO DATABASE-ORDERS
            ORGANIZATION IS INDEXED
            ACCESS MODE IS DYNAMIC
            RECORD KEY IS EXTERNALLY-DESCRIBED-KEY
            WITH DUPLICATES.
       SELECT ORDERLINE-FILE ASSIGN TO DATABASE-ORDLIN
            ACCESS MODE IS SEQUENTIAL.
       SELECT ORDERLINE-VIEW ASSIGN TO DATABASE-ORDLINLF
            ORGANIZATION IS INDEXED
            ACCESS MODE IS DYNAMIC
            RECORD KEY IS EXTERNALLY-DESCRIBED-KEY
            WITH DUPLICATES.
       SELECT HISTORY-FILE ASSIGN TO DATABASE-HSTRY
            ORGANIZATION IS SEQUENTIAL.
D      SELECT DBUG-FILE
D           ASSIGN TO FORMATFILE-DBUGPRT
D           ORGANIZATION IS SEQUENTIAL
D           ACCESS IS SEQUENTIAL.
D      SELECT DBUG-FILE2
D           ASSIGN TO FORMATFILE-DBUGPRT2
D           ORGANIZATION IS SEQUENTIAL
D           ACCESS IS SEQUENTIAL.
D      SELECT ISO-IFILE
D           ASSIGN TO DATABASE-ISOIFILE
D           ORGANIZATION IS SEQUENTIAL
D           ACCESS IS SEQUENTIAL.
       SELECT CUST-REL-FILE ASSIGN TO DATABASE-CSTMRPF
            ORGANIZATION IS RELATIVE
            ACCESS MODE IS RANDOM RELATIVE KEY IS CUSTREL
            FILE STATUS IS CUSTOMER-FILE-STATUS.

   I-O-CONTROL.
      COMMITMENT CONTROL FOR WAREHOUSE-FILE
      COMMITMENT CONTROL FOR DISTRICT-FILE
      COMMITMENT CONTROL FOR CUSTOMER-FILE
      COMMITMENT CONTROL FOR CUST-REL-FILE
      COMMITMENT CONTROL FOR STOCK-FILE
      COMMITMENT CONTROL FOR ITEM-FILE
      COMMITMENT CONTROL FOR NEWORDER-FILE
      COMMITMENT CONTROL FOR ORDERS-FILE
      COMMITMENT CONTROL FOR ORDERS-VIEW
      COMMITMENT CONTROL FOR ORDERLINE-FILE
      COMMITMENT CONTROL FOR ORDERLINE-VIEW
      COMMITMENT CONTROL FOR HISTORY-FILE.
   DATA DIVISION.
   FILE SECTION.

   FD WAREHOUSE-FILE
      LABEL RECORDS ARE STANDARD.
    01 WAREHOUSE-REC.
         COPY DDS-WRRCD OF WRHS.
         05 WRHS-INFO REDEFINES WRRCD.
            06 FILLER                PIC X(12).
            06 WH-ADDRESS.
               08 WADDR1             PIC X(20).
               08 WADDR2             PIC X(20).
               08 CITYW              PIC X(20).
               08 STATEW             PIC X(2).
               08 ZIPW               PIC X(10).
   FD DISTRICT-FILE
      LABEL RECORDS ARE STANDARD.
    01 DISTRICT-REC.
      COPY DDS-DSRCD OF DSTRCT.
       05  DISTRICT-INFO REDEFINES DSRCD.
            06 DIST-KEY.
               08 DID              PIC S9(4) COMP-4.
               08 DWID             PIC S9(4) COMP-4.
            06 FILLER             PIC X(10).
            06 DIST-ADDRESS.
               08 DADDR1             PIC X(20).
               08 DADDR2             PIC X(20).
               08 CITY               PIC X(20).
               08 STATE              PIC X(2).
               08 ZIP                PIC X(10).
   FD  CUSTOMER-FILE
       LABEL RECORDS ARE STANDARD.
    01  CUSTOMER-REC.
```

```
          COPY DDS-CSRCD OF CSTMR.
          05  CUST-KEY REDEFINES CSRCD.
              06 CIDK      PIC S9(9) COMP-4.
              06 CDIDK     PIC S9(4) COMP-4.
              06 CWIDK     PIC S9(4) COMP-4.
          05  CUST-INFO-REC REDEFINES CSRCD.
              06 FILLER    PIC X(8).
              06 CUST-INFO.
                 08 CFIRSTP  PIC X(16).
                 08 CINITP   PIC X(02).
                 08 CLASTP   PIC X(16).
                 08 CLDATEP  PIC S9(8).
                 08 CADDR1P  PIC X(20).
                 08 CCREDTP  PIC X(02).
                 08 CADDR2P  PIC X(20).
                 08 CDCTP    PIC S9(1)V9(4) COMP-3.
                 08 CCITYP   PIC X(20).
                 08 CSTATEP  PIC X(02).
                 08 CZIPP    PIC X(10).
                 08 CPHONEP  PIC X(16).
                 08 CBAL     PIC S9(11)V99 COMP-3.
                 08 CCRDLM   PIC S9(11)V99 COMP-3.

    FD  CUST-REL-FILE
        LABEL RECORDS ARE STANDARD.
     01  CUST-REL-REC.
         COPY DDS-CSRCD OF CSTMRPF.
    FD ITEM-FILE
       LABEL RECORDS ARE STANDARD.
     01 ITEM-REC.
       COPY DDS-ITRCD OF ITEM.
       05 ITEM-DEF REDEFINES ITRCD.
           06 ID-IID            PIC S9(9) COMP-4.
           06 ID-IMID           PIC S9(9) COMP-4.
           06 ITEM-INFO.
              08 ID-INAME       PIC X(24).
              08 ID-IPRICE      PIC S9(3)V99 COMP-3.
           06 ID-IDATA          PIC X(50).
    FD STOCK-FILE
       LABEL RECORDS ARE STANDARD.
     01 STOCK-REC.
       COPY DDS-STRCD OF STOCK.
       05  STOCK-DEF REDEFINES STRCD.
           06  STOCK-IN.
               08 STWID1          PIC S9(4) COMP-4.
               08 STIID1          PIC S9(9) COMP-4.
           06 FILLER              PIC S9(4) COMP-4.
           06  DIST-INFO  OCCURS 10 TIMES.
               07  DIST-IN        PIC X(24).
    FD NEWORDER-FILE
       LABEL RECORDS ARE STANDARD.
     01 NEWORDER-REC.
       COPY DDS-NORCD OF NEWORD.
    FD ORDERS-FILE
       LABEL RECORDS ARE STANDARD.
     01 ORDERS-REC.
       COPY DDS-ORRCD OF ORDERSLF.
    FD ORDERS-VIEW
       LABEL RECORDS ARE STANDARD.
     01 ORDERS-VIEW-REC.
       COPY DDS-ORRCD OF ORDERS.
    FD ORDERLINE-FILE
       LABEL RECORDS ARE STANDARD.
     01 ORDERLINE-REC.
       COPY DDS-OLRCD OF ORDLIN.
       05 ORDERLIN-INPUT REDEFINES OLRCD.
          07 ORDER-KEY.
             08 OLOID                PIC S9(9) COMP-4.
             08 OLDID                PIC S9(4) COMP-4.
             08 OLWID                PIC S9(4) COMP-4.
             08 OLNBR                PIC S9(4) COMP-4.
          07 OL-INPUT.
             08 OL-INPUT-OLSPWH      PIC S9(4) COMP-4.
             08 OL-INPUT-OLIID       PIC S9(9) COMP-4.
             08 OL-INPUT-QTY         PIC S9(3) COMP-3.
          07 FILLER                  PIC X(4).
*         07 OLDATE-TIME             PIC X(14).
          07 OLTIME-OF-DAY           PIC X(8).
          07 TIME-FILLER             PIC X(6).
    FD ORDERLINE-VIEW
       LABEL RECORDS ARE STANDARD.
     01 ORDERLINE-VIEW-REC.
       COPY DDS-OLRCD OF ORDLINLF.
       05 OL-STATUS-OUTPUT REDEFINES OLRCD.
          08 FILLER               PIC X(10).
          08 OL-INFO.
             10 OL-INFO-OLSPWH       PIC S9(4) COMP-4.
             10 OL-INFO-OLIID        PIC S9(9) COMP-4.
             10 OL-INFO-OLQTY        PIC S9(3) COMP-3.
             10 OL-INFO-OLAMNT       PIC S9(5)V9(2) COMP-3.
*            10 OL-INFO-OLDLVD       PIC S9(8).
             10 OL-INFO-OLDLVTOD     PIC X(8).
 FD  HISTORY-FILE
     LABEL RECORDS ARE STANDARD
     DATA RECORD IS HISTORY-REC.
    01  HISTORY-REC.
     COPY DDS-HSRCD OF HSTRY.
      05 HIST-INFO REDEFINES HSRCD.
         07 HISTORY-DATA.
            09 PAY-DIST-KEY.
             10 DID              PIC S9(4) COMP-4.
             10 WID              PIC S9(4) COMP-4.
            09 PAY-CUST-KEY.
             10 CID              PIC S9(9) COMP-4.
             10 CDID             PIC S9(4) COMP-4.
             10 CWID             PIC S9(4) COMP-4.
         07 DATE-TIME.
            09 HTOD              PIC X(8).
*           09 HTIME             PIC 9(6).
         07 AMOUNT             PIC S9(5)V99 COMP-3.
         07 HDATA.
           09 WNAME              PIC X(10).
           09 FILLER             PIC XXXX.
           09 DNAME              PIC X(10).
DFD  DBUG-FILE
D    LABEL RECORDS ARE STANDARD
D    DATA RECORD IS DBUG-REC.
D   01  DBUGPRT-REC.
D    COPY DDS-DBUGRCD OF DBUGPRT.
D       05  DBUG-REC REDEFINES DBUGRCD-O.
D           06  DEBUG-DATA.
D               07  ISONUM                 PIC XX.
D               07  TXTDATA                PIC X(40).
D               07  WID                    PIC S9(4).
D               07  DID                    PIC S9(4).
D               07  DATE6                  PIC S9(8).
D               07  TIME6                  PIC S9(6).
D               07  CID                    PIC S9(4).
D               07  OID                    PIC S9(9).
D               07  DEBUG-ITEM-INFO OCCURS 15 TIMES.
D                 08 DEBUG-OLSPWH          PIC S9(4).
D                 08 DEBUG-OLIID           PIC S9(6).
D                 08 DEBUG-OLQTY           PIC S9(3).
D                 08 DEBUG-IPRICE          PIC S9(5)V9(2).
DFD  DBUG-FILE2
D    LABEL RECORDS ARE STANDARD
D    DATA RECORD IS DBUG-REC2.
D   01  DBUGPRT-REC2.
D    COPY DDS-DBUGRCD2 OF DBUGPRT2.
D       05  DBUG-REC2 REDEFINES DBUGRCD2-O.
D           06  DEBUG-DATA.
D               07  ISONUM                 PIC XX.
D               07  TXTDATA                PIC X(40).
D               07  WID                    PIC S9(4).
D               07  DID                    PIC S9(4).
D               07  DATE6                  PIC S9(8).
D               07  TIME6                  PIC S9(6).
D               07  CID                    PIC S9(4).
D               07  CWID                   PIC S9(4).
D               07  CDID                   PIC S9(4).
D               07  CBAL                   PIC S9(11)V99.
D               07  PAYMNT                 PIC S9(7)V99.
DFD  ISO-IFILE
D    LABEL RECORDS ARE STANDARD.
D01  ISOIFILE-RECORD.
D    COPY DDS-ISOIRCD OF ISOIFILE.
D    05 ISO-IREC REDEFINES ISOIRCD.
D      06 ISOINUM PIC S9(4) COMP-4.


  WORKING-STORAGE SECTION.

   01  TRAN-OUTPUT.
    05 TRAN-OUT PIC X(670).

    05 NEWORD-O REDEFINES TRAN-OUT.
         06    NEWORD-RESULT         PIC X.
         06    OENTTM                PIC S9(6).
         06    CLAST                 PIC X(16).
         06    CCREDT                PIC X(2).
         06    CDCT                  PIC S999V99 COMP-3.
         06    OID                   PIC S9(8) COMP-4.
         06    WTAX                  PIC S9V9999 COMP-3.
         06    DTAX                  PIC S9V9999 COMP-3.
         06    TOTAMT                PIC S9(9)V9(2) COMP-3.
         06  OUTPUT-TABLE.
           10 OUTPUT-LINE OCCURS  16 TIMES.
              15 OUTPUT-STQTY        PIC S9(4) COMP-4.
              15 OUTPUT-BORG         PIC  X(1).
              15 OUTPUT-ITEM-INFO.
                20 OUTPUT-INAME        PIC X(24).
                20 OUTPUT-IPRICE       PIC S9(3)V9(2) COMP-3.
              15 OUTPUT-OLAMNT       PIC S9(5)V9(2) COMP-3.

      05 PAYMNT-O REDEFINES TRAN-OUT.
         06 OUTPUT-FMT-NUM-3      PIC XX.
*                                   17 = bad credit
*                                   18 = no cid or clast
*                                   19 = good credit
*                                   88 = bad last name
         06 HDT.
           08 HDATE    PIC 9(8).
           08 HTIME    pic 9(6).
         06 WH-DATA.
           08 WADDR1   PIC X(20).
           08 WADDR2   PIC X(20).
           08 WCITY    PIC X(20).
           08 WSTATE   PIC X(02).
           08 WZIP     PIC X(9).
         06 DST-DATA.
           08 DADDR1   PIC X(20).
           08 DADDR2   PIC X(20).
           08 DCITY    PIC X(20).
           08 DSTATE   PIC X(02).
           08 DZIP     PIC X(9).
         06 CID        PIC S9(6) COMP-4.
         06 PAYMENT-CUST-INFO.
           08 CFIRST   PIC X(16).
           08 CINIT    PIC X(02).
           08 CLAST    PIC X(16).
           08 CLDATE   PIC S9(8).
           08 CADDR1   PIC X(20).
           08 CCREDT   PIC X(02).
           08 CADDR2   PIC X(20).
           08 CDCT     PIC S999V99 COMP-3.
           08 CCITY    PIC X(20).
           08 CSTATE   PIC X(02).
           08 CZIP     PIC X(9).
           08 CPHONE   PIC X(16).
           08 CBAL     PIC S9(11)V9(2) COMP-3.
           08 CCRDLM   PIC S9(11)V9(2) COMP-3.
         06 MISC-CDATA.
           08  CDAT1    PIC X(50).
           08  CDAT2    PIC X(50).
           08  CDAT3    PIC X(50).
           08  CDAT4    PIC X(50).


    05  ORDSTS-O REDEFINES TRAN-OUT.
        06 ORDSTS-FMT-NUM        PIC XX.
        06 OCID                  PIC S9(6) COMP-4.
        06 CUSTINFO.
           08 CFIRST               PIC X(16).
           08 CINIT                PIC X(2).
           08 CLAST                PIC X(16).
        06 CBAL                  PIC S9(11)V9(2) COMP-3.
        06 ORDERINFO.
           08 OID                  PIC S9(8) COMP-4.
           08 ODT.
              10 OENTTOD             PIC X(8).
              10 OENTTM              PIC 9(6).
           08 OCARID               PIC X(2).
           08 OLINENBR             PIC S9(2) COMP-4.
         06 OLINEINFO OCCURS 15 TIMES.
           07 DO-OLSPWH            PIC S9(4) COMP-4.
           07 DO-OLIID             PIC S9(6) COMP-4.
           07 DO-OLQTY             PIC S9(2) COMP-4.
           07 DO-OLAMNT            PIC S9(5)V9(2) COMP-3.
           07 DO-OLDLVD            PIC S9(8).
    05 STKLVL-O REDEFINES TRAN-OUT.
        06 FILLER                  PIC X(2).
        06 BLWSTK                  PIC S9(2) COMP-4.
    05 DLVRY-O REDEFINES TRAN-OUT.
        06 FILLER                  PIC X(2).
        06 DELIVERY-STATUS         PIC X.



01 ITEM-KEY-PTR  USAGE POINTER.
01 STOCK-KEY-PTR USAGE POINTER.
01 CSTMR-KEY-PTR USAGE POINTER.

01 ITEM-KEY.
  03 ITEM-KEY-DATA OCCURS 5 TIMES.
     05  KNAM   PIC X(10).
     05  KVAL   PIC S9(9) COMP-4.

01 STOCK-KEY.
```

```
   03 STOCK-KEY-DATA OCCURS 5 TIMES.
      05  KNAM   PIC X(10).
      05  KVAL   PIC S9(9) COMP-4.

01 CSTMR-KEY.
   03 CSTMR-KEY-DATA OCCURS 5 TIMES.
      05  KNAM   PIC X(10).
      05  KVAL   PIC S9(9) COMP-4.

01 ITEM-HASH-NAME   PIC X(10).
01 STOCK-HASH-NAME  PIC X(10).
01 CSTMR-HASH-NAME  PIC X(10).

01 ITEM-HASH-PTR    USAGE POINTER.
01 STOCK-HASH-PTR   USAGE POINTER.
01 CSTMR-HASH-PTR   USAGE POINTER.

01 FUNCT            PIC X(1).
01 KEYS             PIC S9(9) COMP-4.
01 RET-CODE         PIC S9(9) COMP-4.

01  RET-CODE-PTR    USAGE POINTER.

01  ITEM-DEF-PTR    USAGE POINTER.
01  STOCK-DEF-PTR   USAGE POINTER.
01  CSTMR-DEF-PTR   USAGE POINTER.

01 UPDATER                PIC X(1) VALUE "3".
01 FETCHR                 PIC X(1) VALUE "1".
01 FETCHUPDATE            PIC X(1) VALUE "2".

01 CUST-KEYS              PIC 9(1) comp-4 VALUE 3.
01 STOCK-KEYS             PIC 9(1) comp-4 VALUE 2.
01 ITEM-KEYS              PIC 9(1) comp-4 VALUE 1.

01 CMNF-INDIC-AREA.
   03 CMNF-INDIC     PIC 1 OCCURS 99 TIMES
                     INDICATOR 1.

 01   TRANSACTION-INPUT.
        06    TXN-TYPE           PIC X.
        06    JOBNAME            PIC X(10).
        06    CLIENT-INPUT       PIC X(202).
        06 NEWORD-I REDEFINES CLIENT-INPUT.
          08 CWID                PIC S9(4) COMP-4.
          08 CDID                PIC S9(4) COMP-4.
          08 CID                 PIC S9(9) COMP-4.
          08 NUMBER-OF-ITEMS     PIC 9(3) COMP-3.
          08 INPUT-LINE OCCURS 15 TIMES.
            10 OLSPWH            PIC S9(4) COMP-4.
            10 OLIID             PIC S9(9) COMP-4.
            10 OLQTY             PIC S9(3) COMP-3.
        06 PAYMENT-I REDEFINES CLIENT-INPUT.
          08 PAYMENT-TYPE        PIC X.
*                               Payment type = C if by CID
*                               Payment type = L if by CLAST
          08 HISTORY-DATA.
           09 PAY-DIST-KEY.
            10 DID               PIC S9(4) COMP-4.
            10 WID               PIC S9(4) COMP-4.
           09 PAY-CUST-KEY.
            10 CID               PIC S9(9) COMP-4.
            10 CDID              PIC S9(4) COMP-4.
            10 CWID              PIC S9(4) COMP-4.
          08 AMOUNT              PIC S9(5)V99 COMP-3.
          08 CLAST               PIC X(16).
          08 FILLER              PIC X(155).
        06 ORDSTS-I REDEFINES CLIENT-INPUT.
          08 ORDSTS-TYPE         PIC X.
          08 ORD-CUST-KEY.
           09 CID                PIC S9(9) COMP-4.
           09 DID                PIC S9(4) COMP-4.
           09 WID                PIC S9(4) COMP-4.
          08 CLAST               PIC X(16).
          08 FILLER              PIC X(165).
        06 STKLVL-I REDEFINES CLIENT-INPUT.
          08 WID                 PIC S9(4) COMP-4.
          08 DID                 PIC S9(4) COMP-4.
          08 THRESHOLD           PIC S9(4) COMP-4.
          08 BLWSTK              PIC S9(4) COMP-4.
        06 DLVRY-I REDEFINES CLIENT-INPUT.
          08 WID                 PIC S9(4) COMP-4.
          08 CARRIER             PIC XX.
          08 D-DATE              PIC 9(8).
          08 D-TIME              PIC 9(8).

77 STATUS-IND PIC X(2).


 01 CWIDR    PIC S9(4) COMP-4.
 01 CDIDR    PIC S9(4) COMP-4.
 01 CLASTR   PIC X(16).
 01 CUSTREL      PIC 9(9) COMP-4.


01 INPUT-ROW.
     05 STOCK-KEY-INPUT.
          10 OLSPWH          PIC S9(4) COMP-4.
          10 OLIID           PIC S9(9) COMP-4.
     05 OLQTY           PIC S9(3) COMP-3.

01  CHAR-HIST-DATA.
    05 DID                  PIC S99.
    05 WID                  PIC S9(4).
    05 CID                  PIC S9(6).
    05 CDID                 PIC S99.
    05 CWID                 PIC S9(4).
    05 AMOUNT               PIC S9(5)V99.


  01 ROLL-BACK-REQUIRED   PIC X(1).
  01 CMD1 PIC X(14) VALUE "DLYJOB DLY(30)".
  01 CMD-LEN  PIC S9(10)V9(5) COMP-3 VALUE 14.
  01 CONTR-1                 PIC S99 COMP-4.

D 01 TESTNUM                 PIC 99 VALUE 1.
D 01 J                       PIC S999 COMP-3.

  01 DATEINT.
     06  YY                 PIC 9(2).
     06  MMDD               PIC 9(4).

*  01 DATETIME.
*     05 TPCDATE.
*       06  TPCMD            PIC 9(4).
*       06  TPCCEN           PIC 9(2)  VALUE 19.
*       06  TPCYEAR          PIC 9(2).
*     05 CURTIME.
*       06 CURTIME6        PIC 9(6).
*       06 FILLER          PIC 9(2).
  01 DATETIME-INIT.
     05 TPCDATE-INIT         PIC X(8) VALUE '00000000'.
*    05 TPCTIME-INIT         PIC 9(6) VALUE 0.
*  01 DATETIME-CHARINIT REDEFINES DATETIME-INIT PIC X(14).
  01 DATETIME-CHARINIT REDEFINES DATETIME-INIT PIC X(8).
  01 TIME-OF-DAY PIC X(8).

   01 CNTR                   PIC 99.
   01 CUST-INDEX   PIC S9(4)  USAGE BINARY.
   01 CUSTR-INDEX   PIC S9(4)  USAGE BINARY.
   01 CUSTOMER-ARRAY.
      03  CUSTOMER-ARRAY-ELEMENT   OCCURS 100 TIMES.
          05 CID      PIC S9(9) COMP-4.
          05 CDID     PIC S9(4) COMP-4.
          05 CWID     PIC S9(4) COMP-4.
   01 TEMP-DATA2        PIC X(468).

   01 HOST-DATA.
      03  HOST-ARRAY-ELEMENT   OCCURS 100 TIMES.
          05 RRN    PIC S9(9) COMP-4.

   01 ERROR-HDLNG-PARAMETERS.
       05  NEWORDER-FILE-STATUS            PIC X(2).
       05  ITEM-FILE-STATUS                PIC X(2).
       05  STOCK-FILE-STATUS               PIC X(2).
       05  CUSTOMER-STATUS-2               PIC X(2).
       05  DISTRICT-FILE-STATUS            PIC X(2).
       05  CUSTOMER-FILE-STATUS            PIC X(2).

   01  SWITCH-AREA.
       05  SW03                             PIC 1.
          88 LOCK-OCCURRED               VALUE B"1".
          88 LOCK-OFF                    VALUE B"0".

   01 I        PIC S999 BINARY.

   01 NPOQLENB     PIC S99999 COMP-3 VALUE 670.
   01 NPOQLEN      PIC S99999 COMP-3 VALUE 227.
   01 NPOQNAME.
      05 NPOQNAMEA         PIC X(5) VALUE "NOPAY".
      05 NPOQNAMEWHS       PIC X(4).
      05 NPOQNAMEWHSA REDEFINES NPOQNAMEWHS PIC S9(4).
      05 NPOQNAMEFIL       PIC X(1) VALUE " ".
   01 NPOQLIB         PIC X(10).
   01 NPOQWAIT     PIC S99999 COMP-3 VALUE -1.

   01  TRANSACTION-CONTROL-AREA.
       05  TXN-IND                PIC X VALUE "0".

   01 CONE   PIC X(1) VALUE "1".
   01 CTWO   PIC X(2) VALUE "2".
   01 CTHREE PIC X(3) VALUE "3".
   01 CFOUR  PIC X(4) VALUE "4".

   01 N1ONE  PIC 9(1) VALUE 1.
   01 NTWO   PIC 9(2) VALUE 2.
   01 NTHREE PIC 9(3) VALUE 3.
   01 NFOUR  PIC 9(4) VALUE 4.

01  TOO-MANY.
     05 TOO-MANY-1.
        06 TOO-MANY-CID     PIC S9(9) COMP-4.
        06 TOO-MANY-DID     PIC S9(4) COMP-4.
        06 TOO-MANY-WID     PIC S9(4) COMP-4.
     06 TOO-MANY-CLAST      PIC X(16).


    EXEC SQL
      INCLUDE SQLCA
    END-EXEC.


*  SQL ERROR/WARNING Messages  *

     EXEC SQL
       WHENEVER SQLERROR CONTINUE
     END-EXEC.

     EXEC SQL
       WHENEVER SQLWARNING CONTINUE
     END-EXEC.

 01 XX  PIC S9(2) COMP-4.

 LINKAGE SECTION.
   01  FIRST-TIME PIC S9(4) COMP-4.
   01  INP-TRAN PIC X(230).
   01  OUT-TRAN PIC X(670).


 PROCEDURE DIVISION USING FIRST-TIME INP-TRAN OUT-TRAN.

 DECLARATIVES.

 HANDLE-ERROR SECTION.

      USE AFTER STANDARD ERROR PROCEDURE ON STOCK-FILE.
      USE AFTER STANDARD ERROR PROCEDURE ON CUST-REL-FILE.

 I-O-ERROR-PARA.

      IF STOCK-FILE-STATUS = "9D"
       SET LOCK-OCCURRED TO TRUE.
      IF CUSTOMER-FILE-STATUS = "9D"
       SET LOCK-OCCURRED TO TRUE.

 END DECLARATIVES.


   MAIN-LINE-ROUTINE.

       IF FIRST-TIME = 0001
       PERFORM SET-UP-ROUTINE.

       MOVE INP-TRAN TO TRANSACTION-INPUT.

       PERFORM TXN-PROC-STRT THRU
               TXN-PROC-END.


*                                                                ***
*  CMD-KEY ROUTINE PERFORMS ALL DISPLAY FILE I/O AND
*  DETERMINES WHAT TRANSACTION TYPE TO PROCESS.
*                                                                ***

 TXN-PROC-STRT.

 D    READ ISO-IFILE.
 D    MOVE ISOINUM TO TESTNUM, ISONUM of DBUG-REC, ISONUM OF
 D    DBUG-REC2.

       IF TXN-TYPE OF TRANSACTION-INPUT = "N"

               MOVE "0" TO ROLL-BACK-REQUIRED

               PERFORM NEW-ORDER-TRANSACTION
                   THROUGH NEW-ORDER-TRANSACTION-EXIT
```

```
           ELSE
           IF TXN-TYPE OF TRANSACTION-INPUT = "P"

               PERFORM PAYMENT-TRANSACTION
                   THROUGH PAYMENT-TRANSACTION-EXIT

           ELSE
           IF TXN-TYPE OF TRANSACTION-INPUT = "O"

               PERFORM ORDER-STATUS-TRANSACTION
                  THROUGH ORDER-STATUS-TRANSACTION-EXIT

           ELSE

                MOVE "1" TO TXN-IND
               GOBACK

           END-IF.

 TXN-PROC-END.
       EXIT.



 NEW-ORDER-TRANSACTION.

*    ACCEPT CURTIME FROM TIME.
*    ACCEPT DATEINT FROM DATE.
*    MOVE YY TO TPCYEAR.
*    MOVE MMDD TO TPCMD.
     CALL "gettime" USING TIME-OF-DAY.

     MOVE CWID of NEWORD-I to CWID OF CUSTOMER-REC
                           WID OF WAREHOUSE-REC
                          DWID OF DSRCD
                         NOWID OF NEWORDER-REC
                          OWID OF ORDERS-REC.

     MOVE CDID OF NEWORD-I TO
                           CDID OF CUSTOMER-REC
                            DID OF DSRCD
                          NODID OF NEWORDER-REC
                           ODID OF ORDERS-REC.

*    MOVE   CURTIME6 TO OENTTM OF NEWORD-O
*                       OENTTM OF ORDERS-REC.
*    MOVE   TPCDATE  TO OENTDT OF ORDERS-REC.
     MOVE   TIME-OF-DAY TO OENTTOD OF ORDERS-REC.


     MOVE CID OF NEWORD-I TO CID OF CUSTOMER-REC
                             OCID OF ORDERS-REC.

     MOVE NUMBER-OF-ITEMS TO OLINES OF ORDERS-REC.
     MOVE 1 TO OLOCAL OF ORDERS-REC.

     MOVE 0 TO TOTAMT.

 RE-DO-NEWORD.


* THE FOLLOWING CODE WILL ONLY BE INCLUDED WHEN THE
* FLAG IS SET AND IS FOR ISOLATION TEST USE
*
D    MOVE "NEW ORDER PRE-READ  " TO TXTDATA OF DBUG-REC.
D    MOVE 9999 TO OID OF DBUG-REC.
D    PERFORM GET-ISOLATION-DATA-1.
D    WRITE DBUGPRT-REC FORMAT IS "DBUGRCD".


     MOVE CID  OF CUSTOMER-REC TO KVAL OF CSTMR-KEY-DATA(1)
     MOVE CDID OF CUSTOMER-REC TO KVAL OF CSTMR-KEY-DATA(2)
     MOVE CWID OF CUSTOMER-REC TO KVAL OF CSTMR-KEY-DATA(3)

      CALL "qdbrunha" USING BY VALUE
           CSTMR-HASH-PTR FETCHUPDATE CUST-KEYS CSTMR-KEY-PTR
           CSTMR-DEF-PTR RET-CODE-PTR

     IF RET-CODE > 0
           IF RET-CODE = 100
              MOVE "2" TO ROLL-BACK-REQUIRED
           ELSE
               IF RET-CODE = 812
                   SET LOCK-OFF TO TRUE
                   ROLLBACK
                   GO TO RE-DO-NEWORD
               END-IF
            END-IF

     ELSE


     READ DISTRICT-FILE
     INVALID KEY
          MOVE "2" TO ROLL-BACK-REQUIRED

     NOT INVALID KEY
     MOVE DNXTOR OF DISTRICT-REC   TO OID OF ORDERS-REC
                                      NOOID OF NEWORDER-REC

     PERFORM VARYING I FROM 1 BY 1 UNTIL I > NUMBER-OF-ITEMS
           MOVE INPUT-LINE(I) TO INPUT-ROW
           MOVE OLIID OF INPUT-ROW TO IID OF ITEM-REC


     MOVE OLIID OF INPUT-ROW TO KVAL OF ITEM-KEY-DATA(1)

      CALL "qdbrunha" USING BY VALUE
           ITEM-HASH-PTR FETCHR ITEM-KEYS ITEM-KEY-PTR
           ITEM-DEF-PTR RET-CODE-PTR

           IF RET-CODE > 0
               MOVE "1" TO ROLL-BACK-REQUIRED

           ELSE

               MOVE ITEM-INFO TO  OUTPUT-ITEM-INFO(I)

               MOVE ZERO TO CONTR-1

               INSPECT IDATA OF ITEM-REC
               TALLYING CONTR-1 FOR ALL "ORIGINAL"

               MOVE STOCK-KEY-INPUT TO STOCK-IN

     MOVE STWID1             TO KVAL OF STOCK-KEY-DATA(1)
     MOVE STIID1             TO KVAL OF STOCK-KEY-DATA(2)

     CALL "qdbrunha" USING BY VALUE
           STOCK-HASH-PTR FETCHUPDATE STOCK-KEYS STOCK-KEY-PTR
           STOCK-DEF-PTR RET-CODE-PTR

           IF RET-CODE > 0
               IF RET-CODE = 100
                   MOVE "2" TO ROLL-BACK-REQUIRED
               ELSE
                   IF RET-CODE = 812
                       SET LOCK-OFF TO TRUE
                       ROLLBACK
                       GO TO RE-DO-NEWORD
                   END-IF
               END-IF
               ELSE


* THE FOLLOWING CODE WILL ONLY BE INCLUDED WHEN THE
* FLAG IS SET AND IS FOR ISOLATION TEST USE
*
D              IF TESTNUM = 7 THEN
D                IF I = 1 THEN
D                   CALL "delayme"
D                 END-IF
D              END-IF


                   SUBTRACT OLQTY OF INPUT-ROW
                                FROM STQTY OF STOCK-REC
                   IF STQTY OF STOCK-REC < 10 THEN
                        ADD 91 TO STQTY OF STOCK-REC
                   END-IF


                   IF (OLSPWH OF INPUT-ROW NOT EQUAL CWID OF
                                                    NEWORD-I)
                        MOVE 0 TO OLOCAL   OF ORDERS-REC
                        ADD  1 TO STREMORD OF STOCK-REC
                   END-IF

                   ADD 1              TO STORDRS OF STOCK-REC
                   ADD OLQTY OF INPUT-ROW TO STYTD OF STOCK-REC

         CALL "qdbrunha" USING BY VALUE
              STOCK-HASH-PTR UPDATER STOCK-KEYS STOCK-KEY-PTR
              STOCK-DEF-PTR RET-CODE-PTR
                   IF CONTR-1 > ZERO THEN
                      MOVE ZERO TO CONTR-1
                      INSPECT STDATA OF STOCK-REC
                      TALLYING CONTR-1 FOR ALL "ORIGINAL"
                   END-IF

                   IF CONTR-1 > ZERO
                   THEN MOVE "B" TO OUTPUT-BORG(I)
                   ELSE MOVE "G" TO OUTPUT-BORG(I)
                   END-IF

                MULTIPLY OLQTY OF INPUT-ROW BY IPRICE
                    GIVING OUTPUT-OLAMNT(I)

                ADD OUTPUT-OLAMNT(I) TO TOTAMT

                MOVE NORCD            TO ORDER-KEY
                MOVE I                TO OLNBR OF OLRCD
                                         OF ORDERLINE-REC
                MOVE INPUT-ROW    TO OL-INPUT
                MOVE OUTPUT-OLAMNT(I) TO OLAMNT OF OLRCD
                                         OF ORDERLINE-REC
                MOVE DIST-INFO(CDID OF NEWORD-I)
                                      TO OLDSTI OF OLRCD
                                         OF ORDERLINE-REC

                MOVE DATETIME-CHARINIT TO OLTIME-OF-DAY
                WRITE ORDERLINE-REC

                MOVE STQTY OF STOCK-REC TO OUTPUT-STQTY(I)
              END-IF
              END-IF
          END-PERFORM
          END-READ
         END-IF.

D    MOVE OID OF ORDERS-REC TO OID OF DBUG-REC.

     ADD 1 TO DNXTOR OF DISTRICT-REC.

     REWRITE DISTRICT-REC.

     WRITE ORDERS-REC.

     WRITE NEWORDER-REC.

     READ WAREHOUSE-FILE.

     IF ROLL-BACK-REQUIRED NOT EQUAL "0"
     THEN
* THE FOLLOWING CODE WILL ONLY BE INCLUDED WHEN THE
* FLAG IS SET AND IS FOR ISOLATION TEST USE
*
D         MOVE "NEW ORDER PRE-RBACK " TO TXTDATA OF DBUG-REC
D         PERFORM GET-ISOLATION-DATA-1
D         WRITE DBUGPRT-REC FORMAT IS "DBUGRCD"
D         CALL "delayme"

          ROLLBACK
* THE FOLLOWING CODE WILL ONLY BE INCLUDED WHEN THE
* FLAG IS SET AND IS FOR ISOLATION TEST USE
*
D         MOVE "NEW ORDER POST-RBACK" TO TXTDATA OF DBUG-REC
D         PERFORM GET-ISOLATION-DATA-2
D         WRITE DBUGPRT-REC FORMAT IS "DBUGRCD"
D         CLOSE DBUG-FILE
D         OPEN OUTPUT DBUG-FILE


          MOVE CLAST OF CUSTOMER-REC
                     TO CLAST OF NEWORD-O
          MOVE CCREDT OF CUSTOMER-REC
                     TO CCREDT OF NEWORD-O
          MOVE OID OF ORDERS-REC
                     TO OID OF NEWORD-O
        IF ROLL-BACK-REQUIRED = "1"
          MOVE "R" TO NEWORD-RESULT of NEWORD-O
        ELSE IF ROLL-BACK-REQUIRED = "2"
          MOVE "I" TO NEWORD-RESULT of NEWORD-O
        END-IF
        END-IF
          MOVE "0" TO ROLL-BACK-REQUIRED
          MOVE NEWORD-O TO OUT-TRAN
          GOBACK
          GO TO NEW-ORDER-TRANSACTION-EXIT
      ELSE
* THE FOLLOWING CODE WILL ONLY BE INCLUDED WHEN THE
* FLAG IS SET AND IS FOR ISOLATION TEST USE
*
D         MOVE "NEW ORDER PRE-COMMIT" TO TXTDATA OF DBUG-REC
D         PERFORM GET-ISOLATION-DATA-1
D         WRITE DBUGPRT-REC FORMAT IS "DBUGRCD"
D         CALL "delayme"
```

```
          COMMIT.

* THE FOLLOWING CODE WILL ONLY BE INCLUDED WHEN THE
* FLAG IS SET AND IS FOR ISOLATION TEST USE
*
D         MOVE "NEW ORDER POST-COMIT" TO TXTDATA OF DBUG-REC.
D         PERFORM GET-ISOLATION-DATA-2.
D         WRITE DBUGPRT-REC FORMAT IS "DBUGRCD".
D         CLOSE DBUG-FILE.
D         OPEN OUTPUT DBUG-FILE.

      MOVE "G" TO NEWORD-RESULT of NEWORD-O
      MOVE CLAST OF CUSTOMER-REC
             TO CLAST OF NEWORD-O
      MOVE CCREDT OF CUSTOMER-REC
            TO CCREDT OF NEWORD-O
      MOVE CDCT of CUSTOMER-REC TO CDCT OF NEWORD-O
      MOVE WTAX of WAREHOUSE-REC TO WTAX OF NEWORD-O
      MOVE DTAX of DISTRICT-REC TO DTAX OF NEWORD-O
      MOVE OID OF ORDERS-REC TO
                OID OF NEWORD-O.

      MOVE NEWORD-O TO OUT-TRAN.
      GOBACK.

 NEW-ORDER-TRANSACTION-EXIT.

      EXIT.


D GET-ISOLATION-DATA-1.
D       ACCEPT CURTIME FROM TIME
D       ACCEPT DATEINT FROM DATE
D       MOVE YY TO TPCYEAR
D       MOVE MMDD TO TPCMD
D       MOVE CDID OF NEWORD-I TO DID OF DBUG-REC
D       MOVE CID OF NEWORD-I TO CID OF DBUG-REC
D       MOVE cwid of NEWORD-I TO WID OF DBUG-REC
D       MOVE TPCDATE TO DATE6 OF DBUG-REC
D       MOVE CURTIME6 TO TIME6 OF DBUG-REC
D       MOVE TESTNUM TO ISONUM OF DBUG-REC
D       PERFORM VARYING J FROM 1 BY 1 UNTIL J = 16
D         MOVE OLSPWH OF INPUT-LINE(J) TO DEBUG-OLSPWH(J)
D         MOVE OLIID OF INPUT-LINE(J) TO DEBUG-OLIID(J)
D         MOVE OLQTY OF INPUT-LINE(J) TO DEBUG-OLQTY(J)
D         MOVE 0 TO DEBUG-IPRICE(J)
D       END-PERFORM.


D GET-ISOLATION-DATA-2.
D       MOVE OID OF ORDERS-REC TO OID OF DBUG-REC.
D       ACCEPT CURTIME FROM TIME
D       ACCEPT DATEINT FROM DATE
D       MOVE YY TO TPCYEAR
D       MOVE MMDD TO TPCMD
D       MOVE CDID OF NEWORD-I TO DID OF DBUG-REC
D       MOVE CID OF NEWORD-I TO CID OF DBUG-REC
D       MOVE cwid of NEWORD-I TO WID OF DBUG-REC
D       MOVE TPCDATE TO DATE6  OF DBUG-REC
D       MOVE CURTIME6 TO TIME6  OF DBUG-REC
D       MOVE TESTNUM TO ISONUM OF DBUG-REC
D       PERFORM VARYING J FROM 1 BY 1 UNTIL J = 16
D         MOVE OLSPWH OF INPUT-LINE(J) TO DEBUG-OLSPWH(J)
D         MOVE OLIID OF INPUT-LINE(J) TO DEBUG-OLIID(J)
D         MOVE OLQTY OF INPUT-LINE(J) TO DEBUG-OLQTY(J)
D         IF OLIID OF INPUT-LINE(J) <= 100000
D         MOVE OUTPUT-IPRICE(J) TO DEBUG-IPRICE(J)
D         ELSE
D         MOVE 0 TO DEBUG-IPRICE(J)
D         END-IF
D       END-PERFORM.


D GET-ISOLATION-DATA-3.
D       ACCEPT CURTIME FROM TIME
D       ACCEPT DATEINT FROM DATE
D       MOVE YY TO TPCYEAR
D       MOVE MMDD TO TPCMD
D       MOVE DID OF ORDSTS-I TO DID OF DBUG-REC
D       MOVE CID OF ORDSTS-I TO CID OF DBUG-REC
D       MOVE WID of ORDSTS-I TO WID OF DBUG-REC
D       MOVE TPCDATE TO DATE6 OF DBUG-REC
D       MOVE CURTIME6 TO TIME6 OF DBUG-REC
D       MOVE TESTNUM TO ISONUM OF DBUG-REC
D       PERFORM VARYING J FROM 1 BY 1 UNTIL J = 16
D         MOVE 0 TO DEBUG-OLSPWH(J)
D         MOVE 0 TO DEBUG-OLIID(J)
D         MOVE 0 TO DEBUG-OLQTY(J)
D         MOVE 0 TO DEBUG-IPRICE(J)
D       END-PERFORM.


D GET-ISOLATION-DATA-4.
D       ACCEPT CURTIME FROM TIME
D       ACCEPT DATEINT FROM DATE
D       MOVE YY TO TPCYEAR
D       MOVE MMDD TO TPCMD
D       MOVE DID OF ORDSTS-I TO DID OF DBUG-REC
D       MOVE CID OF ORDSTS-I TO CID OF DBUG-REC
D       MOVE WID of ORDSTS-I TO WID OF DBUG-REC
D       MOVE TPCDATE TO DATE6 OF DBUG-REC
D       MOVE CURTIME6 TO TIME6 OF DBUG-REC
D       MOVE TESTNUM TO ISONUM OF DBUG-REC
D       PERFORM VARYING J FROM 1 BY 1 UNTIL J = 16
D         MOVE DO-OLSPWH(J) TO DEBUG-OLSPWH(J)
D         MOVE DO-OLIID(J) TO DEBUG-OLIID(J)
D         MOVE DO-OLQTY(J) TO DEBUG-OLQTY(J)
D         MOVE 0 TO DEBUG-IPRICE(J)
D       END-PERFORM.




 PAYMENT-TRANSACTION.

      MOVE HISTORY-DATA OF PAYMENT-I TO HISTORY-DATA OF HIST-INFO.
      MOVE AMOUNT OF PAYMENT-I TO AMOUNT OF HIST-INFO.

*     ACCEPT DATEINT FROM DATE.
*     MOVE YY TO TPCYEAR.
*     MOVE MMDD TO TPCMD.
*     ACCEPT CURTIME FROM TIME.
      CALL "gettime" USING TIME-OF-DAY.

*     MOVE  DATETIME TO DATE-TIME OF HIST-INFO, HDT, ODT.
      MOVE  TIME-OF-DAY TO DATE-TIME OF HIST-INFO, HDT, ODT.

* THE FOLLOWING CODE WILL ONLY BE INCLUDED WHEN THE
* FLAG IS SET AND IS FOR ISOLATION TEST USE
*
D     MOVE "PAY PRE-READ  " TO TXTDATA OF DBUG-REC2.
D     MOVE CDID OF CUSTOMER-REC TO CDID OF DBUG-REC2.
D     MOVE CWID OF CUSTOMER-REC TO CWID OF DBUG-REC2.
D     MOVE CID OF PAYMENT-I TO CID OF DBUG-REC2.
D     MOVE CDID OF CUSTOMER-REC TO DID OF DBUG-REC2.
D     MOVE AMOUNT OF PAYMENT-I TO
D            PAYMNT OF DBUG-REC2.
D     MOVE WID OF PAYMENT-I TO WID OF DBUG-REC2.
D     MOVE 0 TO CBAL OF DBUG-REC2.
D     MOVE CURTIME6 TO TIME6 OF DBUG-REC2.
D     MOVE TPCDATE TO DATE6 OF DBUG-REC2.
D     WRITE DBUGPRT-REC2 FORMAT IS "DBUGRCD2".


      IF PAYMENT-TYPE EQUAL "C"
      MOVE PAY-CUST-KEY OF PAYMENT-I TO CUST-KEY
           OF CUSTOMER-REC
         PERFORM CUSTOMER-BY-NUMBER
      ELSE
      MOVE CWID OF PAYMENT-I TO CWID OF CSRCD OF CUST-REL-REC
      MOVE CDID OF PAYMENT-I TO CDID OF CSRCD OF CUST-REL-REC
      MOVE CLAST OF PAYMENT-I TO CLAST OF CUST-REL-REC


           PERFORM CUSTOMER-BY-NAME THRU CUSTOMER-BY-NAME-EXIT.

D     MOVE CBAL OF CSRCD OF CUSTOMER-REC TO CBAL OF DBUG-REC2.
      SUBTRACT AMOUNT OF PAYMENT-I
               FROM CBAL OF CSRCD OF CUSTOMER-REC.

      ADD AMOUNT OF PAYMENT-I
          TO CYTD OF CUSTOMER-REC.

      ADD 1 TO CPAYCNT OF CUSTOMER-REC.
      MOVE CUST-INFO TO PAYMENT-CUST-INFO.
      MOVE CBAL OF CSRCD OF CUSTOMER-REC TO CBAL OF PAYMNT-O.
      MOVE CCRDLM OF CSRCD OF CUSTOMER-REC
           TO CCRDLM OF PAYMNT-O.
      MULTIPLY CDCT OF CSRCD OF CUSTOMER-REC BY 100 GIVING
           CDCT OF PAYMNT-O.

      MOVE SPACES                  TO MISC-CDATA.
      IF CCREDT OF CSRCD OF CUSTOMER-REC = "BC" THEN
        MOVE CDATA OF CSRCD OF CUSTOMER-REC (1:468)
             TO TEMP-DATA2
        MOVE TEMP-DATA2
             TO CDATA OF CSRCD OF CUSTOMER-REC (32:468)

        MOVE DID OF HISTORY-DATA OF HIST-INFO
                                    TO DID OF CHAR-HIST-DATA
        MOVE WID OF HISTORY-DATA OF HIST-INFO
                                    TO WID OF CHAR-HIST-DATA
        MOVE CID OF HISTORY-DATA OF HIST-INFO
                                   TO CID OF CHAR-HIST-DATA
        MOVE CDID OF HISTORY-DATA OF HIST-INFO
                                    TO CDID OF CHAR-HIST-DATA
        MOVE CWID OF HISTORY-DATA OF HIST-INFO
                                    TO CWID OF CHAR-HIST-DATA
        MOVE AMOUNT OF HIST-INFO
                                 TO AMOUNT OF CHAR-HIST-DATA
        MOVE CHAR-HIST-DATA TO CDATA OF CUSTOMER-REC(1:31)
        MOVE CDATA OF CUSTOMER-FILE TO MISC-CDATA
      END-IF.
      IF PAYMENT-TYPE EQUAL "C"
           CALL "qdbrunha" USING BY VALUE
                CSTMR-HASH-PTR UPDATER CUST-KEYS CSTMR-KEY-PTR
                CSTMR-DEF-PTR RET-CODE-PTR
      ELSE
      MOVE CUSTOMER-REC TO CUST-REL-REC
             REWRITE CUST-REL-REC

      END-IF.
      MOVE  PAY-DIST-KEY OF PAYMENT-I TO DIST-KEY.

      READ DISTRICT-FILE INVALID KEY
           MOVE "77" TO OUTPUT-FMT-NUM-3
           ROLLBACK
           GO TO PAYMENT-TRANSACTION-EXIT.

      ADD AMOUNT OF PAYMENT-I
          TO DYTD OF DISTRICT-REC.

      REWRITE DISTRICT-REC.

      MOVE WID OF PAYMENT-I TO WID OF WAREHOUSE-REC.


      READ WAREHOUSE-FILE.

      ADD AMOUNT OF PAYMENT-I
          TO WYTD OF WAREHOUSE-REC.

      REWRITE WAREHOUSE-REC.

      MOVE CID OF CUSTOMER-REC TO
                CID OF HISTORY-REC.
      MOVE WNAME OF WAREHOUSE-REC TO
                WNAME OF HDATA OF HIST-INFO.
      MOVE DNAME OF DISTRICT-REC TO
                DNAME OF HDATA OF HIST-INFO.

      WRITE HISTORY-REC.

* THE FOLLOWING CODE WILL ONLY BE INCLUDED WHEN THE
* FLAG IS SET AND IS FOR ISOLATION TEST USE
*
D     MOVE "PAY PRE-COMMIT" TO TXTDATA OF DBUG-REC2.
D     MOVE CDID OF CUSTOMER-REC TO CDID OF DBUG-REC2.
D     MOVE CWID OF CUSTOMER-REC TO CWID OF DBUG-REC2.
D     MOVE CID OF PAYMENT-I TO CID OF DBUG-REC2.
D     MOVE CDID OF CUSTOMER-REC TO DID OF DBUG-REC2.
D     MOVE WID OF PAYMENT-I TO WID OF DBUG-REC2.
*     MOVE AMOUNT OF PAYMENT-I TO
*            PAYMNT OF DBUG-REC2.
D     ACCEPT CURTIME FROM TIME.
D     MOVE CURTIME6 TO TIME6 OF DBUG-REC2.
D     MOVE TPCDATE TO DATE6 OF DBUG-REC2.
D     WRITE DBUGPRT-REC2 FORMAT IS "DBUGRCD2".
D                   CALL "delayme".


      COMMIT.

* THE FOLLOWING CODE WILL ONLY BE INCLUDED WHEN THE
* FLAG IS SET AND IS FOR ISOLATION TEST USE
*
D     MOVE "PAY POST-COMMIT" TO TXTDATA OF DBUG-REC2.
D     MOVE CDID OF CUSTOMER-REC TO CDID OF DBUG-REC2.
D     MOVE CWID OF CUSTOMER-REC TO CWID OF DBUG-REC2.
D     MOVE CID OF PAYMENT-I TO CID OF DBUG-REC2.
D     MOVE CDID OF CUSTOMER-REC TO DID OF DBUG-REC2.
*     MOVE AMOUNT OF PAYMENT-I TO
*            PAYMNT OF DBUG-REC2.
D     MOVE WID OF PAYMENT-I TO WID OF DBUG-REC2.
```

```
D    MOVE CBAL OF CSRCD OF CUSTOMER-REC TO CBAL OF DBUG-REC2.
D    ACCEPT CURTIME FROM TIME.
D    MOVE CURTIME6 TO TIME6 OF DBUG-REC2.
D    MOVE TPCDATE TO DATE6 OF DBUG-REC2.
D    WRITE DBUGPRT-REC2 FORMAT IS "DBUGRCD2".
D    CLOSE DBUG-FILE2.
D    OPEN OUTPUT DBUG-FILE2.


     MOVE DIST-ADDRESS TO DST-DATA.
     MOVE WH-ADDRESS TO WH-DATA.
     MOVE CID OF CUSTOMER-REC TO CID OF PAYMNT-O.

     IF CCREDT OF CUSTOMER-REC = "BC"
     THEN MOVE "17" TO OUTPUT-FMT-NUM-3
     ELSE MOVE "19" TO OUTPUT-FMT-NUM-3
     END-IF.

 END-WAREHOUSE-UPDATE.
     MOVE PAYMNT-O TO OUT-TRAN.
     GOBACK.
 PAYMENT-TRANSACTION-EXIT.
     MOVE PAYMNT-O TO OUT-TRAN.
     GOBACK.

 CUSTOMER-BY-NUMBER.
        MOVE CID  OF CUSTOMER-REC TO KVAL OF CSTMR-KEY-DATA(1)
        MOVE CDID OF CUSTOMER-REC TO KVAL OF CSTMR-KEY-DATA(2)
        MOVE CWID OF CUSTOMER-REC TO KVAL OF CSTMR-KEY-DATA(3)

        CALL "qdbrunha" USING BY VALUE
             CSTMR-HASH-PTR FETCHUPDATE CUST-KEYS CSTMR-KEY-PTR
             CSTMR-DEF-PTR RET-CODE-PTR

        IF RET-CODE > 0
          IF RET-CODE = 100
            MOVE "77" TO OUTPUT-FMT-NUM-3
            ROLLBACK
            GO TO PAYMENT-TRANSACTION-EXIT
          ELSE
              IF RET-CODE = 812
                  SET LOCK-OFF TO TRUE
                  GO TO CUSTOMER-BY-NUMBER
              END-IF
           END-IF
         END-IF.

 CUSTOMER-BY-NAME.

      MOVE CWID OF CUST-REL-REC TO CWIDR.
      MOVE CDID OF CUST-REL-REC TO CDIDR.
      MOVE CLAST OF CUST-REL-REC TO CLASTR.

      EXEC SQL DECLARE C1 CURSOR FOR
      SELECT RRN(CSTMRPF)
      FROM CSTMRPF
      WHERE CLAST = :CLASTR AND CDID = :CDIDR AND
      CWID = :CWIDR
      END-EXEC.

      EXEC SQL OPEN C1
      END-EXEC.

      EXEC SQL FETCH C1 FOR 100 ROWS INTO :HOST-ARRAY-ELEMENT
      END-EXEC.
      MOVE SQLERRD OF SQLCA(3) TO XX.

      EXEC SQL CLOSE C1
      END-EXEC.

      IF XX = 0
          IF TXN-TYPE = 'P'
          MOVE "77" TO OUTPUT-FMT-NUM-3
          ROLLBACK
          GO TO payment-transaction-EXIT
          END-IF
          IF TXN-TYPE = 'O'
          MOVE "62" TO ORDSTS-FMT-NUM
          ROLLBACK
          GO TO ORDER-STATUS-TRANSACTION-EXIT
          END-IF
      END-IF

      IF XX = 100
          CALL "TOOMANY" USING CWIDR, CDIDR, CLASTR, CUSTREL
      ELSE
          COMPUTE CUST-INDEX = (XX + 1) / 2.
          MOVE HOST-ARRAY-ELEMENT(CUST-INDEX) TO CUSTREL.

 CUSTOMER-BY-NAME-EXIT.

     READ CUST-REL-FILE.
              IF LOCK-OCCURRED
                  SET LOCK-OFF TO TRUE
                  ROLLBACK
                  GO TO CUSTOMER-BY-NAME
              END-IF
     MOVE CUST-REL-REC TO CUSTOMER-REC.




 ORDER-STATUS-TRANSACTION.

* THE FOLLOWING CODE WILL ONLY BE INCLUDED WHEN THE
* FLAG IS SET AND IS FOR ISOLATION TEST USE
*
D    MOVE "ORDER STATUS PRE-READ  " TO TXTDATA OF DBUG-REC.
D    MOVE 0 TO OID OF DBUG-REC
D    PERFORM GET-ISOLATION-DATA-3.
D    WRITE DBUGPRT-REC FORMAT IS "DBUGRCD".

     MOVE WID OF ORDSTS-I TO OWID OF ORDERS-VIEW-REC.
     MOVE DID OF ORDSTS-I TO ODID OF ORDERS-VIEW-REC.
     MOVE ORD-CUST-KEY OF ORDSTS-I TO CUST-KEY OF CUSTOMER-REC.
 ORDER-STATUS-READS.
     IF ordsts-type = "C" THEN
         MOVE CID  OF CUSTOMER-REC TO KVAL OF CSTMR-KEY-DATA(1)
         MOVE CDID OF CUSTOMER-REC TO KVAL OF CSTMR-KEY-DATA(2)
         MOVE CWID OF CUSTOMER-REC TO KVAL OF CSTMR-KEY-DATA(3)

         CALL "qdbrunha" USING BY VALUE
              CSTMR-HASH-PTR FETCHUPDATE CUST-KEYS CSTMR-KEY-PTR
              CSTMR-DEF-PTR RET-CODE-PTR
         IF RET-CODE > 0
           IF RET-CODE = 100
                 MOVE "62" TO ORDSTS-FMT-NUM
                 GO TO ORDER-STATUS-TRANSACTION-EXIT
           ELSE
               IF RET-CODE = 812
                   SET LOCK-OFF TO TRUE
                   GO TO ORDER-STATUS-READS
               END-IF
             END-IF
           END-IF
         END-IF
       ELSE
         MOVE CLAST OF ORDSTS-I TO CLAST OF CUST-REL-REC
         MOVE CLAST OF CUST-REL-REC TO CLAST OF PAYMENT-I
         MOVE WID OF ORDSTS-I TO CWID OF CUST-REL-REC
         MOVE DID OF ORDSTS-I TO CDID OF CUST-REL-REC
         PERFORM CUSTOMER-BY-NAME THROUGH
             CUSTOMER-BY-NAME-EXIT
       END-IF.
       MOVE CID OF CUSTOMER-REC TO OCID OF ORDERS-VIEW-REC
                                   OCID OF ORDSTS-O.
       MOVE 999999 TO OID OF ORDERS-VIEW-REC.

       START ORDERS-VIEW KEY NOT < EXTERNALLY-DESCRIBED-KEY
       INVALID KEY
                MOVE "62" TO ORDSTS-FMT-NUM
                GO TO ORDER-STATUS-TRANSACTION-EXIT.
       READ ORDERS-VIEW PRIOR
D               IF TESTNUM = 9 THEN
D                   CALL "delayme"
D               END-IF

       MOVE OWID OF ORDERS-VIEW-REC TO
              OLWID OF ORDERLINE-VIEW-REC.
       MOVE ODID OF ORDERS-VIEW-REC TO
              OLDID OF ORDERLINE-VIEW-REC.
       MOVE OID OF ORDERS-VIEW-REC TO
              OLOID OF ORDERLINE-VIEW-REC OID OF ORDSTS-O.
       MOVE 1 TO OLNBR OF ORDERLINE-VIEW-REC.
       MOVE OLINES OF ORDERS-VIEW-REC TO
              OLINENBR OF ORDSTS-O.
       MOVE OCARID OF ORDERS-VIEW-REC TO
              OCARID OF ORDSTS-O.
*      MOVE OENTDT OF ORDERS-VIEW-REC TO
*             OENTDT OF ORDSTS-O.
       MOVE OENTTOD OF ORDERS-VIEW-REC TO
              OENTTOD OF ORDSTS-O.
*      MOVE OENTTM OF ORDERS-VIEW-REC TO
*             OENTTM OF ORDSTS-O.

       READ ORDERLINE-VIEW
       MOVE OL-INFO TO OLINEINFO(1)

       PERFORM varying I
               from 2 by 1 until I > OLINES OF ORDERS-VIEW-REC

           READ ORDERLINE-VIEW NEXT
           MOVE OL-INFO TO OLINEINFO(I)

       END-PERFORM.

         MOVE I TO ORDSTS-FMT-NUM.
D        MOVE OID OF ORRCD OF ORDERS-VIEW-REC TO OID OF DBUG-REC.
       COMMIT.

* THE FOLLOWING CODE WILL ONLY BE INCLUDED WHEN THE
* FLAG IS SET AND IS FOR ISOLATION TEST USE
*
D    MOVE "ORDER STATUS POST-COMMIT " TO TXTDATA OF DBUG-REC.
*    MOVE 0 TO OID OF DBUG-REC
D    PERFORM GET-ISOLATION-DATA-4.
D    WRITE DBUGPRT-REC FORMAT IS "DBUGRCD".
D    CLOSE DBUG-FILE.
D    OPEN OUTPUT DBUG-FILE.


       MOVE CBAL OF CSRCD OF CUSTOMER-REC TO CBAL OF ORDSTS-O.
       MOVE CFIRST OF CUSTOMER-REC TO CFIRST OF ORDSTS-O.
       MOVE CINIT OF CUSTOMER-REC TO CINIT OF ORDSTS-O.
       MOVE CLAST OF CUSTOMER-REC TO CLAST OF ORDSTS-O.

       MOVE ORDSTS-O TO OUT-TRAN.
       GOBACK.
  ORDER-STATUS-TRANSACTION-EXIT.
       MOVE ORDSTS-O TO OUT-TRAN.
       GOBACK.




  SET-UP-ROUTINE.

*      ACCEPT DATEINT FROM DATE.
*      MOVE YY TO TPCYEAR.
*      MOVE MMDD TO TPCMD.
*      IF YY < 70 THEN
*        MOVE 20 TO TPCCEN.
       CALL "gettime" USING TIME-OF-DAY.


 *OPEN FILES
       OPEN  I-O   DISTRICT-FILE
                   WAREHOUSE-FILE
                   STOCK-FILE
                   CUSTOMER-FILE
                   CUST-REL-FILE.
       OPEN INPUT  ITEM-FILE
                   ORDERLINE-VIEW
                   ORDERS-VIEW.
       OPEN EXTEND ORDERLINE-FILE.
       OPEN EXTEND ORDERS-FILE.
       OPEN EXTEND NEWORDER-FILE.
D      OPEN INPUT  ISO-IFILE.
D      OPEN OUTPUT DBUG-FILE.
D      OPEN OUTPUT DBUG-FILE2.
       OPEN EXTEND HISTORY-FILE.

       MOVE "ITEMLF"           TO ITEM-HASH-NAME.
       MOVE "IID"              TO KNAM OF ITEM-KEY-DATA(1).

       SET ITEM-HASH-PTR TO ADDRESS OF ITEM-HASH-NAME.
       SET RET-CODE-PTR TO ADDRESS OF RET-CODE.
       SET ITEM-DEF-PTR TO ADDRESS OF ITEM-DEF.
       SET ITEM-KEY-PTR TO ADDRESS OF ITEM-KEY.

       MOVE "STOCK"            TO STOCK-HASH-NAME.
       MOVE "STIID"            TO KNAM OF STOCK-KEY-DATA(2).
       MOVE "STWID"            TO KNAM OF STOCK-KEY-DATA(1).

       SET STOCK-HASH-PTR TO ADDRESS OF STOCK-HASH-NAME.
       SET RET-CODE-PTR  TO ADDRESS OF RET-CODE.
       SET STOCK-DEF-PTR TO ADDRESS OF STOCK-DEF.
       SET STOCK-KEY-PTR TO ADDRESS OF STOCK-KEY.

       MOVE "CSTMR"            TO CSTMR-HASH-NAME.
       MOVE "CID"              TO KNAM OF STOCK-KEY-DATA(1).
       MOVE "CDID"             TO KNAM OF STOCK-KEY-DATA(2).
       MOVE "CWID"             TO KNAM OF STOCK-KEY-DATA(3).
```

```
           SET CSTMR-HASH-PTR TO ADDRESS OF CSTMR-HASH-NAME.
           SET RET-CODE-PTR   TO ADDRESS OF RET-CODE.
           SET CSTMR-DEF-PTR  TO ADDRESS OF CUSTOMER-REC.
           SET CSTMR-KEY-PTR  TO ADDRESS OF CSTMR-KEY.

           MOVE SPACES TO HDATA OF HIST-INFO
                          OCARID OF ORDERS-REC.
      *    MOVE ZEROES TO OLDLVT OF ORDERLINE-REC.
      *                   OLDLVT OF ORDERLINE-REC.
           MOVE ZEROES TO OLDLVTOD OF ORDERLINE-REC.
```

NPOFEND.CL:

```
/* NPOFEND:  New Order, Payment, Order Status Start Program           */

             PGM
             DCL        VAR(&DATALIB) TYPE(*CHAR) LEN(10)
             DCL        VAR(&OBJLIB) TYPE(*CHAR) LEN(10)
             MONMSG     MSGID(CPF0000)
             RTVDTAARA  DTAARA(TPCCINFO/DATALIB) RTNVAR(&DATALIB)
             RTVDTAARA  DTAARA(TPCCINFO/OBJECTLIB) RTNVAR(&OBJLIB)
             ADDLIBLE   LIB(&DATALIB)
             ADDLIBLE   LIB(&OBJLIB)
             CHGJOB     RUNPTY(10)
             STRCMTCTL  LCKLVL(*ALL)
             RETURN
```

NPOSRVR.C:

```
#include "Common.h"
char *NOPAYOSMOD(short *, char *, char *);

main(  ) {
  char          rcv_buf[BufferSize],  snd_buf[BufferSize];
  int           sd, snd_len=670, rcv_len=230, rc;
  static short  first_time=1;

      for(;;) {

      /* recv() client's request */
      if((rc=recv(sd,rcv_buf,rcv_len,0)) < 0) {
        perror("\nrecv() failed");
        exit(-1);
        }
      else {            /* else 2  */
      if(rc==0)/* the client socket was closed */
      {
         close(sd);/* close the socket */
         exit(0);/* end the operation */
       }
        rcv_buf[rcv_len]='\0';
          if(first_time==1) {
                        NPOFEND();
                        system(
  "OVRDBF FILE(ORDLIN) TOFILE(*FILE) SHARE(*NO) SEQONLY(*YES 16)");
                        system(
  "OVRDBF FILE(NEWORD) TOFILE(*FILE) SHARE(*NO) SEQONLY(*YES 2)");
                        system(
  "OVRDBF FILE(ORDLINLF) TOFILE(*FILE) SHARE(*NO) NBRRCDS(15)");
                        system(
  "OVRDBF FILE(HSTRY) TOFILE(*FILE) SHARE(*NO) SEQONLY(*NO)");
                        system(
  "OVRDBF FILE(ORDERS) TOFILE(*FILE) SHARE(*NO) SEQONLY(*NO) WAITRCD(*NOMAX)");
                        system(
 "OVRDBF FILE(ORDERSLF) TOFILE(*FILE) SHARE(*NO) SEQONLY(*NO) WAITRCD(*NOMAX)");
                        system(
  "OVRDBF FILE(ORDERSPF) TOFILE(*FILE) SHARE(*NO) WAITRCD(5)");
                        system(
  "OVRDBF FILE(STOCKPF) TOFILE(*FILE) SHARE(*NO) WAITRCD(2)");
                        system(
  "OVRDBF FILE(STOCK) TOFILE(*FILE) SHARE(*NO) WAITRCD(2)");
                        system(
  "OVRDBF FILE(CSTMRPF) TOFILE(*FILE) SHARE(*NO) WAITRCD(5)");
                        system(
  "OVRDBF FILE(CSTMR) TOFILE(*FILE) SHARE(*NO) WAITRCD(5)");
                        system(
  "OVRDBF FILE(DSTRCT) TOFILE(*FILE) SHARE(*NO)");
                        system(
  "OVRDBF FILE(CSTMRLFNAM) TOFILE(*FILE) SHARE(*NO) NBRRCDS(5)");
                        system(
  "OVRDBF FILE(CSTMRLFCRT) TOFILE(*FILE) SHARE(*NO)");
                        system(
  "OVRDBF FILE(WRHS) TOFILE(*FILE) LVLCHK(*NO) SHARE(*NO)");
                        NOPAYOSMOD(&first_time, rcv_buf, snd_buf);
                        first_time = 0;
                       }
          else NOPAYOSMOD(&first_time, rcv_buf, snd_buf);

          if(send(sd,snd_buf,snd_len,0)<0) {
            perror("\nsend(2) failed");
            exit(-1);
            }
         }  /* else 2 */
    }  /* for loop  */
  return 0;
}
```

STKFEND.CL:

```
/* STKFEND:  Stock Level Start Program           */

             PGM
             DCL        VAR(&DATALIB) TYPE(*CHAR) LEN(10)
             DCL        VAR(&APPLIB) TYPE(*CHAR) LEN(10)
             MONMSG     MSGID(CPF0000)
             CHGJOB     RUNPTY(20)
             RTVDTAARA  DTAARA(TPCCINFO/DATALIB) RTNVAR(&DATALIB)
             RTVDTAARA  DTAARA(TPCCINFO/OBJECTLIB) RTNVAR(&APPLIB)
             ADDLIBLE   LIB(&APPLIB)
             ADDLIBLE   LIB(&DATALIB)
             STRCMTCTL  LCKLVL(*CS)
             RETURN
```

STKLVL.CBL:

```
   PROCESS NOTRUNC NORANGE.
 IDENTIFICATION DIVISION.

*
*  Stock Level Program        (Transaction 5 (PF6 of TPCC))
*
*     Files Accessed:   - DSTRCT    (Read)
*                         ORDLIN    (Read)
*                         STOCK     (Read)
*
*     Views Used:       - ORDERLINLF (C1 - SQL)
*                         STOCKLF3   (C1 - SQL)
*
*     Join Position 1:    ORDLIN
*     Join Position 2:    STOCK
*
*   This program retrieves the number of unique items ordered
*   in the last 20 orders that have a stock quantity less than
*   the threshold level entered.  The threshold will be between
*   10 and 20.
*
*   The home Warehouse ID and Library of the data queues are
*   passed to this program as parameters.  All jobs processed by
*   this program are received from a data queue and the results
*   sent out using a returning data queue.  One Stock Level
*   program operates for 10 districts (ATPCCs) of a warehouse;
*   therefore, only the district value and threshold are
*   reinitialized with each job processed.  A threshold value of
*   99 will terminate Stock Level.
*
*   Processing the Stock Level transactions consists of:
*   1.  Read input data
*   2.  Reading the DSTRCT file
*   3.  Calculating the order range
*   4.  Read the ORDERLINE and STOCK files (Avg of 200 reads)
*   5.  Count the number of records selected.
*   6.  Write to ICF file to return the answer to the user
*

    PROGRAM-ID. STKLVL.
    AUTHOR. DEPT-53G.
    INSTALLATION. IBM-ROCHESTER.
    DATE-WRITTEN. 09-03-95.
    DATE-COMPILED. 12-14-92.
 ENVIRONMENT DIVISION.
 CONFIGURATION SECTION.
    SOURCE-COMPUTER. IBM-AS400.
    OBJECT-COMPUTER. IBM-AS400.
    INPUT-OUTPUT SECTION.
    FILE-CONTROL.
      SELECT DSTRCT
             ASSIGN TO DATABASE-DSTRCT
             ORGANIZATION IS INDEXED
             ACCESS MODE IS RANDOM
             RECORD KEY IS EXTERNALLY-DESCRIBED-KEY
             WITH DUPLICATES
             FILE STATUS IS DISTRICT-FILE-STATUS.
      SELECT ORDERLINE
             ASSIGN TO DATABASE-ORDLINLF
             ORGANIZATION IS INDEXED
             ACCESS MODE IS DYNAMIC
             FILE STATUS IS ORDERLINE-FILE-STATUS
             RECORD KEY IS EXTERNALLY-DESCRIBED-KEY
             WITH DUPLICATES.
      SELECT STOCK
             ASSIGN TO DATABASE-STOCK
             ORGANIZATION IS INDEXED
             ACCESS MODE IS RANDOM
             RECORD KEY IS EXTERNALLY-DESCRIBED-KEY
             WITH DUPLICATES
             FILE STATUS IS STOCK-FILE-STATUS.
    I-O-CONTROL.
      COMMITMENT CONTROL FOR DSTRCT ORDERLINE.
*
    DATA DIVISION.
    FILE SECTION.
 FD  DSTRCT
      LABEL RECORDS ARE STANDARD
      DATA RECORD IS DSTRCT-RECORD.
 01  DSTRCT-RECORD.
      COPY DDS-DSRCD OF DSTRCT.
 FD  ORDERLINE
      LABEL RECORDS ARE STANDARD
      DATA RECORD IS ORDERLINE-RECORD.
 01  ORDERLINE-RECORD.
      COPY DDS-OLRCD OF ORDLINLF.
 FD  STOCK
      LABEL RECORDS ARE STANDARD
      DATA RECORD IS STOCK-RECORD.
 01  STOCK-RECORD.
      COPY DDS-STRCD OF STOCK.

**************************
 WORKING-STORAGE SECTION.
**************************

    01  TRAN-OUTPUT.
     05 TRAN-OUT PIC X(670).
    05 STKLVL-O REDEFINES TRAN-OUT.
        06 FILLER                  PIC X(2).
        06 BLWSTK                  PIC S9(2) COMP-4.


 01 STOCK-KEY.
   03 STOCK-KEY-DATA OCCURS 5 TIMES.
      05  KNAM   PIC X(10).
      05  KVAL   PIC S9(9) COMP-4.

 01 STOCK-HASH-PTR   USAGE POINTER.

 01 STOCK-HASH-NAME  PIC X(10).
 01 FUNCT            PIC X(1).
 01 KEYS             PIC S9(9) COMP-4.
 01 RET-CODE         PIC S9(9) COMP-4.

 01  RET-CODE-PTR    USAGE POINTER.
 01  STOCK-DEF-PTR   USAGE POINTER.
 01 STOCK-KEY-PTR USAGE POINTER.


 01 UPDATER               PIC X(1) VALUE "3".
 01 FETCHR                PIC X(1) VALUE "1".
 01 FETCHUPDATE           PIC X(1) VALUE "2".

 01 CUST-KEYS             PIC 9(1) COMP-4 VALUE 3.
 01 STOCK-KEYS            PIC 9(1) COMP-4 VALUE 2.
 01 ITEM-KEYS             PIC 9(1) COMP-4 VALUE 1.
```

```
 01   TRANSACTION-INPUT.
        06    TXN-TYPE           PIC X.
        06    JOBNAME            PIC X(10).
        06    CLIENT-INPUT       PIC X(202).
        06 STKLVL-I REDEFINES CLIENT-INPUT.
          08 WID                 PIC S9(4) COMP-4.
          08 DID                 PIC S9(4) COMP-4.
          08 THRESHOLD           PIC S9(4) COMP-4.
          08 BLWSTK              PIC S9(4) COMP-4.

 77 STATUS-IND PIC X(2).
 01  ITEM-TABLE.
     05  ITEM-TABLE-ELEMENT OCCURS 300 TIMES.
         07  ITEM-TABLE-KEY.
             09  ITEM-TABLE-IID       PIC S9(6).

 77  DISTRICT-FILE-STATUS     PIC X(2).
 77  ORDERLINE-FILE-STATUS    PIC X(2).
 77  STOCK-FILE-STATUS        PIC X(2).
*
 01  NUMBER-OF-ITEMS          PIC S9(5) COMP-4.
*
 77 DUPS              PIC S9(5) COMP-4.
 77 HIORDER           PIC S9(9) COMP-4.
 77 LOORDER           PIC S9(9) COMP-4.
 77 IN-LOOP           PIC S9(5) COMP-4.
 77 OUT-LOOP          PIC S9(5) COMP-4.

******************
 LINKAGE SECTION.
******************
   01  FIRST-TIME PIC S9(4) COMP-4.
   01  INP-TRAN PIC X(230).
   01  OUT-TRAN PIC X(670).

 PROCEDURE DIVISION USING FIRST-TIME INP-TRAN OUT-TRAN.

*******************
 START-OF-PROGRAM.
*******************

 STOCK-WATCH-PROGRAM.

       IF FIRST-TIME = 0001
       PERFORM SET-UP-ROUTINE.

     MOVE INP-TRAN TO TRANSACTION-INPUT.

     MOVE WID OF STKLVL-I TO DWID, OLWID, STWID.

     MOVE DID OF STKLVL-I TO DID OF DSRCD, OLDID.

     READ DSTRCT.

*  Select all Detail Order Lines for the Last 20 Orders
*  that are below the threshold and also match a unique
*  stock item.

     SUBTRACT 1  FROM DNXTOR GIVING HIORDER.
     SUBTRACT 20 FROM DNXTOR GIVING LOORDER.
 READ-ORDERLINE-FILE.

     MOVE LOORDER  TO OLOID OF ORDERLINE-RECORD.

     MOVE 0 TO NUMBER-OF-ITEMS.

     READ ORDERLINE.

     MOVE OLIID TO STIID.

     MOVE WID OF STKLVL-I    TO KVAL OF STOCK-KEY-DATA(1)
     MOVE STIID              TO KVAL OF STOCK-KEY-DATA(2)

      CALL "qdbrunha" USING BY VALUE
           STOCK-HASH-PTR FETCHR STOCK-KEYS STOCK-KEY-PTR
           STOCK-DEF-PTR RET-CODE-PTR
     IF STQTY IS LESS THAN THRESHOLD
         ADD 1 TO NUMBER-OF-ITEMS
         MOVE STIID TO ITEM-TABLE-IID(NUMBER-OF-ITEMS)
     END-IF.

 READ-ORDERLINE-FILE-NEXT.

     READ ORDERLINE NEXT AT END GO TO SORT-ITEM-TABLE
     END-READ

     IF OLDID NOT EQUAL TO DID OF STKLVL-I
        GO TO SORT-ITEM-TABLE
     END-IF

     IF (OLOID > HIORDER) THEN GO TO SORT-ITEM-TABLE
        ELSE
          MOVE OLIID TO STIID

     MOVE WID OF STKLVL-I    TO KVAL OF STOCK-KEY-DATA(1)
     MOVE STIID              TO KVAL OF STOCK-KEY-DATA(2)

      CALL "qdbrunha" USING BY VALUE
           STOCK-HASH-PTR FETCHR STOCK-KEYS STOCK-KEY-PTR
           STOCK-DEF-PTR RET-CODE-PTR
         IF STQTY IS LESS THAN THRESHOLD
           ADD 1 TO NUMBER-OF-ITEMS
           MOVE STIID TO ITEM-TABLE-IID(NUMBER-OF-ITEMS)
         END-IF
     END-IF

     GO TO READ-ORDERLINE-FILE-NEXT.

 SORT-ITEM-TABLE.
       COMMIT

     PERFORM DISTINCT-SORT.

 DISPLAY-ANSWER.

     MOVE STKLVL-O to OUT-TRAN.

     GOBACK.


 DISTINCT-SORT.
      IF NUMBER-OF-ITEMS > 0
       MOVE 1 TO BLWSTK OF STKLVL-O
       PERFORM VARYING OUT-LOOP FROM 2 BY 1 UNTIL OUT-LOOP
             IS GREATER THAN NUMBER-OF-ITEMS
         MOVE 0 TO DUPS
         PERFORM VARYING IN-LOOP FROM 1 BY 1 UNTIL IN-LOOP
             IS EQUAL TO OUT-LOOP
           IF ITEM-TABLE-ELEMENT(IN-LOOP) =
             ITEM-TABLE-ELEMENT(OUT-LOOP)
             ADD 1 TO DUPS
             END-IF
           END-PERFORM
         IF DUPS = 0
                ADD 1 TO BLWSTK OF STKLVL-O
                END-IF
           END-PERFORM
           ELSE MOVE 0 TO BLWSTK OF STKLVL-O.
         MOVE STKLVL-O to OUT-TRAN.

         GOBACK.

       SET-UP-ROUTINE.

             OPEN INPUT DSTRCT
                  INPUT STOCK
                  INPUT ORDERLINE.


         MOVE "STOCK"            TO STOCK-HASH-NAME.
         MOVE "STWID"            TO KNAM OF STOCK-KEY-DATA(1).
         MOVE "STIID"            TO KNAM OF STOCK-KEY-DATA(2).

         SET STOCK-HASH-PTR  TO ADDRESS OF STOCK-HASH-NAME.
         SET RET-CODE-PTR    TO ADDRESS OF RET-CODE.
         SET STOCK-DEF-PTR   TO ADDRESS OF STOCK-RECORD.
         SET STOCK-KEY-PTR   TO ADDRESS OF STOCK-KEY.
```

STKSRVR.C:

```
#include "Common.h"
char *STKLVL(short *, char *, char *);

main(  ) {
  char          rcv_buf[BufferSize],  snd_buf[BufferSize];
  int           sd, snd_len=670, rcv_len=230, rc;
  static short  first_time=1;

      for(;;) {

      /* recv() client's request */
      if((rc=recv(sd,rcv_buf,rcv_len,0)) < 0) {
        perror("\nrecv() failed");
        exit(-1);
        }
      else {            /* else 2  */
         if(rc==0)/* the client socket was closed */
         {
            close(sd);/* close the socket */
            exit(0);/* end the operation */
         }
        rcv_buf[rcv_len]='\0';
         if(first_time==1) {
                          STKFEND();
                          system(
  "OVRDBF FILE(ORDLINPF) TOFILE(ORDLINPF) NBRRCDS(16) LVLCHK(*NO) \
   SEQONLY(*YES 16)");
                          system(
  "OVRDBF FILE(ORDLINLF) TOFILE(ORDLINLF) NBRRCDS(16) LVLCHK(*NO) \
   SEQONLY(*YES 16)");
                          system(
  "OVRDBF FILE(ORDLIN) TOFILE(ORDLIN) NBRRCDS(16) LVLCHK(*NO) SEQONLY(*YES 16)"
                          );
                          system(
             "OVRDBF FILE(DSTRCT) TOFILE(DSTRCT) LVLCHK(*NO) SHARE(*YES)");
                          system(
"OVRDBF FILE(STOCKPF) TOFILE(STOCKPF) WAITRCD(2000) LVLCHK(*NO) SHARE(*YES)");
                          system(
             "OVRDBF FILE(STOCK) TOFILE(STOCK) LVLCHK(*NO)");
                          STKLVL(&first_time, rcv_buf, snd_buf);
                          first_time = 0;
                         }
          else STKLVL(&first_time, rcv_buf, snd_buf);

          if(send(sd,snd_buf,snd_len,0)<0) {
            perror("\nsend(2) failed");
            exit(-1);
            }
         }  /* else 2 */
   }  /* for loop  */
  return 0;
}
```

STRSRVR.C:

```
#include "Common.h"
void spawnjobs(int portnum, char *pgmpath);

const char FORMATbc=
   "\nUsage: call strsrvr "
   "{<Transaction>}"
   "\n\n\tWhere:"
   "\n\tTransaction bneworder payment ordsts delivery stklvlc.";

void main(int argc, char *argvbc) {

     if(strcmp(argv[1], "neworder")==0)
       spawnjobs(4000, "/usr/bin/socknsvr");
else if(strcmp(argv[1], "payment")==0)
       spawnjobs(5000, "/usr/bin/sockpsvr");
else if(strcmp(argv[1], "ordsts")==0)
       spawnjobs(6000, "/usr/bin/sockosvr");
else if(strcmp(argv[1], "delivery")==0)
       spawnjobs(7000, "/usr/bin/sockdsvr");
else if(strcmp(argv[1], "stklvl")==0)
       spawnjobs(8000, "/usr/bin/sockssvr");


  exit(0);
}

void spawnjobs(int portnum, char *pgmpath)
{

  pid_t                child;
  struct sockaddr_in   sin;
  struct inheritance   spw_inherit;
  char                 *dumy=NULL;
```

```
  int                  port=4000, length, sd,sd2, i, help=0, on=1;

  Qp0zInitEnv();
  sd=Listen(portnum);
  memset(&spw_inherit,0x00,sizeof(spw_inherit));

  for(;;) {
    length=sizeof(sin);

    if((sd2=accept(sd,(struct sockaddr *)&sin,&length))<0) {
      perror("accept() failed");
      close(sd);
      exit(-1);
    }


    if((child=spawn(
                    pgmpath,
                    1,
                    &sd2,
                    &spw_inherit,
                    &dumy,
                    &dumy
                    ))<0) {
      perror("spawn() failed");
      close(sd);
      exit(-1);
    }

    close(sd2);
  }
  exit(0);
}
```

Appendix E. RTE Scripts

RTE Parameters

```
RTE RUN-PROFILE (TO BE SENT TO AUDITOR)
---------------------------------

Customer Lastname Constant = 117

Transaction Weights
-------------------
Payment: 43.060099
Order Status: 4.0300099
Delivery: 4.0300099
Stock Level: 4.0300099

DELAY CONSTANTS
-----------
Menu Delay= .10000099
Response Delay= .10000099

Think Times
---------------
NEW ORDER: 12.030099
PAYMENT: 12.030099
ORDER STATUS: 10.060099
DELIVERY: 5.0400099
STOCK LEVEL: 5.0400099

SLAVE #0
-------
RTE MACHINE: 9.5.109.50
CONNECTS TO CLIENT: 9.5.109.39
# OF USERS: 1000

SLAVE #1
-------
RTE MACHINE: 9.5.109.52
CONNECTS TO CLIENT: 9.5.109.39
# OF USERS: 1000

SLAVE #2
-------
RTE MACHINE: 9.5.109.67
CONNECTS TO CLIENT: 9.5.109.39
# OF USERS: 1000

SLAVE #3
-------
RTE MACHINE: 9.5.109.68
CONNECTS TO CLIENT: 9.5.109.39
# OF USERS: 1000

SLAVE #4
-------
RTE MACHINE: 9.5.109.69
CONNECTS TO CLIENT: 9.5.109.39
# OF USERS: 1000

SLAVE #5
-------
RTE MACHINE: 9.5.109.70
CONNECTS TO CLIENT: 9.5.109.39
# OF USERS: 1000

SLAVE #6
-------
RTE MACHINE: 9.5.110.50
CONNECTS TO CLIENT: 9.5.110.43
# OF USERS: 1000

SLAVE #7
-------
RTE MACHINE: 9.5.110.52
CONNECTS TO CLIENT: 9.5.110.43
# OF USERS: 1000

SLAVE #8
-------
RTE MACHINE: 9.5.110.67
CONNECTS TO CLIENT: 9.5.110.43
# OF USERS: 1000

SLAVE #9
-------
RTE MACHINE: 9.5.110.68
CONNECTS TO CLIENT: 9.5.110.43
# OF USERS: 1000

SLAVE #10
-------
RTE MACHINE: 9.5.110.69
CONNECTS TO CLIENT: 9.5.110.43
# OF USERS: 1000

SLAVE #11
-------
RTE MACHINE: 9.5.110.70
CONNECTS TO CLIENT: 9.5.110.43
# OF USERS: 1000

SLAVE #12
-------
RTE MACHINE: 9.5.111.50
CONNECTS TO CLIENT: 9.5.111.57
# OF USERS: 1000

SLAVE #13
-------
RTE MACHINE: 9.5.111.52
CONNECTS TO CLIENT: 9.5.111.57
# OF USERS: 1000

SLAVE #14
-------
RTE MACHINE: 9.5.111.67
CONNECTS TO CLIENT: 9.5.111.57
# OF USERS: 1000

SLAVE #15
-------
RTE MACHINE: 9.5.111.68
CONNECTS TO CLIENT: 9.5.111.57
# OF USERS: 1000

SLAVE #16
-------
RTE MACHINE: 9.5.111.69
CONNECTS TO CLIENT: 9.5.111.57
# OF USERS: 1000

SLAVE #17
-------
RTE MACHINE: 9.5.111.70
CONNECTS TO CLIENT: 9.5.111.57
# OF USERS: 1000

SLAVE #18
-------
RTE MACHINE: 9.5.111.71
CONNECTS TO CLIENT: 9.5.111.51
# OF USERS: 1000

SLAVE #19
-------
RTE MACHINE: 9.5.111.72
CONNECTS TO CLIENT: 9.5.111.51
# OF USERS: 1000

SLAVE #20
-------
RTE MACHINE: 9.5.111.74
CONNECTS TO CLIENT: 9.5.111.51
# OF USERS: 1000

SLAVE #21
-------
RTE MACHINE: 9.5.111.75
CONNECTS TO CLIENT: 9.5.111.51
# OF USERS: 1000

SLAVE #22
-------
RTE MACHINE: 9.5.111.76
CONNECTS TO CLIENT: 9.5.111.51
# OF USERS: 1000

SLAVE #23
-------
RTE MACHINE: 9.5.111.77
CONNECTS TO CLIENT: 9.5.111.51
# OF USERS: 1000

SLAVE #24
-------
RTE MACHINE: 9.5.110.71
CONNECTS TO CLIENT: 9.5.110.125
# OF USERS: 1000

SLAVE #25
-------
RTE MACHINE: 9.5.110.72
CONNECTS TO CLIENT: 9.5.110.125
# OF USERS: 1000

SLAVE #26
-------
RTE MACHINE: 9.5.110.74
CONNECTS TO CLIENT: 9.5.110.125
# OF USERS: 1000

SLAVE #27
-------
RTE MACHINE: 9.5.110.75
CONNECTS TO CLIENT: 9.5.110.115
# OF USERS: 1000

SLAVE #28
-------
RTE MACHINE: 9.5.110.76
CONNECTS TO CLIENT: 9.5.110.115
# OF USERS: 1000

SLAVE #29
-------
RTE MACHINE: 9.5.110.77
CONNECTS TO CLIENT: 9.5.110.115
# OF USERS: 1000

SLAVE #30
-------
RTE MACHINE: 9.5.109.71
CONNECTS TO CLIENT: 9.5.109.133
# OF USERS: 1000

SLAVE #31
-------
RTE MACHINE: 9.5.109.72
CONNECTS TO CLIENT: 9.5.109.133
# OF USERS: 1000

SLAVE #32
-------
RTE MACHINE: 9.5.109.74
CONNECTS TO CLIENT: 9.5.109.133
# OF USERS: 1000

SLAVE #33
-------
RTE MACHINE: 9.5.109.75
CONNECTS TO CLIENT: 9.5.109.133
# OF USERS: 1000

SLAVE #34
-------
RTE MACHINE: 9.5.109.76
CONNECTS TO CLIENT: 9.5.109.133
# OF USERS: 500
```

Appendix F. 180-Day DASD Requirements

The appendix documents the information required to calculate the 180-Day DASD requirements for the TPC Benchmark C measurements on the IBM AS/400e server 740-2070 system. Also included is the calculation used to determine the amount of DASD required for the 8 hours of journal space.

Section "IBM AS/400e server 740-2070 -- 180-Day DASD Requirements" shows the values in the calculation of hte 180-Day Space. Also used were the formulas provided in Clause 4.2.3 of the TPC Benchmark C Standard Specification. These formulas are:

Daily-Growth	=	{dynamic-space/(W*62.5)} *tpmC
Daily-Spread	=	MAX (0, Free-Space - 1.5*Daily-Growth)
180-Day-Space	=	Static-Space + 180*(Daily-Growth+Daily-Spread)

IBM AS/400 server 740-2070--180-Day DASD Requirements

Static Files	Number of Warehous	3600		tpmC	43419.15	
File name	Customer	District	Item	New Order	Stock	Warehouse
Number of Records	108000000	36000	100000	32400000	360000000	3600
Data Space	72145252352	3678208	9445376	356433920	110521069568	401408
Index Space	1745903616	663552	1843200	591421440	5790785536	106496
Add'l Index	6384803840			12288	12288	
Add'l Index	12288					

Dynamic Files			
File name	History	Orderline	Orders
Initial Records	108000000	1080018233	108000000
Size Per Record	49.0018765432	55.0018654815	26.0011298765
Initial Data Space	5292202667	59403017569	2808122027
Initial Index Space		28962754560	3546570752
Add'l Index		12288	2129682432
Add'l Index			
Configured Space	6350643200	71282417664	3369746432
Free Space	1058440533	11879400095	561624405

Static Space	242068457677
Dynamic Space	67503342262
Free Space	13499465034
Daily Growth	13026389970
Daily Spread	0
180 Day Space	2586818652229
System Software	1941358080
SubTotal	2588760010309
Journal Space	294692760550
Total DASD Required	2883452770859
DASD Priced	2926920000000
Difference	43467229141

IBM AS/400 server 740-2070--Journal DASD Requirements

To determine the amount of DASD required for the 8 hours of journal, a series of tests were run for an extended period. Through these tests it was determined that an average of 6,607,272 bytes of DASD are required per tpmC.

Appendix G. Third Party Quotes

PAGE.01 408 232 2585 MAY 25 '99 10:38
TOTAL P.01

Software House International Pricing Proposal	Quotation #MO-990518-87674 05/18/99

IBM
Karl Huppler
Qote Good for Ninety Days

Phone: Fax: 507-253-7743

SHI Account Exec: Eddie Loomis
Telephone : (800) 52 - SOFTWare
Fax : (908) 805 - 0818

Reference:

Product	Part #	Qty	List	Your Price	Total
10BT+1 8 Port Hub	Z99552	5449		$27.55	$150,119.95
Total					$150,119.95

Additional Comments:

MAY-25-1999 08:06 SOFTWARE HOUSE INTL 408 232 2585 P.01/01

BEA ENTERPRISE MIDDLEWARE SOLUTIONS

May 20, 1999

Mr. Karl Huppler
Bldg. 006-2
Dept. 53 GA
3605 Highway 52 North
Rochester, MN 55901
Phone: 5072534091
FAX: 507 2537743

Dear Mr. Huppler

Per your request I am enclosing the pricing information regarding TUXEDO 6.x that you requested. This pricing applies to Tuxedo 6.1, 6.2, 6.3, 6.4 and 6.5. Please note that Tuxedo 6.5 is our most recent version of Tuxedo but that all 6.x releases are generally available. Core functionality services pricing is appropriate for your activities. As per the table below, server systems are classified in one of 5 tiers based on CPU type and capacity. The Invader 9406-170 is a tier 1 system. This quote is valid for 90 days from the date of issue 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 best 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 of the server and uses the same five tier computer classification as TUXEDO 6.x. 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 and pay the appropriate per server license fees.

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 workstations and servers	Unlimited	$3,000.00	$480.00	$690.00
Tier 2 -- PC Servers with 3 or 4 CPUs, Midrange RISC servers and workstations	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) Multiprocessor Systems.	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

	Tier 1	Tier 2	Tier 3	Tier 3	Tier 4	Tier 5
Platform	Class 2	Class 3	Class 4	Class 5	Class 6	Class 7
IBM AS400	Invader (9406-170)	Models 20s, 30s, 40s, 200, 300, 400, 600, S10, 150	Models 310, 320, 500, 510, 520, 620, S20, 50s	Model 530, 640, S30, S40, 53s	Model 650	

Very Truly Yours,

Lewis D. Brentano,
Director, Market Planning

Appendix H. Auditor Letter

INFO SIZING

TPC TRANSACTION PROCESSING PERFORMANCE COUNCIL CERTIFIED AUDITOR

Test Sponsor: Karl R. Huppler
Advisory Programmer
IBM Corp Dept 53G
3605 Highway 52 N.
Rochester, MN 55901

May 27, 1999

I verified the TPC Benchmark™ C performance of the following configuration:

Platform: AS/400e 740 Model 2070 c/s
DataBase Manager: DB2/400 Integrated Relational Database V4 R3
Operating System: OS/400 Version 4 Release 3
Transaction Manager: Tuxedo Version 6.4

The results were:

CPU's	Memory	Disks	NewOrder 90% Response Time	tpmC
Server: AS/400e 740 Model 2070				
12 x 262 MHz PowerPC (9406 740-2070)	40 GB	304 x 8.5 GB 36 x 17.5 GB	0.4 Seconds	43,419.15
Six Clients (see note), specification for each AS/400e 9406-170 2388 priced client				
2 x 255 MHz PowerPC (9406 170-2388)	3.5 GB	10 x 8.5 GB	n/a	n/a

In my opinion, these performance results were produced in compliance with the TPC 3.3 requirements for the benchmark. The following verification items were given special attention:

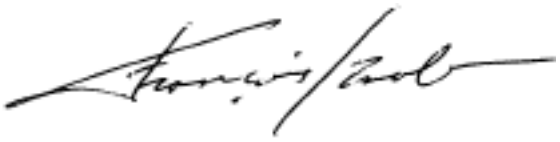
- The transactions were correctly implemented
- The database records were the proper size
- The database was properly scaled and populated
- The ACID properties were met
- Input data was generated according to the specified percentages
- The transaction cycle times included the required keying and think times

- The reported response times were correctly measured.
- At least 90% of all delivery transactions met the 80 Second completion time limit
- All 90% response times were under the specified maximums
- The measurement interval was representative of steady state conditions
- The reported measurement interval was 20 minutes
- At least 5 file synchronization cycles occurred during the measurement interval
- Measurement repeatability was verified
- The 180 day storage requirement was correctly computed
- The system pricing was verified for major components and maintenance

Additional Audit Notes:

The tested configuration included (5) priced clients model AS/400e 9406-170 2388 (3.5 GB) and (2) non-priced clients model AS/400e 9406-170 2388 (3.5 GB). The priced configuration includes (6) AS/400e 9406-170 2388 (3.5 GB). Based on data collected on all the individual priced and non priced clients, it is my opinion that this substitution would have no negative effect on the reported performance.

Respectfully Yours,



François Raab
President

AS/400e 740/2070

1373 North Franklin Street • Colorado Springs, CO 80903-2527 • Office : 719/473-7555 • Fax : 719/473-7554